Guida per l'utente

AWS CodePipeline

Versione API 2015-07-09

AWS CodePipeline: Guida per l'utente

Table of Contents

Che cos'è AWS CodePipeline?

AWS CodePipeline è un servizio di distribuzione continua che puoi utilizzare per modellare, visualizzare e automatizzare i passaggi necessari per il rilascio del software. È possibile modellare e configurare rapidamente le diverse fasi di un processo di rilascio del software. CodePipeline automatizza i passaggi necessari per rilasciare continuamente le modifiche al software. Per informazioni sui prezzi di CodePipeline, vedi Prezzi.

Argomenti

Integrazione e distribuzione continua

CodePipeline è un servizio di fornitura continua che automatizza la creazione, il test e l'implementazione del software in produzione.

La distribuzione continua è una metodologia di sviluppo software in cui il processo di rilascio è automatizzato. Ogni modifica software viene automaticamente compilata, testata e distribuita in un ambiente di produzione. Una persona, un test automatico o una regola di business decide quando eseguire il push finale in produzione. Anche se ogni modifica software riuscita può essere immediatamente rilasciata per la produzione con la distribuzione continua, non tutte le modifiche devono essere rilasciate immediatamente.

L'integrazione continua è una pratica di sviluppo software in cui i membri di un team utilizzano un sistema di controllo delle versioni e spesso integrano il proprio lavoro nella stessa posizione, ad esempio in una filiale principale. Ogni modifica viene compilata e verificata per rilevare gli errori di

integrazione il più rapidamente possibile. L'integrazione continua è incentrata sulla compilazione e il test automatico del codice, in confronto alla distribuzione continua che automatizza l'intero processo di rilascio del software fino alla produzione.

Per ulteriori informazioni, consulta Practicing Continuous Integration and Continuous Delivery on AWS: Accelerating Software Delivery with. DevOps

Puoi utilizzare la CodePipeline console, il AWS Command Line Interface (AWS CLI), gli AWS SDK o qualsiasi combinazione di questi per creare e gestire le tue pipeline.

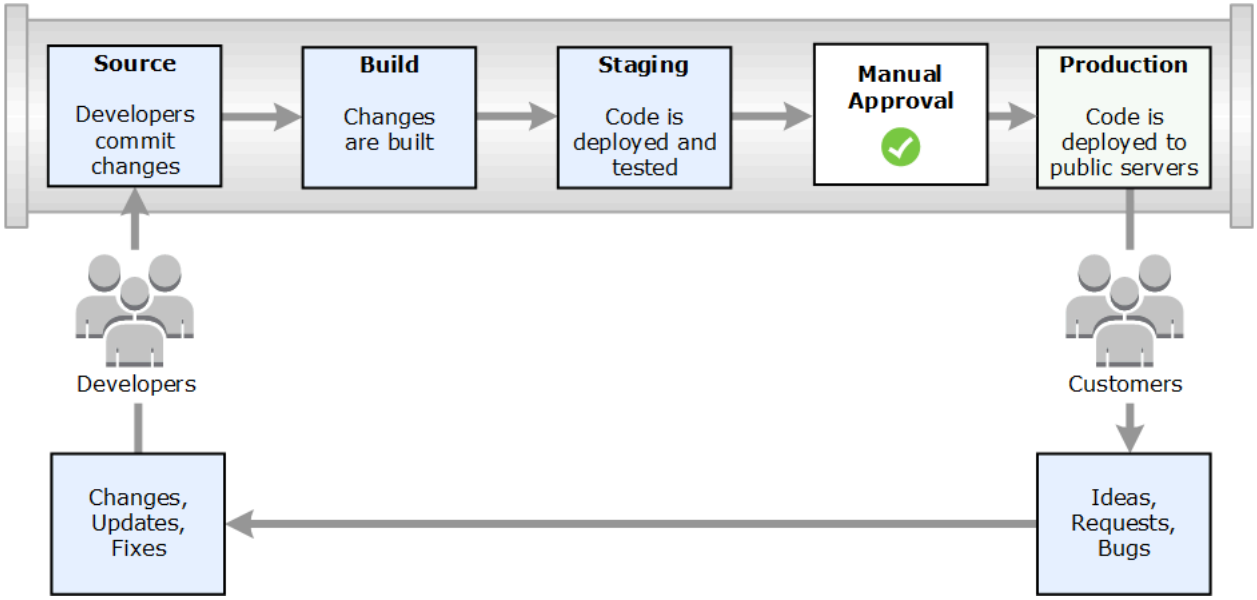
Cosa posso fare con? CodePipeline

Puoi CodePipeline utilizzarlo per aiutarti a creare, testare e distribuire automaticamente le tue applicazioni nel cloud. Nello specifico, puoi eseguire le operazioni seguenti:

- Automatizza i processi di rilascio: automatizza CodePipeline completamente il processo di rilascio dall'inizio alla fine, a partire dal repository di origine fino alla compilazione, al test e alla distribuzione. Puoi impedire lo spostamento delle modifiche in una pipeline includendo un'operazione di approvazione manuale in una fase qualsiasi, ad eccezione della fase di origine. Puoi rilasciare quando vuoi, come vuoi, sui sistemi che vuoi, attraverso una o più istanze.
- Stabilisci un processo di rilascio coerente: definisci una serie coerente di passaggi per ogni modifica del codice. CodePipeline esegue ogni fase del rilascio in base ai tuoi criteri.
- Accelera la distribuzione migliorando la qualità: puoi automatizzare il processo di rilascio per consentire agli sviluppatori di testare e rilasciare il codice in modo incrementale e accelerare il rilascio di nuove funzionalità per i propri clienti.
- Usa i tuoi strumenti preferiti: puoi incorporare nella pipeline gli strumenti di origine, compilazione e distribuzione esistenti. Per un elenco completo degli strumenti Servizi AWS di terze parti attualmente supportati da CodePipeline, consultaIntegrazioni di prodotti e servizi con CodePipeline.
- Visualizza i progressi a colpo d'occhio: puoi esaminare lo stato in tempo reale delle tue pipeline, controllare i dettagli di eventuali avvisi, riprovare le fasi o le azioni non riuscite, visualizzare i dettagli sulle revisioni dei sorgenti utilizzate nell'ultima esecuzione della pipeline in ogni fase e rieseguire manualmente qualsiasi pipeline.
- Visualizza i dettagli della cronologia delle pipeline: puoi visualizzare i dettagli sulle esecuzioni di una pipeline, inclusi gli orari di inizio e fine, la durata dell'esecuzione e gli ID di esecuzione.

Una rapida occhiata a CodePipeline

Il diagramma seguente mostra un esempio di processo di rilascio che utilizza CodePipeline.



In questo esempio, quando gli sviluppatori eseguono modifiche a un repository di origine, rileva CodePipeline automaticamente le modifiche. Queste modifiche vengono compilate e, se configurati, vengono eseguiti i test. Dopo aver completato i test, il codice compilato viene distribuito ai server della gestione temporanea per il test. Dal server di staging, CodePipeline esegue più test, come test di integrazione o di carico. Una volta completati con successo tali test e dopo l'approvazione di un'azione di approvazione manuale aggiunta alla pipeline, CodePipeline distribuisce il codice testato e approvato nelle istanze di produzione.

CodePipeline può distribuire applicazioni su istanze EC2 utilizzando, o. CodeDeploy AWS Elastic Beanstalk AWS OpsWorks Stacks CodePipeline può anche distribuire applicazioni basate su container ai servizi utilizzando Amazon ECS. Gli sviluppatori possono anche utilizzare i punti di integrazione forniti CodePipeline per collegare altri strumenti o servizi, tra cui servizi di compilazione, fornitori di test o altri obiettivi o sistemi di distribuzione.

Una pipeline può essere semplice o complessa, come richiede il processo di rilascio.

Come posso iniziare con CodePipeline?

Per iniziare con CodePipeline:

1. Scopri come CodePipeline funziona leggendo la CodePipeline concetti sezione.
2. Preparati all'uso CodePipeline seguendo i passaggi riportati di seguitoGuida introduttiva con CodePipeline.

3. Sperimenta CodePipeline seguendo i passaggi dei CodePipeline tutorial tutorial.
4. Utilizzalo CodePipeline per i tuoi progetti nuovi o esistenti seguendo i passaggi riportati di seguito. Creare una pipeline in CodePipeline

CodePipeline concetti

La modellazione e la configurazione del processo di rilascio automatico sono più semplici se si comprendono i concetti e i termini utilizzati in. AWS CodePipeline Ecco alcuni concetti da conoscere durante l'uso. CodePipeline

Per un esempio di DevOps pipeline, vediDevOps esempio di pipeline.

I seguenti termini vengono utilizzati in CodePipeline:

Argomenti

- Pipeline
- Esecuzioni pipeline
- Operazioni sullo stage
- Esecuzioni di operazioni
- Tipi di esecuzione
- Tipi di operazione
- Artifacts
- Revisioni delle origini
- Trigger
- Variables

Pipeline

Una pipeline è una struttura di flusso di lavoro che descrive le fasi del processo di rilascio delle modifiche software. Ogni pipeline è composta da una serie di fasi.

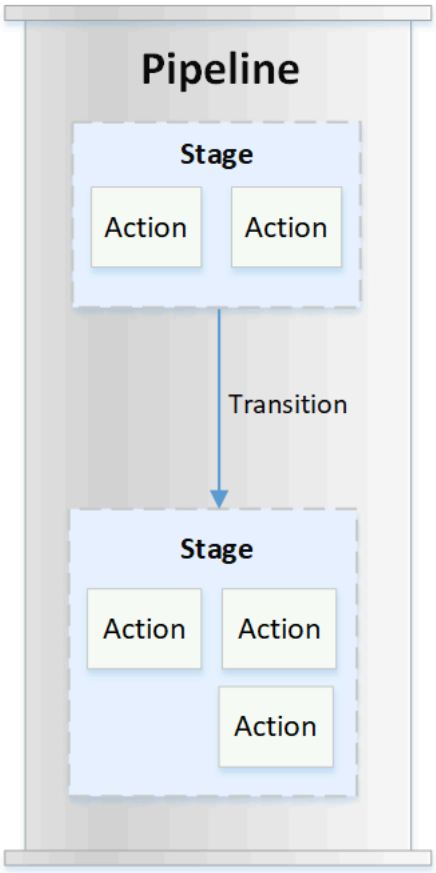
Stage

Una fase è un'unità logica che è possibile utilizzare per isolare un ambiente e limitare il numero di modifiche simultanee in tale ambiente. Ogni fase contiene operazioni eseguite sugli

artefattidell'applicazione. Il codice sorgente è un esempio di un artefatto. Una fase potrebbe essere una fase di compilazione, in cui viene creato il codice sorgente e vengono eseguiti i test. Può anche essere una fase di distribuzione, in cui il codice viene distribuito in ambienti runtime. Ogni fase è composta da una serie di azioni seriali o parallele.

Transizioni

Una transizione è il punto in cui l'esecuzione di una pipeline passa alla fase successiva della pipeline. È possibile disabilitare la transizione in ingresso di uno stage per impedire che le esecuzioni entrino in quella fase e quindi abilitare la transizione per consentire alle esecuzioni di continuare. Quando più di un'esecuzione arriva a una transizione disabilitata, solo l'esecuzione più recente continua alla fase successiva quando la transizione è abilitata. Ciò significa che le esecuzioni più recenti continuano a sostituire le esecuzioni in attesa mentre la transizione è disabilitata e quindi, dopo che la transizione è attivata, l'esecuzione che continua è l'esecuzione sostitutiva.



Azioni

Un'azione è un insieme di operazioni eseguite sul codice dell'applicazione e configurate in modo che le azioni vengano eseguite nella pipeline in un punto specificato. Ciò può includere elementi

come un'azione di origine da una modifica del codice, un'azione per la distribuzione dell'applicazione nelle istanze e così via. Ad esempio, una fase di distribuzione potrebbe contenere un'azione di distribuzione che distribuisce codice a un servizio di elaborazione come Amazon EC2 o. AWS Lambda

I tipi di CodePipeline azione validi sono`source`,`build`,, `testdeploy`, `approval` e. `invoke` Per un elenco dei provider di operazioni, consulta Tipi di azioni e provider validi in CodePipeline .

Le azioni possono essere eseguite in serie o in parallelo. Per informazioni sulle azioni seriali e parallele in una fase, consulta le `runOrder` informazioni in Requisiti della struttura delle azioni.

Esecuzioni pipeline

Un'esecuzione è un insieme di modifiche rilasciate da una pipeline. Ogni esecuzione della pipeline è univoca e ha un proprio ID. Un'esecuzione corrisponde a una serie di modifiche, ad esempio un commit unito o una versione manuale dell'ultimo commit. Due esecuzioni possono rilasciare lo stesso insieme di modifiche in momenti diversi.

Mentre una pipeline può elaborare più esecuzioni contemporaneamente, una fase di pipeline elabora solo un'esecuzione alla volta. A tale scopo, uno stage viene bloccato mentre elabora un'esecuzione. Due esecuzioni di pipeline non possono occupare la stessa fase nello stesso momento. L'esecuzione in attesa di entrare nella fase occupata viene riferita a un'esecuzione in entrata. Un'esecuzione in entrata può comunque fallire, essere sostituita o essere interrotta manualmente. Per ulteriori informazioni sul funzionamento delle esecuzioni in entrata, consulta. Come funzionano le esecuzioni in entrata

Le esecuzioni della pipeline attraversano le fasi della pipeline in ordine. Gli stati validi per le pipeline sono `InProgress`, `Stopping`, `Stopped`, `Succeeded`, `Superseded` e `Failed`.

Per ulteriori informazioni, vedere. PipelineExecution

Esecuzioni interrotte

L'esecuzione della pipeline può essere interrotta manualmente in modo che l'esecuzione della pipeline in corso non continui attraverso la pipeline. Se viene interrotta manualmente, l'esecuzione di una pipeline mostra uno stato `Stopping` fino a quando non viene completamente interrotta. Quindi mostra uno stato `Stopped`. È possibile ripetere l'esecuzione di una pipeline `Stopped`.

Esistono due modi per interrompere l'esecuzione di una pipeline:

- Fermati e aspetta
- Fermati e abbandona

Per informazioni sui casi d'uso per arrestare un'esecuzione e i dettagli della sequenza per queste opzioni, vedere Come vengono interrotte le esecuzioni di pipeline.

Esecuzioni non riuscite

Se un'esecuzione fallisce, si arresta e non attraversa completamente la pipeline. Il suo stato è `FAILED` e la fase è sbloccata. Un'esecuzione più recente può recuperare ed entrare nella fase sbloccata e bloccarla. È possibile riprovare un'esecuzione non riuscita a meno che l'esecuzione non riuscita non sia stata sostituita o non sia ripristinabile. È possibile ripristinare una fase fallita riportandola a una precedente esecuzione riuscita.

Modalità di esecuzione

Per fornire l'ultimo set di modifiche tramite una pipeline, le esecuzioni più recenti passano e sostituiscono le esecuzioni meno recenti già in esecuzione attraverso la pipeline. In questo caso, l'esecuzione precedente viene sostituita dall'esecuzione più recente. Un'esecuzione può essere sostituita da un'esecuzione più recente in un certo punto, che è il punto tra le fasi. SUPERSEDED è la modalità di esecuzione predefinita.

In modalità SUPERSEDED, se un'esecuzione è in attesa di entrare in una fase bloccata, un'esecuzione più recente potrebbe recuperarla e sostituirla. L'esecuzione più recente ora attende lo sblocco dello stage e l'esecuzione sostituita si arresta con uno stato `SUPERSEDED`. Quando l'esecuzione di una pipeline viene sostituita, l'esecuzione viene interrotta e non attraversa completamente la pipeline. Non è più possibile riprovare l'esecuzione sostituita dopo che è stata sostituita in questa fase. Le altre modalità di esecuzione disponibili sono la modalità PARALLEL o QUEUED.

Per ulteriori informazioni sulle modalità di esecuzione e sulle fasi bloccate, vedere. Come vengono elaborate le esecuzioni in modalità SOSTITUITA

Operazioni sullo stage

Quando l'esecuzione di una pipeline attraversa una fase, questa è in fase di completamento di tutte le azioni al suo interno. Per informazioni sul funzionamento delle operazioni sullo stage e sulle fasi bloccate, vedereCome vengono elaborate le esecuzioni in modalità SOSTITUITA.

Gli stati validi per gli stadi sono `InProgressStopping`,`Stopped`,`Succeeded`, `eFailed`. È possibile riprovare una fase fallita a meno che la fase fallita non sia riprovabile. Per ulteriori informazioni, vedere. StageExecution È possibile ripristinare una fase fino a un'esecuzione precedente specificata con successo. Una fase può essere configurata per il rollback automatico in caso di errore, come descritto inConfigurazione del rollback dello stage. Per ulteriori informazioni, vedere RollbackStage.

Esecuzioni di operazioni

Un'esecuzione di un'operazione è il processo di completamento di un'operazione configurata che opera su artefatti designati. Questi possono essere artefatti di input, artefatti di output, o entrambi. Ad esempio, un'azione di compilazione potrebbe eseguire comandi di compilazione su un artefatto di input, ad esempio la compilazione del codice sorgente dell'applicazione. I dettagli relativi all'esecuzione dell'azione includono un ID di esecuzione dell'azione, il trigger dell'origine dell'esecuzione della pipeline correlata e gli artefatti di input e output per l'azione.

Gli stati validi per le azioni sono`InProgress`, `AbandonedSucceeded`, o`Failed`. Per ulteriori informazioni, vedere ActionExecution.

Tipi di esecuzione

L'esecuzione di una pipeline o di uno stage può essere un'esecuzione standard o ripristinata.

Per i tipi standard, l'esecuzione ha un ID univoco ed è un'esecuzione completa della pipeline. Un rollback della pipeline ha una fase da ripristinare e un'esecuzione riuscita per la fase come esecuzione di destinazione a cui eseguire il rollback. L'esecuzione della pipeline di destinazione viene utilizzata per recuperare le revisioni e le variabili di origine per la riesecuzione dello stage.

Tipi di operazione

I tipi di azione sono azioni preconfigurate disponibili per la selezione in. CodePipeline Il tipo di azione è definito dal proprietario, dal provider, dalla versione e dalla categoria. Il tipo di azione fornisce parametri personalizzati che vengono utilizzati per completare le attività di azione in una pipeline.

Per informazioni sui prodotti Servizi AWS e servizi di terze parti che puoi integrare nella tua pipeline in base al tipo di azione, consulta. Integrazioni con tipi di CodePipeline azioni

Per informazioni sui modelli di integrazione supportati per i tipi di azione in CodePipeline, consultaRiferimento al modello di integrazione.

Per informazioni su come i provider di terze parti possono configurare e gestire i tipi di azione in CodePipeline, vedereLavorare con i tipi di azione.

Artifacts

Gli artefatti si riferiscono alla raccolta di dati, come il codice sorgente dell'applicazione, le applicazioni create, le dipendenze, i file di definizioni, i modelli e così via, che vengono elaborati dalle azioni della pipeline. Gli artefatti sono prodotti da alcune azioni e consumati da altre. In una pipeline, gli artefatti possono essere l'insieme di file lavorati da un'azione (artefatti di input) o l'output aggiornato di un'azione completata (artefatti di output).

Le azioni passano l'output a un'altra azione per un'ulteriore elaborazione utilizzando il bucket di artefatti della pipeline. CodePipeline copia gli artefatti nel negozio degli artefatti, dove l'azione li raccoglie. Per ulteriori informazioni sugli artefatti, vedi Artefatti di input e output.

Revisioni delle origini

Quando si effettua una modifica del codice sorgente, viene creata una nuova versione. Una revisione dell'origine è la versione di una modifica di origine che attiva l'esecuzione di una pipeline. Un'esecuzione elabora le revisioni dei sorgenti. Per i CodeCommit repository GitHub e gli archivi, questo è il commit. Per bucket o azioni S3, questa è la versione dell'oggetto.

È possibile avviare l'esecuzione di una pipeline con una revisione del codice sorgente, ad esempio un commit, specificata dall'utente. L'esecuzione elaborerà la revisione specificata e sostituirà quella che sarebbe stata la revisione utilizzata per l'esecuzione. Per ulteriori informazioni, consulta Avvia una pipeline con una modifica della revisione del codice sorgente.

Trigger

I trigger sono eventi che avviano la pipeline. Alcuni trigger, come l'avvio manuale di una pipeline, sono disponibili per tutti i provider di azioni di origine presenti in una pipeline. Alcuni trigger dipendono dal fornitore di origine di una pipeline. Ad esempio, CloudWatch gli eventi devono essere configurati con risorse di eventi di Amazon CloudWatch a cui è stato aggiunto l'ARN della pipeline come destinazione nella regola dell'evento. Amazon CloudWatch Events è il trigger consigliato per il rilevamento automatico delle modifiche per le pipeline con un'azione sorgente CodeCommit o S3.
I webhook sono un tipo di trigger configurato per eventi di repository di terze parti. Ad esempio, WebHookV2 è un tipo di trigger che consente di utilizzare i tag Git per avviare pipeline con provider di sorgenti di terze parti come GitHub .com, GitHub Enterprise Server, GitLab .com, GitLab self-

managed o Bitbucket Cloud. Nella configurazione della pipeline, puoi specificare un filtro per i trigger, come la richiesta push o pull. Puoi filtrare gli eventi push del codice su tag Git, branch o percorsi di file. È possibile filtrare gli eventi di pull request in base a eventi (aperti, aggiornati, chiusi), rami o percorsi di file.

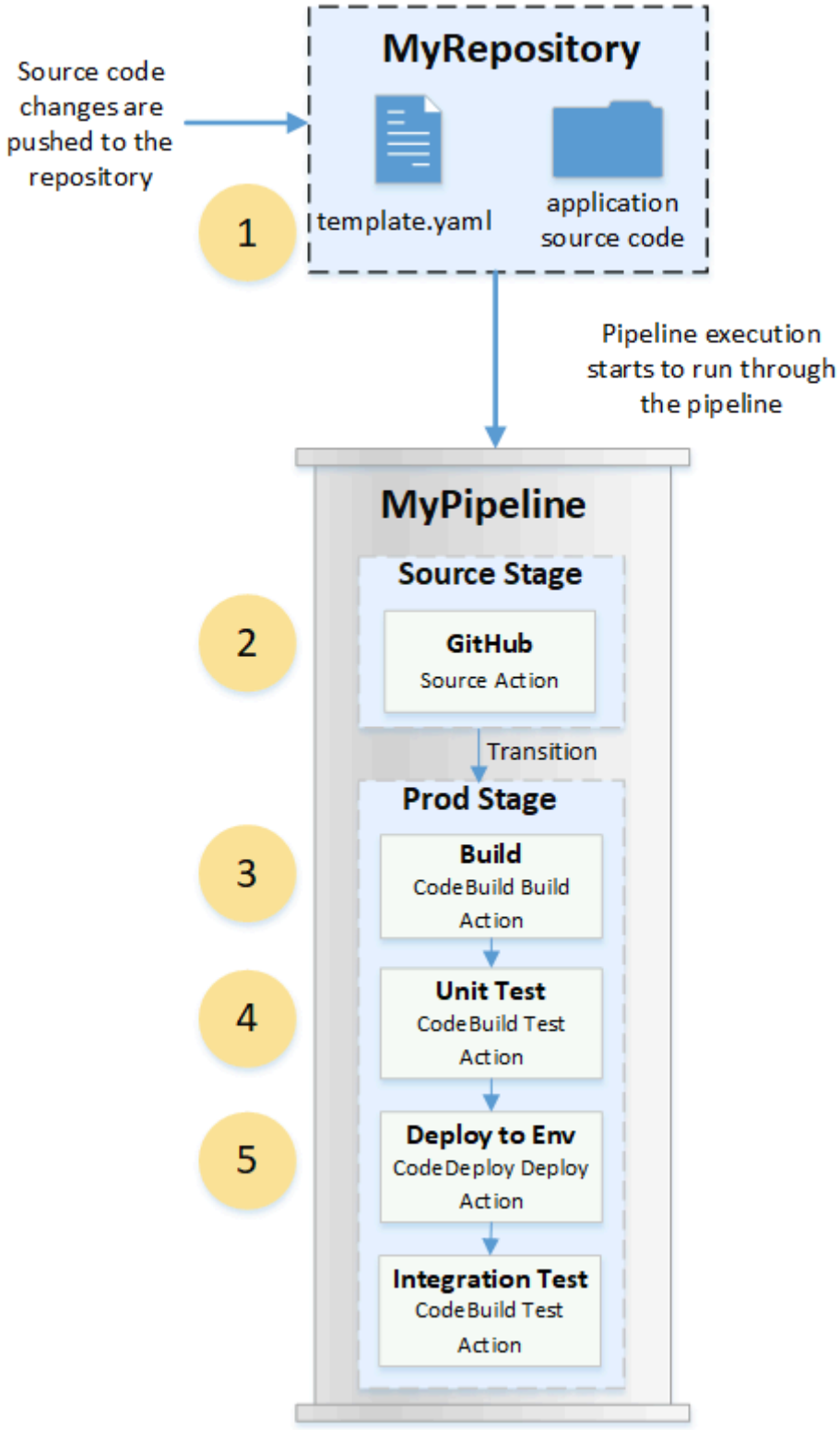
Per ulteriori informazioni sui trigger, consulta Avvia una pipeline in CodePipeline. Per un tutorial che ti spiega come usare i tag Git come trigger per la tua pipeline, vedi. Tutorial: usa i tag Git per avviare la tua pipeline

Variables

Una variabile è un valore che può essere utilizzato per configurare dinamicamente le azioni nella pipeline. Le variabili possono essere dichiarate a livello di pipeline o emesse da azioni nella pipeline. I valori delle variabili vengono risolti al momento dell'esecuzione della pipeline e possono essere visualizzati nella cronologia delle esecuzioni. Per le variabili dichiarate a livello di pipeline, potete definire valori predefiniti nella configurazione della pipeline o sovrascriverli per una determinata esecuzione. Per le variabili emesse da un'azione, il valore è disponibile dopo che un'azione è stata completata con successo. Per ulteriori informazioni, consulta Variables.

DevOps esempio di pipeline

Come esempio di DevOps pipeline, una pipeline a due stadi potrebbe avere uno stadio di origine chiamato Source e un secondo stadio chiamato Prod. In questo esempio, la pipeline aggiorna l'applicazione con le ultime modifiche e distribuisce continuamente il risultato più recente. Prima di distribuire l'applicazione più recente, la pipeline crea e verifica l'applicazione Web. In questo esempio, un gruppo di sviluppatori ha impostato un modello di infrastruttura e il codice sorgente per un'applicazione Web in un repository chiamato. GitHub MyRepository



Ad esempio, uno sviluppatore spinge una correzione alla pagina indice dell'applicazione Web e si verifica quanto segue:

1. Il codice sorgente dell'applicazione viene mantenuto in un repository configurato come azione GitHub sorgente nella pipeline. Quando gli sviluppatori inviano i commit al repository, CodePipeline rilevano la modifica inviata e l'esecuzione della pipeline inizia dal Source Stage.
2. L'azione GitHub source viene completata correttamente (ovvero, le ultime modifiche sono state scaricate e archiviate nel bucket di artefatti unico per quell'esecuzione). Gli artefatti di output prodotti dall'azione GitHub source, che sono i file dell'applicazione presenti nel repository, vengono quindi utilizzati come artefatti di input su cui lavorare le azioni nella fase successiva.
3. L'esecuzione della pipeline passa da fase Source a fase Prod. La prima azione in Prod Stage esegue un progetto di compilazione creato CodeBuild e configurato come azione di compilazione nella pipeline. L'attività di compilazione estrae un'immagine dell'ambiente di compilazione e crea l'applicazione Web in un contenitore virtuale.
4. L'azione successiva in Prod Stage è un progetto di unit test creato CodeBuild e configurato come azione di test nella pipeline.
5. Il codice unità testato viene successivamente elaborato da un'azione di distribuzione nella fase Prod che distribuisce l'applicazione in un ambiente di produzione. Una volta completata con successo l'azione di distribuzione, l'azione finale nella fase è un progetto di test di integrazione creato CodeBuild e configurato come azione di test nella pipeline. L'azione di test chiama gli script della shell che installano ed eseguono uno strumento di test, ad esempio un controllo dei collegamenti, nell'applicazione Web. Dopo il completamento con successo, l'output è un'applicazione web integrata e una serie di risultati dei test.

Gli sviluppatori possono aggiungere azioni alla pipeline che distribuiscono o testano ulteriormente l'applicazione dopo che è stata compilata e testata per ogni modifica.

Per ulteriori informazioni, consulta Funzionamento delle esecuzioni pipeline.

Funzionamento delle esecuzioni pipeline

Questa sezione fornisce una panoramica del modo in cui CodePipeline elabora una serie di modifiche. CodePipelinetiene traccia di ogni esecuzione della pipeline che inizia quando una pipeline viene avviata manualmente o viene apportata una modifica al codice sorgente. CodePipeline utilizza le seguenti modalità di esecuzione per gestire il modo in cui ogni esecuzione procede nella pipeline.

- Modalità SOSTITUITA: un'esecuzione più recente può sostituire un'esecuzione precedente. Questa è l'impostazione predefinita.

- Modalità CODA: le esecuzioni vengono elaborate una per una nell'ordine in cui sono messe in coda. Ciò richiede una pipeline di tipo V2.
- Modalità PARALLEL: in modalità PARALLEL, le esecuzioni vengono eseguite simultaneamente e indipendentemente l'una dall'altra. Le esecuzioni non attendono il completamento di altre esecuzioni prima di iniziare o terminare. Ciò richiede una pipeline di tipo V2.

Come vengono avviate le esecuzioni pipeline

È possibile avviare un'esecuzione quando si modifica il codice sorgente o si avvia manualmente la pipeline. Puoi anche attivare un'esecuzione tramite una regola Amazon CloudWatch Events che pianifichi. Ad esempio, quando una modifica del codice sorgente viene inviata in un repository configurato come azione di origine della pipeline, la pipeline rileva la modifica e avvia un'esecuzione.

> **Note**
>
> Se una pipeline contiene più operazioni di origine, vengono tutte eseguite nuovamente, anche se viene rilevata una modifica per una sola operazione di origine.

Come vengono elaborate le revisioni dei sorgenti nelle esecuzioni della pipeline

Per ogni esecuzione della pipeline che inizia con modifiche al codice sorgente (revisioni del codice sorgente), le revisioni dei sorgenti vengono determinate come segue.

- Per le pipeline con una CodeCommit fonte, l'HEAD viene clonato nel momento in cui viene CodePipeline inviato il commit. Ad esempio, viene inviato un commit, che avvia la pipeline per l'esecuzione 1. Nel momento in cui viene inviato un secondo commit, viene avviata la pipeline per l'esecuzione 2.

  > **Note**
  >
  > Per le pipeline in modalità PARALLEL con un' CodeCommit origine, indipendentemente dal commit che ha attivato l'esecuzione della pipeline, l'azione di origine clonerà sempre l'HEAD nel momento in cui viene avviato. Per ulteriori informazioni, consulta CodeCommit

oppure le revisioni dei sorgenti S3 in modalità PARALLEL potrebbero non corrispondere all'evento EventBridge .

- Per le pipeline con un'origine S3, viene utilizzato l' EventBridge evento per l'aggiornamento del bucket S3. Ad esempio, l'evento viene generato quando un file viene aggiornato nel bucket di origine, che avvia la pipeline per l'esecuzione 1. Nel momento in cui viene effettuato l'evento per un secondo aggiornamento del bucket, viene avviata la pipeline per l'esecuzione 2.

  > **Note**
  >
  > Per le pipeline in modalità PARALLEL con una sorgente S3, indipendentemente dal tag di immagine che ha attivato l'esecuzione, l'azione source inizierà sempre con il tag di immagine più recente. Per ulteriori informazioni, consulta CodeCommit oppure le revisioni dei sorgenti S3 in modalità PARALLEL potrebbero non corrispondere all'evento EventBridge .

- Per le pipeline con un'origine di connessione, ad esempio verso Bitbucket, l'HEAD viene clonato da nel CodePipeline momento in cui viene inviato il commit. Ad esempio, per una pipeline in modalità PARALLEL, viene inviato un commit, che avvia la pipeline per l'esecuzione 1, mentre la seconda esecuzione della pipeline utilizza il secondo commit.

Come vengono interrotte le esecuzioni di pipeline

Per utilizzare la console per interrompere l'esecuzione di una pipeline, è possibile scegliere Interrompi esecuzione nella pagina di visualizzazione della pipeline, nella pagina della cronologia delle esecuzioni o nella pagina della cronologia dettagliata. Per utilizzare l'interfaccia della riga di comando per interrompere l'esecuzione di una pipeline, utilizzare il comando `stop-pipeline-execution`. Per ulteriori informazioni, consulta Interrompere l'esecuzione di una pipeline in CodePipeline.

Esistono due modi per interrompere l'esecuzione di una pipeline:

- Interrompi e attendi: tutte le esecuzioni di azioni in corso possono essere completate e le azioni successive non vengono avviate. L'esecuzione della pipeline non prosegue con le fasi successive. Non è possibile utilizzare questa opzione in un'esecuzione già in uno stato `Stopping`.
- Arresto e abbandono: tutte le esecuzioni delle azioni in corso vengono abbandonate e non completate e le azioni successive non vengono avviate. L'esecuzione della pipeline non prosegue

con le fasi successive. È possibile utilizzare questa opzione su un'esecuzione che è già in uno stato `Stopping`.

> **Note**
>
> Questa opzione può portare ad attività non riuscite o fuori sequenza.

Ogni opzione si traduce in una sequenza diversa di fasi di esecuzione della pipeline e delle azioni, come segue.

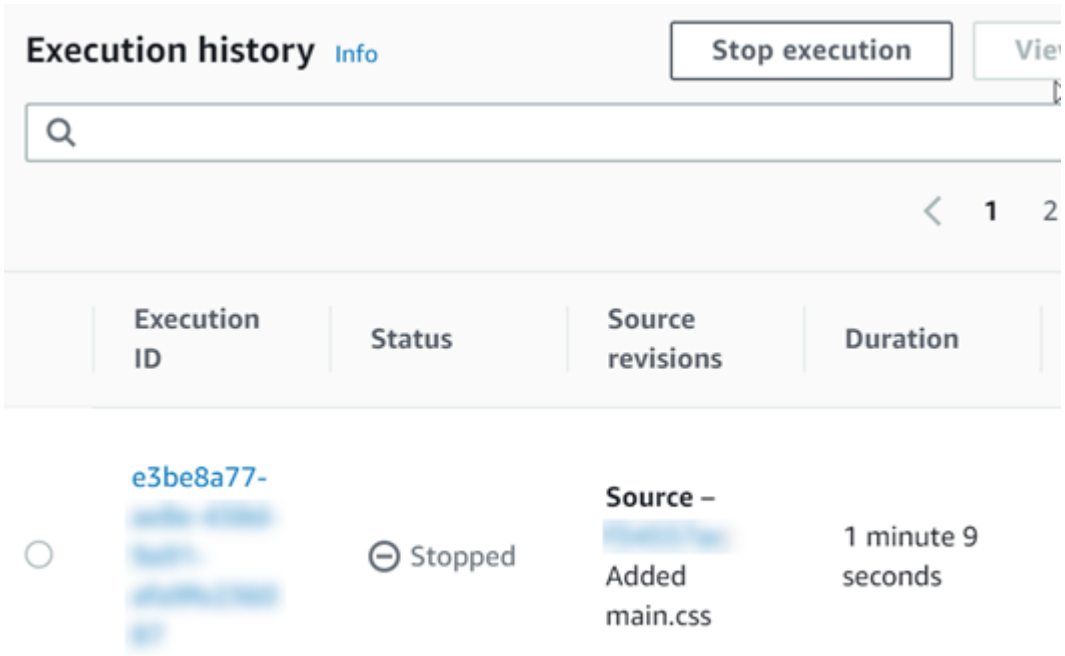
Opzione 1: Ferma e attendi

Quando si sceglie di interrompere e attendere, l'esecuzione selezionata continua fino al completamento delle azioni in corso. Ad esempio, l'esecuzione della pipeline seguente è stata interrotta mentre l'azione di compilazione era in corso.

1. Nella visualizzazione pipeline, viene visualizzato il banner del messaggio di successo e l'azione di compilazione continua fino al completamento. Lo stato di esecuzione della pipeline è In arresto.

   Nella visualizzazione cronologia, lo stato delle azioni in corso, ad esempio l'azione di compilazione, è In corso fino al completamento dell'azione di compilazione. Mentre le azioni sono in corso, lo stato di esecuzione della pipeline è In arresto.
2. L'esecuzione si interrompe al termine del processo di arresto. Se l'azione di compilazione viene completata correttamente, il relativo stato è Eseguito correttamente e l'esecuzione della pipeline mostra lo stato Arrestato. Le azioni successive non iniziano. Il pulsante Riprova è abilitato.

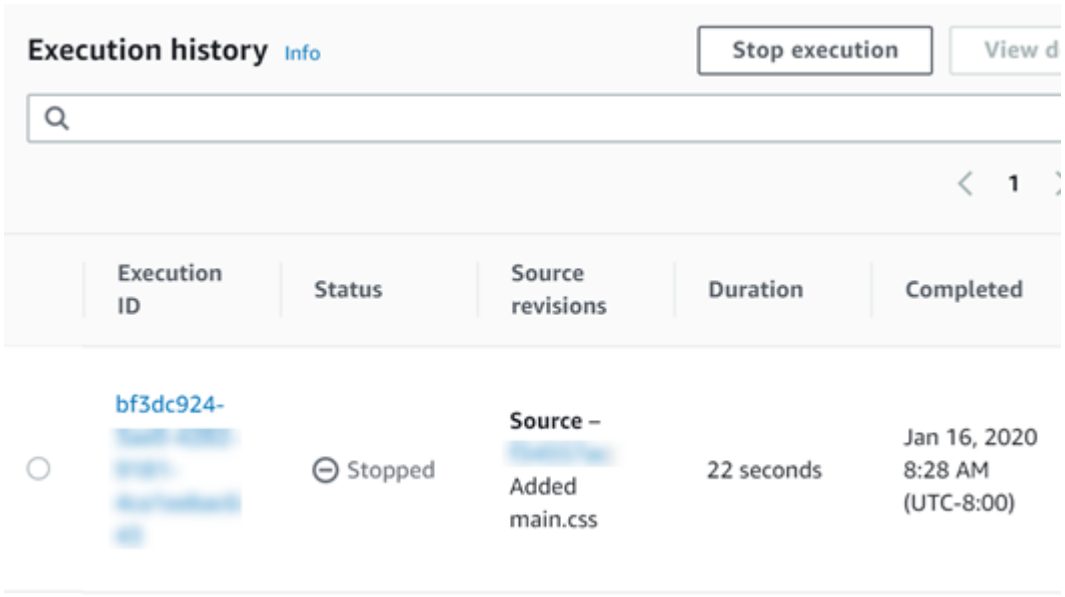
   Nella visualizzazione cronologia, lo stato di esecuzione viene Arrestato dopo il completamento dell'azione in corso.



Opzione 2: fermarsi e abbandonare

Quando si sceglie di interrompere e abbandonare, l'esecuzione selezionata non attende il completamento delle azioni in corso. Le azioni vengono abbandonate. Ad esempio, l'esecuzione della pipeline seguente è stata interrotta e abbandonata mentre l'azione di compilazione era in corso.

1. Nella visualizzazione della pipeline viene visualizzato il messaggio del banner di successo, l'azione di compilazione mostra lo stato In corso e l'esecuzione della pipeline mostra uno stato In arresto.
2. Dopo l'interruzione dell'esecuzione della pipeline, l'azione di compilazione mostra lo stato Abbandonato e l'esecuzione della pipeline mostra lo stato Arrestato. Le azioni successive non iniziano. Il pulsante Riprova è abilitato.
3. Nella visualizzazione cronologia, lo stato di esecuzione è Arrestato.



Casi d'uso per arrestare l'esecuzione di una pipeline

Si consiglia di utilizzare l'opzione di arresto e attesa per interrompere l'esecuzione di una pipeline. Questa opzione è più sicura perché evita possibili errori o out-of-sequence attività nella pipeline. Quando un'azione viene abbandonata in CodePipeline, il fornitore dell'azione continua tutte le attività correlate all'azione. Nel caso di un' AWS CloudFormation azione, l'azione di distribuzione nella pipeline viene abbandonata, ma l'aggiornamento dello stack potrebbe continuare e causare un aggiornamento non riuscito.

Ad esempio di azioni abbandonate che possono comportare out-of-sequence attività, se stai distribuendo un file di grandi dimensioni (1 GB) tramite un'azione di distribuzione S3 e scegli di interrompere e abbandonare l'azione mentre la distribuzione è già in corso, l'azione viene abbandonata in Amazon S3 CodePipeline, ma continua in Amazon S3. Amazon S3 non riceve alcuna istruzione per annullare il caricamento. Successivamente, se si avvia una nuova esecuzione della pipeline con un file molto piccolo, ora sono in corso due distribuzioni. Poiché le dimensioni del file della nuova esecuzione sono ridotte, la nuova distribuzione viene completata mentre la precedente distribuzione è ancora in fase di caricamento. Al termine della precedente distribuzione, il nuovo file viene sovrascritto dal vecchio file.

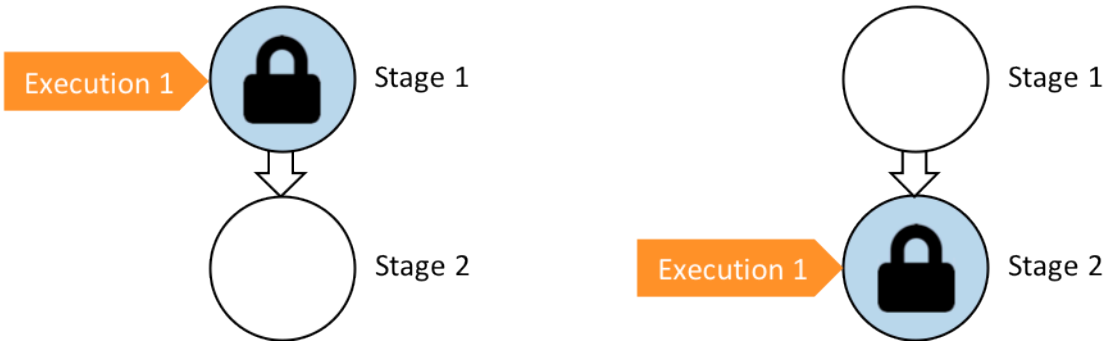
Potresti voler utilizzare l'opzione stop and abbandon nel caso in cui tu abbia un'azione personalizzata. Ad esempio, puoi abbandonare un'azione personalizzata con un lavoro che non deve essere completato prima di iniziare una nuova esecuzione per la correzione di un bug.

Come vengono elaborate le esecuzioni in modalità SOSTITUITA

La modalità predefinita per l'elaborazione delle esecuzioni è la modalità SOSTITUITA. Un'esecuzione consiste in un insieme di modifiche raccolte ed elaborate dall'esecuzione. Le pipeline possono elaborare più esecuzioni contemporaneamente. Ogni esecuzione viene eseguita separatamente attraverso la pipeline. La pipeline elabora ogni esecuzione in ordine e potrebbe sostituire un'esecuzione precedente con una successiva. Le seguenti regole vengono utilizzate per elaborare le esecuzioni in una pipeline per la modalità SOSTITUITA.

Regola 1: Le fasi vengono bloccate quando viene elaborata un'esecuzione

Poiché ogni fase può elaborare solo un'esecuzione alla volta, la fase viene bloccata mentre è in corso. Quando l'esecuzione completa una fase, passa alla fase successiva della pipeline.

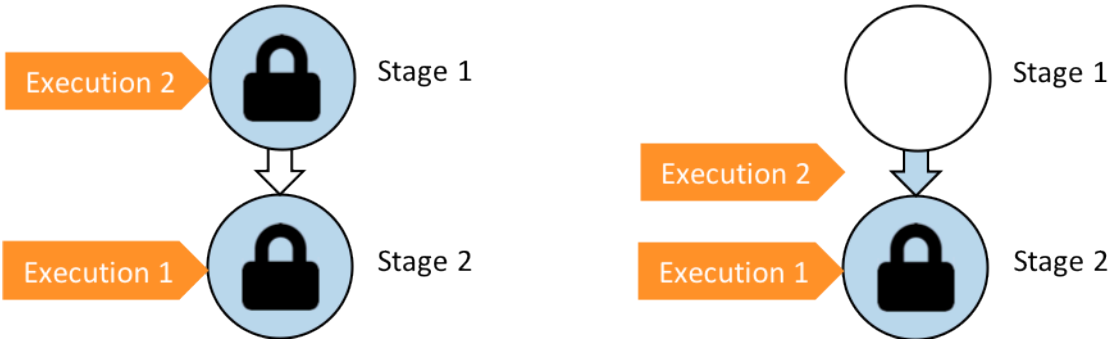


Prima: Stage 1 is locked as Execution 1 enters. Dopo: Stage 2 is locked as Execution 1 enters.

Regola 2: le esecuzioni successive attendono lo sblocco dello fase

Mentre una fase è bloccata, le esecuzioni in attesa sono tenute davanti alla fase bloccata. Tutte le operazioni configurate per una fase devono essere completate prima che fase sia considerata completata. Un errore comporta l'applicazione del lucchetto sulla fase. Quando un'esecuzione viene interrotta, l'esecuzione non continua in uno stage e lo stage viene sbloccato.

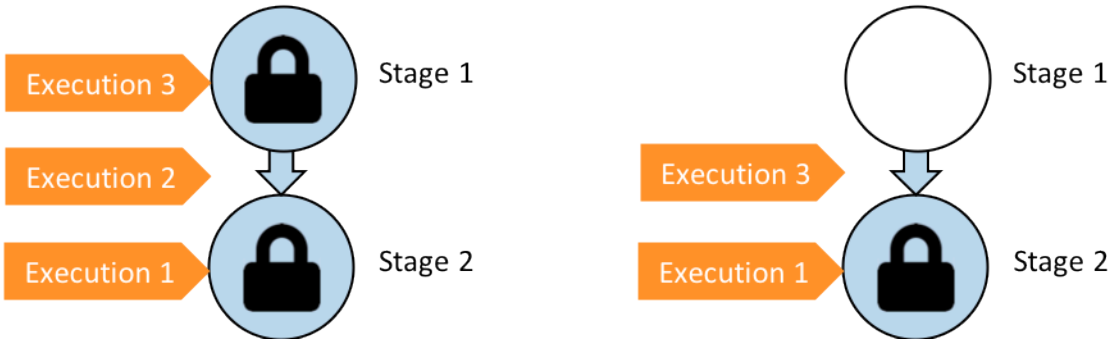
> **Note**
>
> Prima di interrompere un'esecuzione, è consigliabile disabilitare la transizione davanti allo stage. In questo modo, quando lo stage viene sbloccato a causa dell'esecuzione interrotta, lo stage non accetta un'esecuzione successiva della pipeline.



Prima: Stage 2 is locked as Execution 1 enters. Dopo: Execution 2 exits Stage 1 and waits between stages.

Regola 3: le esecuzioni in attesa vengono sostituite da esecuzioni più recenti

Le esecuzioni vengono sostituite solo tra una fase e l'altra. Una fase bloccata contiene un'esecuzione all'inizio della fase in attesa del completamento della fase. Un'esecuzione più recente supera un'esecuzione in attesa e continua alla fase successiva non appena la fase viene sbloccata. L'esecuzione sostituita non continua. In questo esempio, l'esecuzione 2 è stata sostituita dall'esecuzione 3 in attesa della fase bloccata. L'esecuzione 3 entra nella fase successiva.



Prima: l'esecuzione 2 attende tra le fasi mentre l'esecuzione 3 entra nella fase 1. Dopo: l'esecuzione 3 esce dalla fase 1. L'esecuzione 2 è sostituita dall'esecuzione 3.

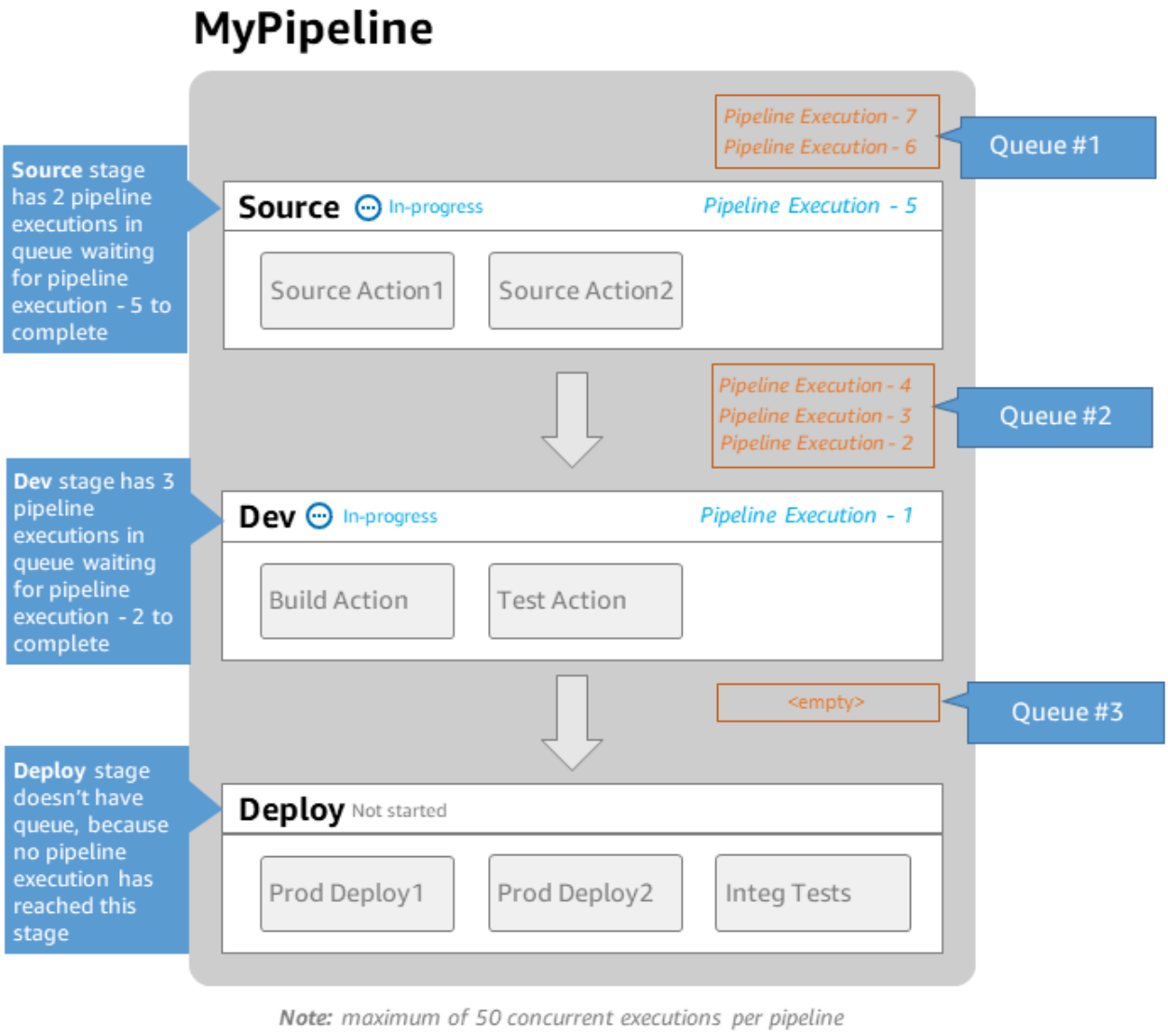
Per ulteriori informazioni sulle considerazioni relative alla visualizzazione e al passaggio da una modalità di esecuzione all'altra, vedereImpostare o modificare la modalità di esecuzione della pipeline. Per ulteriori informazioni sulle quote con modalità di esecuzione, vedere. Quote in AWS CodePipeline

Come vengono elaborate le esecuzioni in modalità QUEUED

Per le pipeline in modalità QUEUED, le fasi sono bloccate durante l'elaborazione di un'esecuzione; tuttavia, le esecuzioni in attesa non superano le esecuzioni già iniziate.

Le esecuzioni in attesa si riuniscono nei punti di ingresso delle fasi bloccate nell'ordine in cui raggiungono la fase, formando una coda di esecuzioni in attesa. Con la modalità QUEUED, puoi avere più code nella stessa pipeline. Quando un'esecuzione in coda entra in una fase, questa viene bloccata e non possono entrare altre esecuzioni. Questo comportamento rimane lo stesso della modalità SUPERSEDED. Al termine dell'esecuzione, la fase viene sbloccata e pronta per l'esecuzione successiva.

Il diagramma seguente mostra come le fasi di una pipeline in modalità QUEUED elaborano le esecuzioni. Ad esempio, mentre la fase Source elabora l'esecuzione 5, le esecuzioni per 6 e 7 formano Queue #1 e attendono al punto di ingresso dello stage. L'esecuzione successiva nella coda verrà elaborata dopo lo sblocco della fase.



Per ulteriori informazioni sulle considerazioni relative alla visualizzazione e al passaggio da una modalità di esecuzione all'altra, vedere. Impostare o modificare la modalità di esecuzione della pipeline Per ulteriori informazioni sulle quote con modalità di esecuzione, vedere. Quote in AWS CodePipeline

Come vengono elaborate le esecuzioni in modalità PARALLEL

Per le pipeline in modalità PARALLEL, le esecuzioni sono indipendenti l'una dall'altra e non attendono il completamento delle altre esecuzioni prima di iniziare. Non ci sono code. Per visualizzare le esecuzioni parallele nella pipeline, usa la visualizzazione della cronologia delle esecuzioni.

Utilizza la modalità PARALLEL in ambienti di sviluppo in cui ogni funzionalità ha il proprio ramo di funzionalità e viene distribuita su destinazioni non condivise da altri utenti.

Per ulteriori informazioni sulle considerazioni relative alla visualizzazione e al passaggio da una modalità di esecuzione all'altra, vedere. Impostare o modificare la modalità di esecuzione della pipeline Per ulteriori informazioni sulle quote con modalità di esecuzione, vedere. Quote in AWS CodePipeline

Gestione del flusso della pipeline

Il flusso delle esecuzioni di pipeline può essere controllato da:

- Una transizione, che controlla il flusso delle esecuzioni nella fase. Le transizioni possono essere abilitate o disabilitate. Quando una transizione è disabilitata, le esecuzioni della pipeline non possono entrare nella fase. L'esecuzione della pipeline in attesa di entrare in una fase in cui la transizione è disabilitata viene chiamata esecuzione in entrata. Dopo aver abilitato la transizione, un'esecuzione in entrata passa allo stage e la blocca.

  Analogamente alle esecuzioni in attesa di una fase bloccata, quando una transizione è disabilitata, l'esecuzione in attesa di entrare nella fase può comunque essere sostituita da una nuova esecuzione. Quando una transizione disabilitata viene riabilitata, l'esecuzione più recente, inclusa quella che ha sostituito le esecuzioni meno recenti mentre la transizione è stata disabilitata, entra nella fase.
- Un'azione di approvazione, che impedisce a una pipeline di passare all'azione successiva fino a quando non viene concessa l'autorizzazione (ad esempio, tramite l'approvazione manuale da parte di un'identità autorizzata). È possibile utilizzare un'azione di approvazione quando si desidera controllare il momento in cui una pipeline passa a una fase finale di produzione ad esempio.

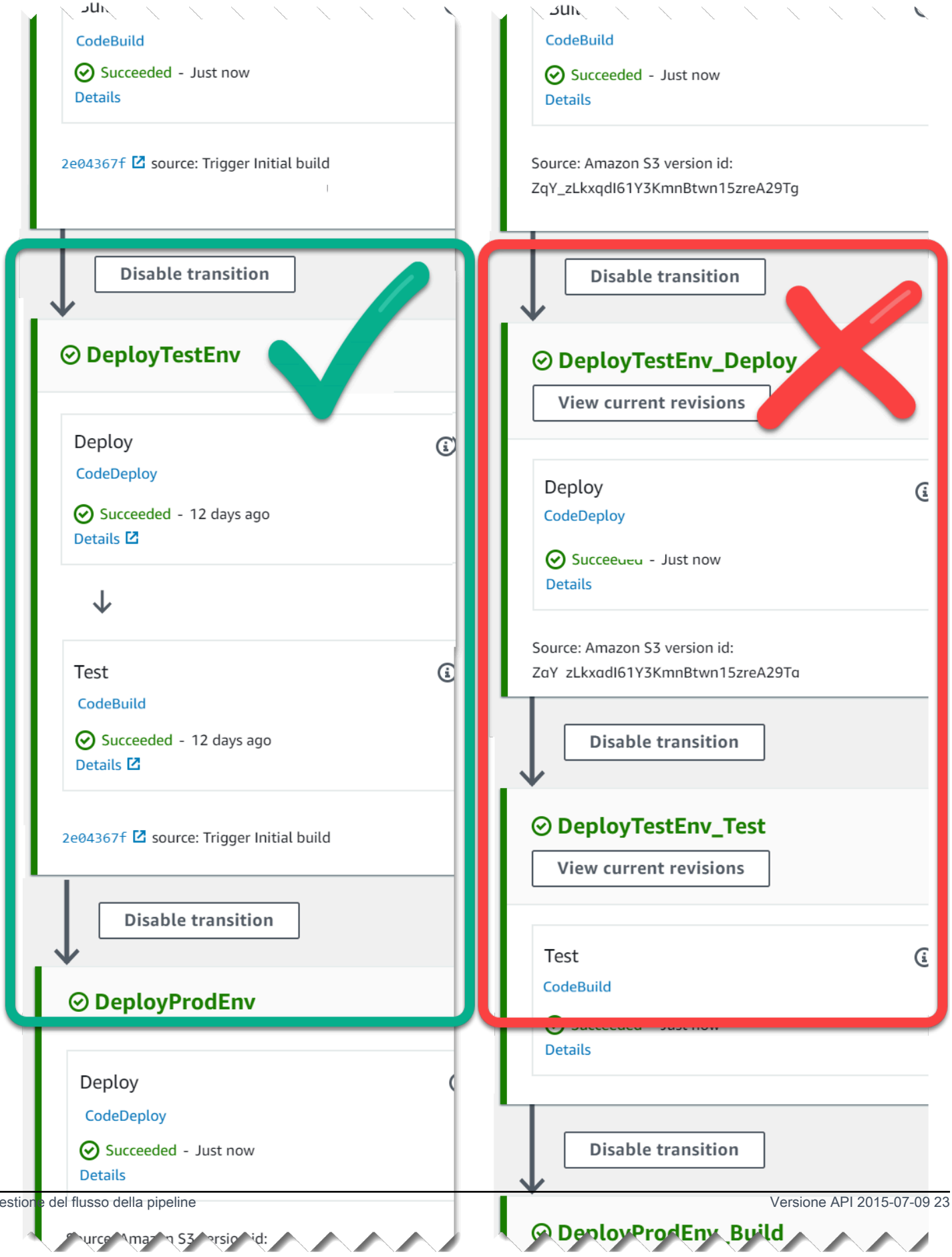
> **Note**
>
> Una fase con un'azione di approvazione viene bloccata fino a quando l'azione di approvazione non viene approvata o rifiutata o è scaduta. Un'azione di approvazione scaduta viene elaborata allo stesso modo di un'azione non riuscita.

- Un errore, quando un'azione in una fase non viene completata correttamente. La revisione non esegue la transizione all'operazione successiva nella fase o alla fase successiva nella pipeline. Si può verificare quanto segue:
  - Viene eseguito un tentativo di ripetizione della fase che contiene le operazioni non riuscite. Questo riprende l'esecuzione (riprova le azioni fallite e, se riescono, continua nella fase/pipeline).
  - Un'altra esecuzione entra nella fase fallita e sostituisce l'esecuzione non riuscita. A questo punto, l'esecuzione non riuscita non può essere ripetuta.

Struttura consigliata per pipeline

Quando si decide come una modifica del codice deve fluire attraverso la pipeline, è meglio raggruppare le azioni correlate all'interno di una fase in modo che, quando la fase si blocca, tutte le azioni elaborino la stessa esecuzione. È possibile creare una fase per ogni ambiente applicativo o zona di disponibilità e così via. Regione AWS Una pipeline con troppe fasi (cioè troppo granulare) può consentire troppe modifiche simultanee, mentre una pipeline con molte azioni in una fase di grandi dimensioni (troppo grossolana) può richiedere troppo tempo per rilasciare una modifica.

Ad esempio, un'azione di test dopo un'azione di distribuzione nella stessa fase è garantita per testare la stessa modifica che è stata distribuita. In questo esempio, una modifica viene compilata, testata e distribuita in un ambiente test, quindi la modifica più recente dell'ambiente di test viene distribuita in un ambiente di produzione. Nell'esempio consigliato, l'ambiente Test e l'ambiente Prod sono fasi separate.


CodeBuild
Succeeded - Just now
Details
2e04367f source: Trigger Initial build
Disable transition
DeployTestEnv
Deploy
CodeDeploy
Succeeded - 12 days ago
Details
Test
CodeBuild
Succeeded - 12 days ago
Details
2e04367f source: Trigger Initial build
Disable transition
DeployProdEnv
Deploy
CodeDeploy
Succeeded - Just now
Details
CodeBuild
Succeeded - Just now
Details
Source: Amazon S3 version id:
ZqY_zLkxqdI61Y3KmnBtwn15zreA29Tg
Disable transition
DeployTestEnv_Deploy
View current revisions
Deploy
CodeDeploy
Succeeded - Just now
Details
Source: Amazon S3 version id:
Disable transition
DeployTestEnv_Test
View current revisions
Test
CodeBuild
Details
Disable transition
DeployProdEnv_Build

Sinistra: operazioni correlate di test, distribuzione e approvazione raggruppate insieme (scelta consigliata). Destra: operazioni correlate in fasi separate (scelta non consigliata).

Come funzionano le esecuzioni in entrata

Un'esecuzione in entrata è un'esecuzione in attesa che diventi disponibile una fase, una transizione o un'azione non disponibili prima di procedere. La fase, la transizione o l'azione successiva potrebbero non essere disponibili perché:

- Un'altra esecuzione è già entrata nella fase successiva e l'ha bloccata.
- La transizione per accedere alla fase successiva è disabilitata.

È possibile disabilitare una transizione per mantenere un'esecuzione in entrata se si desidera controllare se un'esecuzione corrente ha il tempo di essere completata nelle fasi successive o se si desidera interrompere tutte le azioni in un determinato momento. Per determinare se è in corso un'esecuzione in entrata, è possibile visualizzare la pipeline nella console o visualizzare l'output del comando. get-pipeline-state

Le esecuzioni in entrata funzionano in base alle seguenti considerazioni:

- Non appena l'azione, la transizione o la fase bloccata diventano disponibili, l'esecuzione in entrata in corso entra nella fase e prosegue attraverso la pipeline.
- Mentre l'esecuzione in entrata è in attesa, può essere interrotta manualmente. Un'esecuzione in entrata può avere uno stato `InProgressStopped`, o`Failed`.
- Quando un'esecuzione in entrata è stata interrotta o non è riuscita, non può essere ritentata perché non ci sono azioni fallite da riprovare. Quando un'esecuzione in entrata è stata interrotta e la transizione è abilitata, l'esecuzione in entrata interrotta non prosegue nella fase.

È possibile visualizzare o interrompere un'esecuzione in entrata.

Artefatti di input e output

CodePipeline si integra con gli strumenti di sviluppo per verificare le modifiche al codice e quindi creare e distribuire in tutte le fasi del processo di distribuzione continua. Gli artefatti sono i file su cui agiscono le azioni della pipeline, ad esempio file o cartelle con codice dell'applicazione, file di pagine indice, script e così via. Ad esempio, l'artefatto di azione sorgente di Amazon S3 è un nome di file (o

percorso di file) in cui vengono forniti i file di codice sorgente dell'applicazione per l'azione di origine della pipeline e i file vengono generalmente forniti come file ZIP, come il seguente nome di artefatto di esempio: _Windows.zip. SampleApp L'artefatto di output per l'azione di origine, i file di codice sorgente dell'applicazione, sono l'artefatto di output dell'azione di origine e sono anche l'artefatto di input per l'azione successiva, ad esempio un'azione di compilazione. Come altro esempio, un'azione di compilazione potrebbe eseguire comandi di compilazione che compilano il codice sorgente dell'applicazione per un artefatto di input, che è il file del codice sorgente dell'applicazione dall'azione di origine. Consulta la pagina di riferimento sulla configurazione dell'azione per un'azione specifica per i dettagli sui parametri degli artefatti, ad esempio per l'azione. AWS CodeBuild CodeBuild

Le azioni utilizzano artefatti di input e output archiviati nel bucket di artefatti Amazon S3 che hai scelto al momento della creazione della pipeline. CodePipeline comprime e trasferisce i file per gli artefatti di input o output in base al tipo di azione nello stage.

> **Note**
>
> Il bucket di artefatti non è lo stesso bucket utilizzato come posizione del file di origine per una pipeline in cui l'azione sorgente scelta è S3.

Per esempio:

1. CodePipeline attiva l'esecuzione della pipeline quando viene eseguito un commit nel repository di origine, fornendo l'artefatto di output (qualsiasi file da creare) dalla fase Source.
2. L'artefatto di output (qualsiasi file da compilare) della fase precedente viene incluso come artefatto di input per la fase di compilazione. Un artefatto di output (l'applicazione di compilazione) dalla fase di compilazione può essere un'applicazione aggiornata o un'immagine Docker aggiornata compilata in un contenitore.
3. L'artefatto di output del passaggio precedente (l'applicazione creata) viene inserito come elemento di input nella fase di distribuzione, ad esempio negli ambienti di staging o di produzione in. Cloud AWS Puoi distribuire le applicazioni in un parco istanze di distribuzione oppure puoi distribuire le applicazioni basate su container alle attività in esecuzione nei cluster ECS.

Quando si crea o si modifica un'azione, si designano l'artefatto o gli artefatti di input e output per l'azione. Ad esempio, per una pipeline a due fasi con una fase di origine e distribuzione, in Modifica azione, si sceglie il nome dell'artefatto dell'azione di origine per l'artefatto di input per l'azione di distribuzione.

- Quando usi la console per creare la tua prima pipeline, CodePipeline crea un bucket Amazon S3 nella Account AWS stessa Regione AWS e per archiviare gli elementi per tutte le pipeline. Ogni volta che usi la console per creare un'altra pipeline in quella regione, CodePipeline crea una cartella per quella pipeline nel bucket. Questa cartella viene utilizzata per archiviare artefatti della pipeline durante l'esecuzione del processo di rilascio automatico. *Questo bucket è denominato codepipeline- region - 12345EXAMPLE, dove region è la AWS regione in cui è stata creata la pipeline e 12345EXAMPLE è un numero casuale di 12 cifre che assicura che il nome del bucket sia univoco.*

  > **Note**
  >
  > Se hai già un bucket che inizia con codepipeline- region - nella regione in cui stai creando la pipeline, lo utilizza come bucket predefinito. CodePipeline Segue anche l'ordine lessicografico; ad esempio, codepipeline- region-abcexample viene scelto prima di codepipeline- region-defexample.

CodePipeline tronca i nomi degli artefatti, il che può far apparire simili i nomi di alcuni bucket. Anche se il nome dell'artefatto sembra troncato, viene mappato al bucket degli artefatti in modo da non CodePipeline risentire degli artefatti con nomi troncati. La pipeline può funzionare normalmente. Questo non è un problema con la cartella o gli artefatti. I nomi di pipeline hanno un limite di 100 caratteri. Anche se il nome della cartella degli artefatti potrebbe sembrare accorciato, è ancora univoco per la pipeline.

Quando si crea o si modifica una pipeline, è necessario disporre di un bucket di artefatti nella pipeline Account AWS e Regione AWS di un bucket di artefatti per regione in cui si intende eseguire un'azione. Se utilizzi la console per creare una pipeline o azioni interregionali, i bucket di artefatti predefiniti vengono configurati nelle regioni in cui sono presenti le azioni. CodePipeline

Se usi il AWS CLI per creare una pipeline, puoi archiviare gli elementi di quella pipeline in qualsiasi bucket Amazon S3 purché il bucket si trovi nella stessa pipeline. Account AWS Regione AWS Puoi farlo se sei preoccupato di superare i limiti dei bucket Amazon S3 consentiti per il tuo account. Se utilizzi il AWS CLI per creare o modificare una pipeline e aggiungi un'azione interregionale (un'azione con un AWS provider in una regione diversa dalla tua pipeline), devi fornire un bucket di artefatti per ogni regione aggiuntiva in cui intendi eseguire un'azione.

- Ogni operazione dispone di un tipo. A seconda del tipo, l'operazione potrebbe contenere uno o entrambi gli elementi seguenti:
  - Un artefatto di input, che è l'artefatto che viene consumato o elaborato durante l'esecuzione dell'operazione.
  - Un artefatto di output, che è l'output dell'operazione.

  Ogni artefatto di output nella pipeline deve avere un nome univoco. Ogni artefatto di input per un'operazione deve corrispondere all'artefatto di output di un'operazione precedente nella pipeline, a prescindere che tale operazione sia immediatamente precedente a quella in una fase o che venga eseguita diverse fasi prima.

  Un artefatto può essere elaborato da più azioni.

Tipi di pipeline

CodePipeline fornisce i seguenti tipi di tubazioni, che differiscono per caratteristiche e prezzo, in modo da poter adattare le caratteristiche e i costi della pipeline alle esigenze delle applicazioni.

- Le pipeline di tipo V1 hanno una struttura JSON che contiene parametri standard di pipeline, stage e action.
- Le pipeline di tipo V2 hanno la stessa struttura di quelle di tipo V1, oltre a parametri aggiuntivi per la sicurezza del rilascio e la configurazione dei trigger.

Per informazioni sui prezzi di CodePipeline, consulta la sezione Prezzi.

Consulta la CodePipeline riferimento alla struttura della tubazione pagina per i dettagli sui parametri di ogni tipo di pipeline. Per informazioni sul tipo di tubazione da scegliere, consulta. Quale tipo di pipeline è adatto a me?

Quale tipo di pipeline è adatto a me?

Il tipo di pipeline è determinato dall'insieme di caratteristiche e funzionalità supportate da ciascuna versione della pipeline.

Di seguito è riportato un riepilogo dei casi d'uso e delle caratteristiche disponibili per ogni tipo di pipeline.

	Tipo V1	Tipo V2
Caratteristiche		
Casi d'uso	• Implementazioni standard	• Distribuzioni con configurazione derivante dal passaggio di variabili a livello di pipeline in fase di esecuzione • Distribuzioni in cui le pipeline sono configurate per iniziare con tag Git
Variabili a livello di azione	Supportato	Supportato
modalità di esecuzione PARALLELA	Non supportato	Supportata
Variabili a livello di pipeline	Non supportato	Supportata
modalità di esecuzione QUEUED	Non supportato	Supportata
Rollback per le fasi della pipeline	Non supportato	Supportata
La revisione del codice sorgente sostituisce	Non supportato	Supportata
Trigger e filtraggio di tag Git, richieste pull, rami o percorsi di file	Non supportato	Supportata

Per informazioni sui prezzi di CodePipeline, vedi Prezzi.

Utilizzate lo script seguente su una pipeline di tipo V1 per analizzare il costo dello spostamento della pipeline in una pipeline di tipo V2.

Per eseguire un'analisi dei costi per i tipi di pipeline (script)

1. Apri una finestra del terminale. Esegui il comando seguente per creare un nuovo script python chiamato PipelineCostAnalyzer .py.

```
vi PipelineCostAnalyzer.py
```

2. Copia e incolla il codice seguente nello PipelineCostAnalyzerscript.py.

```
import boto3
import sys
import math
from datetime import datetime, timedelta, timezone

if len(sys.argv) < 3:
    raise Exception("Please provide region name and pipeline name as arguments.
 Example usage: python PipelineCostAnalyzer.py us-east-1 MyPipeline")
session = boto3.Session(profile_name='default', region_name=sys.argv[1])
pipeline = sys.argv[2]
codepipeline = session.client('codepipeline')

def analyze_cost_in_v2(pipeline_name):
    if codepipeline.get_pipeline(name=pipeline)['pipeline']['pipelineType'] ==
 'V2':
        raise Exception("Provided pipeline is already of type V2.")
    total_action_executions = 0
    total_blling_action_executions = 0
    total_action_execution_minutes = 0
    cost = 0.0
    hasNextToken = True
    nextToken = ""

    while hasNextToken:
        if nextToken=="":
            response =
 codepipeline.list_action_executions(pipelineName=pipeline_name)
        else:
            response =
 codepipeline.list_action_executions(pipelineName=pipeline_name,
 nextToken=nextToken)
        if 'nextToken' in response:
            nextToken = response['nextToken']
        else:
```

```
            hasNextToken= False
        for action_execution in response['actionExecutionDetails']:
            start_time = action_execution['startTime']
            end_time = action_execution['lastUpdateTime']
            if (start_time < (datetime.now(timezone.utc) - timedelta(days=30))):
                hasNextToken= False
                continue
            total_action_executions += 1
            if (action_execution['status'] in ['Succeeded', 'Failed', 'Stopped']):
                action_owner = action_execution['input']['actionTypeId']['owner']
                action_category = action_execution['input']['actionTypeId']
['category']
                if (action_owner == 'Custom' or (action_owner == 'AWS' and
 action_category == 'Approval')):
                    continue

                total_blling_action_executions += 1
                action_execution_minutes = (end_time -
 start_time).total_seconds()/60
                action_execution_cost = math.ceil(action_execution_minutes) * 0.02
                total_action_execution_minutes += action_execution_minutes
                cost = round(cost + action_execution_cost, 2)

   print ("{:<40}".format('Activity in last 30 days:'))
   print ("| {:<40} | {:<10}".format('_____________________________________',
 '___________________'))
   print ("| {:<40} | {:<10}".format('Total action executions:',
 total_action_executions))
   print ("| {:<40} | {:<10}".format('Total billing action executions:',
 total_blling_action_executions))
   print ("| {:<40} | {:<10}".format('Total billing action execution minutes:',
 round(total_action_execution_minutes, 2)))
   print ("| {:<40} | {:<10}".format('Cost of moving to V2 in $:', cost - 1))

analyze_cost_in_v2(pipeline)
```

3. Esegui lo script sulla pipeline V1 che preferisci in un determinato momento. Regione AWS

   Esegui il comando seguente per eseguire lo script python denominato .py. PipelineCostAnalyzer In questo esempio, la regione è us-west-2.

```
python3 PipelineCostAnalyzer.py us-west-2
```

4. Nel seguente esempio di output dello script, possiamo vedere l'elenco delle esecuzioni di azioni, l'elenco delle esecuzioni di azioni idonee alla fatturazione, la durata totale di queste esecuzioni di azioni e infine il costo dell'aggiornamento della pipeline al tipo V2.

```
Activity in last 30 days:
 | ____________________________________      | ________________
 | Total action executions:                  | 9
 | Total billing action executions:          | 9
 | Total billing action execution minutes:   | 5.59
 | Cost of moving to V2 in $:                | -0.76
```

> **Note**
>
> In questo esempio, il valore negativo nell'ultima riga rappresenta l'importo da risparmiare passando a pipeline di tipo V2.

Guida introduttiva con CodePipeline

Se sei un principiante CodePipeline, puoi seguire i tutorial di questa guida dopo aver seguito i passaggi di questo capitolo per la configurazione.

La CodePipeline console include informazioni utili in un pannello pieghevole che puoi aprire dall'icona delle informazioni o da qualsiasi link Info sulla pagina.

().

Puoi chiudere questo pannello in qualsiasi momento.

La CodePipeline console offre anche un modo per cercare rapidamente le risorse, come repository, progetti di creazione, applicazioni di distribuzione e pipeline. Scegli Go to resource (Vai alla risorsa) o premi il tasto / e digita il nome della risorsa. Qualsiasi corrispondenza verrà visualizzata nell'elenco. Le ricerche rispettano la distinzione tra maiuscole e minuscole. Puoi visualizzare solo le risorse per le quali disponi dell'autorizzazione di visualizzazione. Per ulteriori informazioni, consulta Visualizzazione di risorse nella console.

Prima di poterlo utilizzare AWS CodePipeline per la prima volta, è necessario creare Account AWS e creare il primo utente amministrativo.

Argomenti

Fase 1: Creare un Account AWS utente amministrativo

Registrati per un Account AWS

Se non ne hai uno Account AWS, completa i seguenti passaggi per crearne uno.

Per iscriverti a un Account AWS

1. Apri la pagina all'indirizzo https://portal.aws.amazon.com/billing/signup.

2. Segui le istruzioni online.

   Nel corso della procedura di registrazione riceverai una telefonata, durante la quale sarà necessario inserire un codice di verifica attraverso la tastiera del telefono.

   Quando ti iscrivi a un Account AWS, Utente root dell'account AWSviene creato un. L'utente root dispone dell'accesso a tutte le risorse e tutti i Servizi AWS nell'account. Come procedura consigliata in materia di sicurezza, assegnate l'accesso amministrativo a un utente e utilizzate solo l'utente root per eseguire attività che richiedono l'accesso da parte dell'utente root.

AWS ti invia un'e-mail di conferma dopo il completamento della procedura di registrazione. È possibile visualizzare l'attività corrente dell'account e gestire l'account in qualsiasi momento accedendo all'indirizzo https://aws.amazon.com/ e selezionando Il mio account.

Crea un utente con accesso amministrativo

Dopo esserti registrato Account AWS, proteggi Utente root dell'account AWS AWS IAM Identity Center, abilita e crea un utente amministrativo in modo da non utilizzare l'utente root per le attività quotidiane.

Proteggi i tuoi Utente root dell'account AWS

1. Accedi AWS Management Consolecome proprietario dell'account scegliendo Utente root e inserendo il tuo indirizzo Account AWS email. Nella pagina successiva, inserisci la password.

   Per informazioni sull'accesso utilizzando un utente root, consulta la pagina Signing in as the root user della Guida per l'utente di Accedi ad AWS .
2. Abilita l'autenticazione a più fattori (MFA) per l'utente root.

   Per istruzioni, consulta Abilitare un dispositivo MFA virtuale per l'utente Account AWS root (console) nella Guida per l'utente IAM.

Crea un utente con accesso amministrativo

1. Abilita Centro identità IAM.

   Per istruzioni, consulta Abilitazione di AWS IAM Identity Center nella Guida per l'utente di AWS IAM Identity Center .
2. In IAM Identity Center, concedi l'accesso amministrativo a un utente.

Per un tutorial sull'utilizzo di IAM Identity Center directory come fonte di identità, consulta Configurare l'accesso utente con le impostazioni predefinite IAM Identity Center directory nella Guida per l'AWS IAM Identity Center utente.

Accedi come utente con accesso amministrativo

- Per accedere con l'utente IAM Identity Center, utilizza l'URL di accesso che è stato inviato al tuo indirizzo e-mail quando hai creato l'utente IAM Identity Center.

  Per informazioni sull'accesso utilizzando un utente IAM Identity Center, consulta AWS Accedere al portale di accesso nella Guida per l'Accedi ad AWS utente.

Assegna l'accesso ad altri utenti

1. In IAM Identity Center, crea un set di autorizzazioni che segua la migliore pratica di applicazione delle autorizzazioni con privilegi minimi.

   Per istruzioni, consulta Creare un set di autorizzazioni nella Guida per l'utente.AWS IAM Identity Center
2. Assegna gli utenti a un gruppo, quindi assegna l'accesso Single Sign-On al gruppo.

   Per istruzioni, consulta Aggiungere gruppi nella Guida per l'utente.AWS IAM Identity Center

Fase 2: applicare una policy gestita per l'accesso amministrativo a CodePipeline

È necessario concedere le autorizzazioni con CodePipeline cui interagire. Il modo più rapido per farlo consiste nell'applicare la policy `AWSCodePipeline_FullAccess` gestita all'utente amministrativo.

> **Note**
>
> La `AWSCodePipeline_FullAccess` policy include autorizzazioni che consentono all'utente della console di passare un ruolo IAM a un CodePipeline altro. Servizi AWS Ciò consente al servizio di assumere il ruolo ed eseguire operazioni per conto dell'utente. Quando colleghi la policy a un utente, un ruolo o un gruppo, le autorizzazioni `iam:PassRole` vengono applicate. Verifica che la policy sia applicata solo a utenti attendibili. Quando gli utenti con queste

autorizzazioni utilizzano la console per creare o modificare una pipeline, sono disponibili le seguenti opzioni:

- Crea un ruolo CodePipeline di servizio o scegline uno esistente e trasferisci il ruolo a CodePipeline
- Potrebbe scegliere di creare una regola CloudWatch Events per il rilevamento delle modifiche e passare il ruolo del servizio CloudWatch Events a CloudWatch Events

Per ulteriori informazioni, vedere Concessione a un utente delle autorizzazioni per passare un ruolo a un. Servizio AWS

Note

La `AWSCodePipeline_FullAccess` policy fornisce l'accesso a tutte CodePipeline le azioni e le risorse a cui l'utente IAM ha accesso, nonché a tutte le azioni possibili durante la creazione di fasi in una pipeline, come la creazione di fasi che includono CodeDeploy Elastic Beanstalk o Amazon S3. Come best practice, dovresti concedere agli individui solo le autorizzazioni necessarie per eseguire le proprie funzioni. Per ulteriori informazioni su come limitare gli utenti IAM a un insieme limitato di CodePipeline azioni e risorse, consulta. Rimozione delle autorizzazioni dal ruolo di servizio CodePipeline

Per fornire l'accesso, aggiungi autorizzazioni ai tuoi utenti, gruppi o ruoli:

- Utenti e gruppi in AWS IAM Identity Center:

  Crea un set di autorizzazioni. Segui le istruzioni riportate nella pagina Create a permission set (Creazione di un set di autorizzazioni) nella Guida per l'utente di AWS IAM Identity Center .
- Utenti gestiti in IAM tramite un provider di identità:

  Crea un ruolo per la federazione delle identità. Segui le istruzioni riportate nella pagina Creating a role for a third-party identity provider (federation) (Creazione di un ruolo per un provider di identità di terze parti [federazione]) nella Guida per l'utente di IAM.
- Utenti IAM:
  - Crea un ruolo che l'utente possa assumere. Per istruzioni, consulta la pagina Creating a role for an IAM user (Creazione di un ruolo per un utente IAM) nella Guida per l'utente di IAM.

- (Non consigliato) Collega una policy direttamente a un utente o aggiungi un utente a un gruppo di utenti. Segui le istruzioni riportate nella pagina Aggiunta di autorizzazioni a un utente (console) nella Guida per l'utente di IAM.

Fase 3: Installare AWS CLI

Per richiamare CodePipeline i comandi dalla AWS CLI su una macchina di sviluppo locale, è necessario installare la CLI AWS . Questo passaggio è facoltativo se intendi iniziare a utilizzare solo i passaggi di questa guida per la CodePipeline console.

Per installare e configurare AWS CLI

1. Sul computer locale, scarica e installa il AWS CLI. Ciò ti consentirà di interagire con CodePipeline dalla riga di comando. Per ulteriori informazioni, consulta Configurazione con l'interfaccia a riga di AWS comando.

   > **Note**
   >
   > CodePipeline funziona solo con AWS CLI le versioni 1.7.38 e successive. Per determinare quale versione di quella AWS CLI che potresti aver installato, esegui il comando. aws --version Per aggiornare una versione precedente di AWS CLI alla versione più recente, segui le istruzioni riportate in Disinstallazione di AWS CLI, quindi segui le istruzioni riportate in Installazione di AWS Command Line Interface.

2. Configuratela AWS CLI con il configure comando, come segue:

   ```
   aws configure
   ```

   Quando richiesto, specifica la chiave di AWS accesso e la chiave di accesso AWS segreta dell'utente IAM con CodePipeline cui utilizzerai. Quando viene richiesto il nome della regione predefinito, specificare la regione in cui verrà creata la pipeline, ad esempio `us-east-2`. Quando viene richiesto il formato di output predefinito, specificare `json`. Per esempio:

   ```
   AWS Access Key ID [None]: Type your target AWS access key ID here, and then press
    Enter
   AWS Secret Access Key [None]: Type your target AWS secret access key here, and then
    press Enter
   Default region name [None]: Type us-east-2 here, and then press Enter
   ```

```
Default output format [None]: Type json here, and then press Enter
```

> **Note**
>
> Per ulteriori informazioni su IAM, le chiavi di accesso e le chiavi segrete, consulta Managing Access Keys for IAM Users e How Do I Get Credentials? .
> Per ulteriori informazioni sulle regioni e gli endpoint disponibili CodePipeline, consulta AWS CodePipeline endpoint e quote.

Passaggio 4: Apri la console per CodePipeline

- Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

Passaggi successivi

Hai completato i prerequisiti. Puoi iniziare a usare CodePipeline. Per iniziare a lavorare con CodePipeline, consulta ilCodePipeline tutorial.

Integrazioni di prodotti e servizi con CodePipeline

Per impostazione predefinita, AWS CodePipeline è integrato con una serie di prodotti Servizi AWS e servizi dei partner. Utilizza le informazioni contenute nelle sezioni seguenti per aiutarti CodePipeline a configurare l'integrazione con i prodotti e i servizi che utilizzi.

Le seguenti risorse correlate possono rivelarsi utili durante l'utilizzo di questo servizio.

Argomenti

Integrazioni con tipi di CodePipeline azioni

Le informazioni sulle integrazioni in questo argomento sono organizzate per tipo di CodePipeline azione.

Argomenti

Integrazioni di operazioni di origine

Le seguenti informazioni sono organizzate per tipo di CodePipeline azione e possono aiutarti CodePipeline a configurare l'integrazione con i seguenti provider di source action.

Argomenti

- Azioni di origine di Amazon S3
- Connessioni a Bitbucket Cloud, GitHub (versione 2), GitHub Enterprise Server, GitLab .com e gestione automatica GitLab
- CodeCommit azioni di origine
- GitHub (versione 1) azioni di origine

Azioni di origine di Amazon ECR

Amazon ECR è un servizio di archiviazione di immagini AWS Docker. È possibile utilizzare i comandi pull e push di Docker per caricare immagini Docker nel repository. L'URI e l'immagine del repository Amazon ECR vengono utilizzati nelle definizioni delle attività di Amazon ECS per fare riferimento alle informazioni sull'immagine di origine.

Ulteriori informazioni:

- Per visualizzare i parametri di configurazione e un frammento JSON/YAML di esempio, consulta Amazon ECR
- Creare una pipeline in CodePipeline
- Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy

Azioni di origine di Amazon S3

Amazon S3 è uno storage per Internet. È possibile utilizzare Amazon S3 per memorizzare e recuperare qualsiasi volume di dati, in qualunque momento e da qualunque luogo tramite il Web. Puoi configurare l'utilizzo CodePipeline di un bucket Amazon S3 con versione diversa come azione sorgente per il tuo codice.

> **Note**
>
> Amazon S3 può anche essere incluso in una pipeline come azione di distribuzione.

Ulteriori informazioni:

- Per visualizzare i parametri di configurazione e un esempio di frammento JSON/YAML, consulta Azione all'origine di Amazon S3

- Fase 1: creazione di un bucket S3 per l'applicazione
- Creazione di una pipeline (CLI)
- CodePipeline utilizza Amazon EventBridge (in precedenza Amazon CloudWatch Events) per rilevare le modifiche nel bucket di origine Amazon S3. Per informazioni, consulta Integrazioni generali con CodePipeline.

Connessioni a Bitbucket Cloud, GitHub (versione 2), GitHub Enterprise Server, GitLab .com e gestione automatica GitLab

Le connessioni (`CodeStarSourceConnection`azioni) vengono utilizzate per accedere a Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com di terze parti o al repository autogestito. GitLab

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hong Kong), Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Africa (Città del Capo), Medio Oriente (Bahrain), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna), Europa (Zurigo), Israele (Tel Aviv) o AWS GovCloud (Stati Uniti occidentali). Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Bitbucket Cloud	Puoi configurare l'utilizzo CodePipeline di un repository Bitbucket Cloud come sorgente per il tuo codice. Devi aver precedentemente creato un account Bitbucket e almeno un repository Bitbucket Cloud. Puoi aggiungere un'azione sorgente per il tuo repository Bitbucket Cloud creando una pipeline o modificandone una esistente. **Note** È possibile creare connessioni a un repository Bitbucket Cloud. I tipi di provider Bitbucket installati, ad esempio Bitbucket Server, non sono supportati.

	È possibile configurare le risorse denominate connessioni per permettere alle pipeline di accedere ai repository di codice di terze parti. Quando crei una connessione, installi l' AWS CodeStar app con il tuo repository di codice di terze parti e poi la associ alla tua connessione. Per Bitbucket Cloud, utilizza l'opzione Bitbucket nella console o l'azione nella CLI`CodestarSourceConnection` . Per informazioni, consulta Connessioni Bitbucket Cloud. Puoi usare l'opzione Full clone per questa azione per fare riferimento ai metadati Git del repository in modo che le azioni a valle possano eseguire direttamente i comandi Git. Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild Ulteriori informazioni: • Per visualizzare i parametri di configurazione e un frammento JSON/YAML di esempio, consulta. CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite • Per visualizzare un tutorial introduttivo che crea una pipeline con una sorgente Bitbucket Cloud, consulta Guida introduttiva alle connessioni.
GitHub o Enterprise Cloud GitHub	È possibile CodePipeline configurare l'utilizzo di un GitHub repository come sorgente per il codice. È necessario aver precedentemente creato un GitHub account e almeno un GitHub repository. Puoi aggiungere un'azione sorgente per il tuo GitHub repository creando una pipeline o modificandone una esistente. È possibile configurare le risorse denominate connessioni per permettere alle pipeline di accedere ai repository di codice di terze parti. Quando crei una connessione, installi l' AWS CodeStar app con il tuo repository di codice di terze parti e poi la associ alla tua connessione. Utilizza l'opzione provider GitHub (versione 2) nella console o l'`CodestarSourceConnection` azione nella CLI. Per informazioni, consulta GitHub connessioni.

	Puoi usare l'opzione Full clone per questa azione per fare riferimento ai metadati Git del repository in modo che le azioni a valle possano eseguire direttamente i comandi Git. Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild Ulteriori informazioni: • Per visualizzare i parametri di configurazione e un frammento JSON/YAML di esempio, vedere CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite • Per un tutorial che mostra come connettersi a un GitHub repository e utilizzare l'opzione Full clone, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline • L' GitHub azione corrente è l'azione sorgente della versione 2 per. GitHub L' GitHub azione della versione 1 è gestita con l'autenticazione tramite token OAuth. Sebbene non sia consigliabile utilizzare l'azione della GitHub versione 1, le pipeline esistenti con l'azione della GitHub versione 1 continueranno a funzionare senza alcun impatto. Ora puoi utilizzare un'azione di CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite origine nella tua pipeline che gestisce l'azione di GitHub origine con GitHub le app. Se disponi di una pipeline che utilizza l' GitHub azione della versione 1, consulta i passaggi per aggiornarla in modo che utilizzi un' GitHub azione della versione 2 in. Aggiornare un'azione di origine della GitHub versione 1 a un'azione di origine della GitHub versione 2
GitHub Server aziendale	È possibile CodePipeline configurare l'utilizzo di un repository GitHub Enterprise Server come sorgente per il codice. È necessario aver precedent emente creato un GitHub account e almeno un GitHub repository. È possibile aggiungere un'azione di origine per il repository di GitHub Enterprise Server creando una pipeline o modificandone una esistente. È possibile configurare le risorse denominate connessioni per permettere alle pipeline di accedere ai repository di codice di terze parti. Quando si crea una connessione, si installa l' AWS CodeStar app con l'archivio di codice di terze parti e quindi la si associa alla connessione.

Utilizza l'opzione provider GitHub Enterprise Server nella console o l'`CodestarSourceConnection` azione nella CLI. Per informazioni, consulta GitHub Connessioni Enterprise Server.

> **Important**
>
> AWS CodeStar Connections non supporta la versione 2.22.0 di GitHub Enterprise Server a causa di un problema noto nella versione. Per connetterti, esegui l'aggiornamento alla versione 2.22.1 o all'ultim a versione disponibile.

Puoi usare l'opzione Full clone per questa azione per fare riferimento ai metadati Git del repository in modo che le azioni a valle possano eseguire direttamente i comandi Git. Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild

Ulteriori informazioni:

- Per visualizzare i parametri di configurazione e un frammento JSON/YAML di esempio, vedere CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite
- Per un tutorial che mostra come connettersi a un GitHub repository e utilizzare l'opzione Full clone, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline

GitLab.com

È possibile CodePipeline configurare l'utilizzo di un repository GitLab .com come sorgente per il codice. È necessario aver creato in precedenza un account GitLab .com e almeno un repository GitLab .com. Puoi aggiunger e un'azione source per il tuo repository GitLab .com creando una pipeline o modificandone una esistente.

Utilizza l'opzione GitLabprovider nella console o l'`CodestarSourceConn ection` azione con il `GitLab` provider nella CLI. Per informazioni, consulta GitLabconnessioni.com.

Ulteriori informazioni:

	• Per visualizzare i parametri di configurazione e uno snippet JSON/YAML di esempio, vedi CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite
GitLab autogestito	È possibile CodePipeline configurare l'utilizzo di un'installazione GitLab autogestita come origine del codice. È necessario aver precedentemente creato un GitLab account e disporre di un abbonamento per la gestione automatica GitLab (Enterprise Edition o Community Edition). Puoi aggiungere un'azione sorgente per il tuo repository GitLab autogestito creando una pipeline o modificandone una esistente. È possibile configurare le risorse denominate connessioni per permettere alle pipeline di accedere ai repository di codice di terze parti. Quando crei una connessione, installi l' AWS CodeStar app con il tuo repository di codice di terze parti e poi la associ alla tua connessione. Utilizza l'opzione provider GitLab autogestito nella console o l'`CodestarSourceConnection` azione nella CLI. Per informazioni, consulta Connessioni per la GitLab gestione automatica. Puoi usare l'opzione Full clone per questa azione per fare riferimento ai metadati Git del repository in modo che le azioni a valle possano eseguire direttamente i comandi Git. Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild Ulteriori informazioni: • Per visualizzare i parametri di configurazione e un frammento JSON/YAML di esempio, vedere CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite • Per i passaggi per creare una connessione con questo tipo di provider, vedi. Connessioni per la GitLab gestione automatica

CodeCommit azioni di origine

CodeCommitè un servizio di controllo delle versioni che puoi utilizzare per archiviare e gestire in modo privato risorse (come documenti, codice sorgente e file binari) nel cloud. È possibile CodePipeline configurare l'utilizzo di un ramo in un CodeCommit repository come sorgente per il codice. Crea il repository e associalo a una directory di lavoro sul tuo computer locale. Quindi è possibile creare una pipeline che utilizza il ramo come parte di un'operazione di origine in un fase. È possibile connettersi al CodeCommit repository creando una pipeline o modificandone una esistente.

Puoi usare l'opzione Full clone per questa azione per fare riferimento ai metadati Git del repository in modo che le azioni a valle possano eseguire direttamente i comandi Git. Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild

Ulteriori informazioni:

- Per visualizzare i parametri di configurazione e un frammento JSON/YAML di esempio, consulta. CodeCommit
- Tutorial: crea una pipeline semplice (CodeCommitrepository)
- CodePipeline utilizza Amazon CloudWatch Events per rilevare le modifiche nei CodeCommit repository utilizzati come origine per una pipeline. A ogni operazione di origine corrisponde una regola di evento. Questa regola di evento avvia la pipeline quando si verifica una modifica nel repository. Per informazioni, consulta Integrazioni generali con CodePipeline.

GitHub (versione 1) azioni di origine

L'azione della GitHub versione 1 è gestita con le app OAuth. Nelle regioni disponibili, puoi anche utilizzare un'azione di CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite origine nella tua pipeline che gestisce l'azione di GitHub origine con le app. GitHub Se disponi di una pipeline che utilizza l'azione della GitHub versione 1, consulta i passaggi per aggiornarla in modo da utilizzare un'azione della GitHub versione 2 in. Aggiornare un'azione di origine della GitHub versione 1 a un'azione di origine della GitHub versione 2

> **Note**
>
> Sebbene non sia consigliabile utilizzare l'azione della GitHub versione 1, le pipeline esistenti con l'azione della GitHub versione 1 continueranno a funzionare senza alcun impatto.

Ulteriori informazioni:

- Per ulteriori informazioni sull'accesso basato su OAuth rispetto GitHub all'accesso basato su app GitHub , consulta. https://docs.github.com/en/developers/apps/differences-between-github-apps-and-oauth-apps
- Per visualizzare un'appendice contenente i dettagli delle GitHub azioni della versione 1, vedere. Appendice A: azioni di origine della GitHub versione 1

Integrazioni di operazioni di compilazione

Le seguenti informazioni sono organizzate per tipo di CodePipeline azione e possono aiutarti CodePipeline a configurare l'integrazione con i seguenti provider di azioni di build.

Argomenti

- CodeBuild creare azioni
- CloudBees costruire azioni
- Operazioni di compilazione Jenkins
- TeamCity costruisci azioni

CodeBuild creare azioni

CodeBuildè un servizio di compilazione completamente gestito che compila il codice sorgente, esegue test unitari e produce artefatti pronti per l'implementazione.

È CodeBuild possibile aggiungere un'azione di compilazione alla fase di creazione di una pipeline. Per ulteriori informazioni, consulta il riferimento alla configurazione delle CodePipeline azioni perAWS CodeBuild.

> **Note**
>
> CodeBuild può anche essere incluso in una pipeline come azione di test, con o senza un output di build.

Ulteriori informazioni:

- Per visualizzare i parametri di configurazione e un frammento di codice JSON/YAML di esempio, consulta. AWS CodeBuild
- CodeBuildChe cos'è?
- CodeBuild— Servizio di costruzione completamente gestito

CloudBees costruire azioni

Puoi configurarlo CodePipeline per CloudBeescreare o testare il codice in una o più azioni in una pipeline.

Ulteriori informazioni:

- re:Invent 2017: Cloud First con AWS

Operazioni di compilazione Jenkins

Puoi configurare l'utilizzo CodePipeline di Jenkins CI per creare o testare il codice in una o più azioni in una pipeline. È necessario aver precedentemente creato un progetto Jenkins e installato e configurato il CodePipeline Plugin for Jenkins per quel progetto. Puoi connetterti al progetto Jenkins creando una nuova pipeline o modificandone una esistente.

L'accesso per Jenkins è configurato in base al progetto. È necessario installare il CodePipeline Plugin for Jenkins su ogni istanza Jenkins con cui si desidera utilizzare. CodePipeline È inoltre necessario configurare CodePipeline l'accesso al progetto Jenkins. Proteggi il progetto Jenkins configurandolo per accettare solo connessioni HTTPS/SSL. Se il tuo progetto Jenkins è installato su un'istanza Amazon EC2, valuta la possibilità di fornire le AWS tue credenziali AWS CLI installandole su ogni istanza. Quindi configura un AWS profilo su quelle istanze con le credenziali che desideri utilizzare per le connessioni. Questa è un'alternativa all'aggiunta e alla memorizzazione tramite l'interfaccia Web Jenkins.

Ulteriori informazioni:

- Accessing Jenkins
- Tutorial: creazione di una pipeline a quattro fasi

TeamCity costruisci azioni

Puoi configurarlo CodePipeline per TeamCitycreare e testare il codice in una o più azioni in una pipeline.

Ulteriori informazioni:

- TeamCity Plugin per CodePipeline

Integrazioni di operazioni di test

Le seguenti informazioni sono organizzate per tipo di CodePipeline azione e possono aiutarti CodePipeline a configurare l'integrazione con i seguenti fornitori di azioni di test.

Argomenti

- CodeBuild azioni di test
- AWS Device Farm azioni di test
- Azioni di test di Ghost Inspector
- OpenText LoadRunner Azioni di test nel cloud

CodeBuild azioni di test

CodeBuildè un servizio di compilazione completamente gestito nel cloud. CodeBuild compila il codice sorgente, esegue test unitari e produce artefatti pronti per l'implementazione.

Puoi aggiungerlo CodeBuild a una pipeline come azione di test. Per ulteriori informazioni, vedere il riferimento alla configurazione delle CodePipeline azioni perAWS CodeBuild.

> **Note**
>
> CodeBuild può anche essere incluso in una pipeline come azione di compilazione, con un artefatto di output di compilazione obbligatorio.

Ulteriori informazioni:

- Per visualizzare i parametri di configurazione e un frammento di codice JSON/YAML di esempio, consulta. AWS CodeBuild

- CodeBuildChe cos'è?

AWS Device Farm azioni di test

AWS Device Farm è un servizio di test delle app che puoi utilizzare per effettuare test e interagire con applicazioni Android, iOS e Web su telefoni e tablet fisici reali. Puoi configurarlo CodePipeline per AWS Device Farm testare il codice in una o più azioni in una pipeline. AWS Device Farm consente di caricare i propri test o utilizzare test di compatibilità integrati e privi di script. Poiché i test vengono eseguiti in parallelo, i test su più dispositivi iniziano entro pochi minuti. Un rapporto di test che contiene risultati di alto livello, log di basso livello, pixel-to-pixel schermate e dati sulle prestazioni viene aggiornato man mano che i test vengono completati. AWS Device Farm supporta il test di app Android, iOS e Fire OS native e ibride, incluse quelle create con Titanium PhoneGap, Xamarin, Unity e altri framework. Supporta l'accesso remoto di app Android, che consente di interagire direttamente con i dispositivi di test.

Ulteriori informazioni:

- Per visualizzare i parametri di configurazione e un esempio di snippet JSON/YAML, consulta. AWS Device Farm
- AWS Device Farm Che cos'è?
- Utilizzo AWS Device Farm in una fase CodePipeline di test

Azioni di test di Ghost Inspector

Puoi configurare l'utilizzo CodePipeline di Ghost Inspector per testare il codice in una o più azioni in una pipeline.

Ulteriori informazioni:

- Documentazione di Ghost Inspector per l'integrazione dei servizi con CodePipeline

OpenText LoadRunner Azioni di test nel cloud

Puoi configurare l'utilizzo CodePipeline di OpenText LoadRunner Cloud in una o più azioni in una pipeline.

Ulteriori informazioni:

- LoadRunner Documentazione cloud per l'integrazione con CodePipeline

Integrazioni di operazioni di distribuzione

Le seguenti informazioni sono organizzate per tipo di CodePipeline azione e possono aiutarti CodePipeline a configurare l'integrazione con i seguenti provider di azioni di distribuzione.

Argomenti

- Azioni di distribuzione di Amazon S3
- AWS AppConfig distribuire azioni
- AWS CloudFormation distribuire azioni
- AWS CloudFormation StackSets implementare azioni
- Azioni di distribuzione di Amazon ECS
- Azioni di distribuzione di Elastic Beanstalk
- AWS OpsWorks distribuire azioni
- Azioni di distribuzione del Service Catalog
- Azioni di distribuzione di Amazon Alexa
- CodeDeploy distribuire azioni
- XebiaLabs azioni di distribuzione

Azioni di distribuzione di Amazon S3

Amazon S3 è uno storage per Internet. È possibile utilizzare Amazon S3 per memorizzare e recuperare qualsiasi volume di dati, in qualunque momento e da qualunque luogo tramite il Web. Puoi aggiungere un'azione a una pipeline che utilizza Amazon S3 come provider di distribuzione.

> **Note**
>
> Amazon S3 può anche essere incluso in una pipeline come source action.

Ulteriori informazioni:

- Creare una pipeline in CodePipeline

- Tutorial: crea una pipeline che utilizzi Amazon S3 come provider di distribuzione

AWS AppConfig distribuire azioni

AWS AppConfig è una capacità di AWS Systems Manager creare, gestire e distribuire rapidamente configurazioni di applicazioni. Puoi utilizzarlo AppConfig con applicazioni ospitate su istanze EC2, contenitori AWS Lambda, applicazioni mobili o dispositivi IoT.

Ulteriori informazioni:

- CodePipeline Riferimento alla configurazione delle azioni per AWS AppConfig
- Tutorial: crea una pipeline da utilizzare AWS AppConfig come provider di distribuzione

AWS CloudFormation distribuire azioni

AWS CloudFormationoffre agli sviluppatori e agli amministratori di sistema un modo semplice per creare e gestire una raccolta di AWS risorse correlate, utilizzando modelli per fornire e aggiornare tali risorse. Puoi utilizzare i modelli di esempio del servizio o crearne di nuovi personalizzati. I modelli descrivono le AWS risorse e le eventuali dipendenze o parametri di runtime necessari per eseguire l'applicazione.

Il AWS Serverless Application Model (AWS SAM) si estende AWS CloudFormation per fornire un modo semplificato per definire e distribuire applicazioni serverless. AWS SAM supporta le API Amazon API Gateway, le funzioni AWS Lambda e le tabelle Amazon DynamoDB. Puoi usare CodePipeline with AWS CloudFormation and the AWS SAM per distribuire continuamente le tue applicazioni serverless.

È possibile aggiungere un'azione a una pipeline che utilizza AWS CloudFormation come provider di distribuzione. Quando lo utilizzi AWS CloudFormation come fornitore di distribuzione, puoi intervenire sugli AWS CloudFormation stack e sui set di modifiche come parte dell'esecuzione di una pipeline. AWS CloudFormation può creare, aggiornare, sostituire ed eliminare stack e set di modifiche durante l'esecuzione di una pipeline. Di conseguenza, AWS è possibile creare, fornire, aggiornare o terminare risorse personalizzate durante l'esecuzione di una pipeline in base alle specifiche fornite nei modelli e nelle AWS CloudFormation definizioni dei parametri.

Ulteriori informazioni:

- CodePipeline Riferimento alla configurazione delle azioni per AWS CloudFormation

- Distribuzione continua con CodePipeline: scopri come utilizzare per CodePipeline creare un flusso di lavoro di distribuzione continua per AWS CloudFormation.
- Automatizzazione della distribuzione di applicazioni basate su Lambda: scopri come utilizzare il modello di applicazione AWS serverless e creare un flusso di lavoro di distribuzione continua AWS CloudFormation per la tua applicazione basata su Lambda.

AWS CloudFormation StackSets implementare azioni

AWS CloudFormationti offre un modo per distribuire risorse su più account e AWS regioni.

È possibile utilizzare CodePipeline with AWS CloudFormation per aggiornare la definizione del set di stack e distribuire aggiornamenti alle istanze.

È possibile aggiungere le seguenti azioni a una pipeline da utilizzare AWS CloudFormation StackSets come provider di distribuzione.

- CloudFormationStackSet
- CloudFormationStackInstances

Ulteriori informazioni:

- CodePipeline Riferimento alla configurazione delle azioni per AWS CloudFormation StackSets
- Tutorial: creare una pipeline con azioni AWS CloudFormation StackSets di distribuzione

Azioni di distribuzione di Amazon ECS

Amazon ECS è un servizio di gestione dei container altamente scalabile e ad alte prestazioni che consente di eseguire applicazioni basate su container in. Cloud AWS Quando crei una pipeline, puoi selezionare Amazon ECS come fornitore di distribuzione. Una modifica al codice nel tuo repository di controllo del codice sorgente attiva la pipeline per creare una nuova immagine Docker, inviarla al registro dei contenitori e quindi distribuire l'immagine aggiornata in Amazon ECS. Puoi anche utilizzare l'azione del provider ECS (Blue/Green) CodePipeline per indirizzare e distribuire il traffico verso Amazon ECS con. CodeDeploy

Ulteriori informazioni:

- Che cos'è Amazon ECS?
- Tutorial: distribuzione continua con CodePipeline

- Creare una pipeline in CodePipeline
- Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy

Azioni di distribuzione di Elastic Beanstalk

Elastic Beanstalk è un servizio per la distribuzione e la scalabilità di applicazioni e servizi Web sviluppati con Java, .NET, PHP, Node.js, Python, Ruby, Go e Docker su server familiari come Apache, Nginx, Passenger e IIS. Puoi configurare l'utilizzo CodePipeline di Elastic Beanstalk per distribuire il codice. È possibile creare l'applicazione e l'ambiente Elastic Beanstalk da utilizzare in un'azione di distribuzione in una fase prima di creare la pipeline o quando si utilizza la procedura guidata Create Pipeline.

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hyderabad), Asia Pacifico (Melbourne), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna) o Europa (Zurigo). Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline

Ulteriori informazioni:

- Guida introduttiva a Elastic Beanstalk
- Creare una pipeline in CodePipeline

AWS OpsWorks distribuire azioni

AWS OpsWorks è un servizio di gestione della configurazione che consente di configurare e utilizzare applicazioni di tutte le forme e dimensioni utilizzando Chef. Utilizzando AWS OpsWorks Stacks, è possibile definire l'architettura dell'applicazione e le specifiche di ogni componente, inclusa l'installazione del pacchetto, la configurazione del software e le risorse come lo storage. È possibile configurare l'utilizzo CodePipeline AWS OpsWorks Stacks per distribuire il codice insieme ai ricettari e alle applicazioni Chef personalizzati in. AWS OpsWorks

- Custom Chef Cookbooks: AWS OpsWorks utilizza Chef Cookbooks per gestire attività come l'installazione e la configurazione di pacchetti e la distribuzione di applicazioni.

- Applicazioni: un' AWS OpsWorks applicazione è costituita da codice che si desidera eseguire su un server di applicazioni. Il codice dell'applicazione è archiviato in un repository, ad esempio un bucket Amazon S3.

Prima di creare la pipeline, crei lo stack e il AWS OpsWorks layer. È possibile creare l' AWS OpsWorks applicazione da utilizzare in un'azione di distribuzione in una fase prima di creare la pipeline o quando si utilizza la procedura guidata Create Pipeline.

CodePipeline il supporto per AWS OpsWorks è attualmente disponibile solo nella regione Stati Uniti orientali (Virginia settentrionale) (us-east-1).

Ulteriori informazioni:

- Utilizzo con CodePipeline AWS OpsWorks Stacks
- Libri di ricette e ricette
- AWS OpsWorks Web

Azioni di distribuzione del Service Catalog

Service Catalog consente alle organizzazioni di creare e gestire cataloghi di prodotti approvati per l'uso su AWS.

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna), Europa (Zurigo) o Israele (Tel Aviv). Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline

È possibile CodePipeline configurare la distribuzione degli aggiornamenti e delle versioni dei modelli di prodotto in Service Catalog. È possibile creare il prodotto Service Catalog da utilizzare in un'azione di distribuzione e quindi utilizzare la procedura guidata Create Pipeline per creare la pipeline.

Ulteriori informazioni:

- Tutorial: crea una pipeline da distribuire su Service Catalog
- Creare una pipeline in CodePipeline

Azioni di distribuzione di Amazon Alexa

Amazon Alexa Skills Kit ti consente di creare e distribuire le competenze basate sul cloud agli utenti di dispositivi compatibili con Alexa.

> **Note**
>
> Questa funzionalità non è disponibile nella regione Asia Pacifico (Hong Kong) o Europa (Milano). Per utilizzare altre azioni di distribuzione disponibili in quella regione, vediIntegrazioni di operazioni di distribuzione.

Puoi aggiungere un'operazione a una pipeline che utilizza Alexa Skills Kit come provider di distribuzione. Le modifiche di origine vengono rilevate dalla pipeline, che quindi distribuisce gli aggiornamenti alla competenza Alexa nel servizio Alexa.

Ulteriori informazioni:

- Tutorial: creazione di una pipeline che distribuisce una competenza Amazon Alexa

CodeDeploy distribuire azioni

CodeDeploycoordina le distribuzioni delle applicazioni su istanze Amazon EC2, istanze locali o entrambe. Puoi configurarlo per utilizzarlo CodePipeline per distribuire il codice. CodeDeploy È possibile creare l' CodeDeploy applicazione, la distribuzione e il gruppo di distribuzione da utilizzare in un'azione di distribuzione in una fase prima di creare la pipeline o quando si utilizza la procedura guidata Crea pipeline.

Ulteriori informazioni:

- Passaggio 3: Creare un'applicazione in CodeDeploy
- Tutorial: crea una pipeline semplice (CodeCommitrepository)

XebiaLabs azioni di distribuzione

È possibile configurare l'utilizzo CodePipeline XebiaLabsper distribuire il codice in una o più azioni in una pipeline.

Ulteriori informazioni:

- Utilizzo di XL Deploy con CodePipeline

Integrazione dell'azione di approvazione con Amazon Simple Notification Service

Amazon SNS è un servizio di notifica push veloce, flessibile e completamente gestito che ti consente di inviare messaggi singoli o di inviarli a un gran numero di destinatari. Amazon SNS semplifica ed economica l'invio di notifiche push a utenti di dispositivi mobili, destinatari di e-mail o persino l'invio di messaggi ad altri servizi distribuiti.

Quando crei una richiesta di approvazione manuale in CodePipeline, puoi facoltativamente pubblicarla su un argomento in Amazon SNS in modo che tutti gli utenti IAM abbonati ricevano una notifica che l'azione di approvazione è pronta per essere esaminata.

Ulteriori informazioni:

- Che cos'è Amazon SNS?
- Concedi le autorizzazioni Amazon SNS per un ruolo di servizio CodePipeline

Integrazioni di operazioni di invocazione

Le seguenti informazioni sono organizzate per tipo di CodePipeline azione e possono aiutarti CodePipeline a configurare l'integrazione con i seguenti provider di azioni di invoke.

Argomenti

- Azioni di richiamo Lambda
- Azioni di richiamo Snyk
- Step Functions invoca le azioni

Azioni di richiamo Lambda

Lambda consente di eseguire codice senza effettuare il provisioning o gestire i server. Puoi CodePipeline configurare l'uso delle funzioni Lambda per aggiungere flessibilità e funzionalità alle tue pipeline. È possibile creare la funzione Lambda da aggiungere come azione in una fase prima di creare la pipeline o quando si utilizza la procedura guidata Crea pipeline.

Ulteriori informazioni:

- CodePipeline Riferimento alla configurazione delle azioni per AWS Lambda
- Invoca una AWS Lambda funzione in una pipeline in CodePipeline

Azioni di richiamo Snyk

Puoi configurare l'utilizzo CodePipeline di Snyk per proteggere i tuoi ambienti open source rilevando e correggendo le vulnerabilità di sicurezza e aggiornando le dipendenze nel codice dell'applicazione e nelle immagini dei contenitori. Puoi anche utilizzare l'azione Snyk per automatizzare i controlli dei test di sicurezza nella tua CodePipeline pipeline.

Ulteriori informazioni:

- CodePipeline Riferimento alla configurazione delle azioni per Riferimento alla struttura d'azione Snyk
- Automatizza la scansione delle vulnerabilità con Snyk AWS CodePipeline

Step Functions invoca le azioni

Step Functions consente di creare e configurare macchine a stati. È possibile configurare l'utilizzo delle azioni CodePipeline di invoca di Step Functions per attivare le esecuzioni di macchine a stati.

Ulteriori informazioni:

- CodePipeline Riferimento alla configurazione delle azioni per AWS Step Functions
- Tutorial: utilizzare un'azione di AWS Step Functions richiamo in una pipeline

Integrazioni generali con CodePipeline

Le seguenti Servizio AWS integrazioni non si basano sui tipi di CodePipeline azioni.

Amazon CloudWatch	Amazon CloudWatch monitora le tue AWS risorse. Ulteriori informazioni: • Che cos'è Amazon CloudWatch?

Amazon EventBridge	Amazon EventBridge è un servizio web che rileva le modifiche apportate all'utente in Servizi AWS base a regole da te definite e richiama un'azione in uno o più casi specifici Servizi AWS quando si verifica una modifica. • Avvia automaticamente l'esecuzione di una pipeline quando qualcosa cambia: puoi CodePipeline configurarla come destinazione nelle regole impostate in Amazon EventBridge. Questo consente di impostare le pipeline per essere avviata automaticamente quando un altro servizio cambia. Ulteriori informazioni: • Che cos'è Amazon EventBridge? • Avvia una pipeline in CodePipeline. • CodeCommit azioni di origine e EventBridge • Ricevi notifiche quando lo stato di una pipeline cambia: puoi impostare EventBridge regole per rilevare e reagire ai cambiamenti nello stato di esecuzione per una pipeline, una fase o un'azione. Ulteriori informazioni: • Monitoraggio CodePipeline degli eventi • Tutorial: imposta una regola CloudWatch Events per ricevere notifiche e-mail per le modifiche allo stato della pipeline
AWS Cloud9	AWS Cloud9 è un IDE online, a cui puoi accedere tramite il tuo browser web. L'IDE offre una ricca esperienza di modifica del codice con supporto per diversi linguaggi di programmazione e debugger runtime, nonché un terminale integrato. In background, un'istanza Amazon EC2 ospita un ambiente di AWS Cloud9 sviluppo. Per ulteriori informazioni, consulta la Guida per l'utente AWS Cloud9. Ulteriori informazioni: • Configurazione di AWS Cloud9

AWS CloudTrail	CloudTrailacquisisce le chiamate AWS API e gli eventi correlati effettuati da o per conto di un AWS account e invia i file di log a un bucket Amazon S3 specificato dall'utente. Puoi CloudTrail configurare l'acquisizione delle chiamate API dalla CodePipeline console, dei CodePipeline comandi da e dall'API. AWS CLI CodePipeline Ulteriori informazioni: • Registrazione delle chiamate CodePipeline API con AWS CloudTrail
AWS CodeStar Notifiche	È possibile configurare le notifiche per mettere a conoscenza gli utenti di modifiche importanti, ad esempio quando una pipeline avvia l'esecuzione. Per ulteriori informazioni, consulta Creazione di una regola di notifica.

AWS Key Management Service	AWS KMS è un servizio gestito che semplifica la creazione e il controllo di chiavi di crittografia per la codifica dei dati. Per impostazione predefinita, CodePipeline vengono utilizzati AWS KMS per crittografare gli artefatti per le pipeline archiviate nei bucket Amazon S3. Ulteriori informazioni: • Per creare una pipeline che utilizzi un bucket di origine, un bucket di artefatti e un ruolo di servizio da un AWS account e CodeDeploy risorse da un altro AWS account, devi creare una chiave KMS gestita dal cliente, aggiungere la chiave alla pipeline e configurare le politiche e i ruoli degli account per consentire l'accesso tra account. Per ulteriori informazioni, consulta Crea una pipeline CodePipeline che utilizzi le risorse di un altro account AWS. • Per creare una pipeline da un AWS account che distribuisce uno AWS CloudFormation stack a un altro AWS account, devi creare una chiave KMS gestita dal cliente, aggiungere la chiave alla pipeline e configurare le politiche e i ruoli dell'account per distribuire lo stack su un altro AWS account. Per ulteriori informazioni, vedi Come si usa CodePipeline per distribuire uno stack in un account diverso? AWS CloudFormation • Per configurare la crittografia lato server per il bucket di artefatti S3 della pipeline, puoi utilizzare la chiave KMS AWS gestita predefinita o creare una chiave KMS gestita dal cliente e configurare la policy del bucket per utilizzare la chiave di crittografia. Per ulteriori informazioni, consulta Configura la crittografia lato server per gli artefatti archiviati in Amazon S3 per CodePipeline. Per un AWS KMS key, puoi usare l'ID chiave, la chiave ARN o l'alias ARN. **Note** Gli alias sono riconosciuti solo nell'account che ha creato la chiave KMS. Per le operazioni tra account, puoi utilizzare solo l'ID della chiave o l'ARN della chiave per identificare la chiave.

	Le operazioni tra account comportano l'utilizzo del ruolo dell'altro account (AccountB), pertanto specificando l'ID chiave verrà utilizzata la chiave dell'altro account (AccountB).

Esempi della community

Le seguenti sezioni forniscono collegamenti a post di blog, articoli ed esempi della community.

> **Note**
>
> Questi collegamenti sono forniti solo a scopo informativo e non devono essere considerati né un elenco completo né un'approvazione del contenuto degli esempi. AWS non è responsabile per il contenuto o l'accuratezza dei contenuti esterni.

Argomenti

Esempi di integrazione: post di blog

- Monitoraggio dello stato della AWS CodePipeline build dal repository Git di terze parti

  Scopri come configurare risorse che mostreranno la tua pipeline e lo stato delle azioni nel tuo repository di terze parti, permettendo allo sviluppatore di monitorare facilmente lo stato senza cambiare contesto.

  Pubblicato a marzo 2021

- CI/CD completo con AWS CodeCommit, AWS CodeBuild, e AWS CodeDeployAWS CodePipeline

  Scopri come configurare una pipeline che utilizza i CodeDeploy servizi CodeCommit,, e per compilare CodePipeline CodeBuild, creare e installare un'applicazione Java con controllo di versione su un set di istanze Amazon EC2 Linux.

  Pubblicato a settembre 2020

- Come eseguire la distribuzione GitHub da Amazon EC2 con CodePipeline

Scopri come configurare CodePipeline da zero per distribuire le filiali di sviluppo, test e produzione in gruppi di distribuzione separati. Scopri come utilizzare e configurare i ruoli IAM, l' CodeDeploy agente e CodeDeploy, insieme a. CodePipeline

Pubblicato aprile 2020

- Test e creazione di pipeline CI/CD per Step Functions AWS

  Scopri come configurare le risorse che coordineranno la tua macchina a stati Step Functions e la tua pipeline.

  Pubblicato a marzo 2020

- Implementazione DevSecOps utilizzando CodePipeline

  Scopri come utilizzare una pipeline CI/CD per CodePipeline automatizzare i controlli di sicurezza preventivi e investigativi. Questo post spiega come utilizzare una pipeline per creare un semplice gruppo di sicurezza ed eseguire controlli di sicurezza durante le fasi di origine, test e produzione per migliorare il livello di sicurezza dei tuoi account. AWS

  Pubblicazione: marzo 2017

- Distribuzione continua su Amazon ECS utilizzando CodePipeline Amazon ECR e CodeBuild AWS CloudFormation

  Scopri come creare una pipeline di distribuzione continua verso Amazon Elastic Container Service (Amazon ECS). Le applicazioni vengono distribuite come contenitori Docker utilizzando CodePipeline Amazon ECR e. CodeBuild AWS CloudFormation

  - Scarica un AWS CloudFormation modello di esempio e le istruzioni per utilizzarlo per creare la tua pipeline di distribuzione continua dal repository ECS Reference Architecture: Continuous Deployment on. GitHub

  Pubblicazione: gennaio 2017

- Distribuzione continua per applicazioni serverless

  Scopri come utilizzare una raccolta di per Servizi AWS creare una pipeline di distribuzione continua per le tue applicazioni serverless. Utilizzerai il Serverless Application Model (SAM) per definire l'applicazione e le relative risorse e CodePipeline per orchestrare la distribuzione delle applicazioni.

  - Visualizza un'applicazione di esempio scritta in Go con il framework Gin e uno shim proxy Gateway API.

Pubblicazione: dicembre 2016

- Scalabilità delle implementazioni con e DevOps Dynatrace CodePipeline

  Scopri come utilizzare le soluzioni di monitoraggio Dynatrace per scalare le pipeline CodePipeline, analizzare automaticamente le esecuzioni dei test prima che il codice venga eseguito e mantenere tempi di consegna ottimali.

  Pubblicazione: novembre 2016

- Crea una pipeline per InUsing e AWS Elastic Beanstalk CodePipeline AWS CloudFormation CodeCommit

  Scopri come implementare la distribuzione continua in una CodePipeline pipeline per un'applicazione in. AWS Elastic Beanstalk Tutte AWS le risorse vengono fornite automaticamente tramite l'uso di un AWS CloudFormation modello. Questa procedura dettagliata include anche CodeCommit and AWS Identity and Access Management (IAM).

  Pubblicazione: maggio 2016

- Automatizza e attiva CodeCommit CodePipeline AWS CloudFormation

  AWS CloudFormation Utilizzalo per automatizzare l'approvvigionamento di AWS risorse per una pipeline di distribuzione continua che utilizza CodeCommit,, CodePipeline e. CodeDeploy AWS Identity and Access Management

  Pubblicazione: aprile 2016

- Crea una pipeline tra account in AWS CodePipeline

  Ulteriori informazioni su come automatizzare il provisioning dell'accesso tra account a pipeline in AWS CodePipeline utilizzando AWS Identity and Access Management. Include esempi in un AWS CloudFormation modello.

  Pubblicazione: marzo 2016

- Esplorazione ASP.NET Core Parte 2: distribuzione continua

  Scopri come creare un sistema di distribuzione continua completo per un'applicazione ASP.NET Core utilizzando CodeDeploy e AWS CodePipeline.

  Pubblicazione: marzo 2016

- Crea una pipeline utilizzando la console AWS CodePipeline

Scopri come utilizzare la AWS CodePipeline console per creare una pipeline in due fasi in una procedura dettagliata basata su. AWS CodePipeline Tutorial: creazione di una pipeline a quattro fasi

Pubblicazione: marzo 2016

- AWS CodePipeline Mocking Pipelines con AWS Lambda

  Scopri come richiamare una funzione Lambda che ti consente di visualizzare le azioni e le fasi di CodePipeline un processo di distribuzione del software durante la progettazione, prima che la pipeline diventi operativa. Durante la progettazione della struttura della pipeline, è possibile utilizzare la funzione Lambda per verificare se la pipeline verrà completata correttamente.

  Pubblicazione: febbraio 2016

- Esecuzione di funzioni in AWS Lambda Using CodePipeline AWS CloudFormation

  Scopri come creare uno AWS CloudFormation stack che fornisca tutte le AWS risorse utilizzate nell'attività Invoca una AWS Lambda funzione in una pipeline in CodePipeline della guida per l'utente.

  Pubblicazione: febbraio 2016

- Fornitura di azioni personalizzate CodePipeline in AWS CloudFormation

  Scopri come utilizzarlo AWS CloudFormation per fornire azioni personalizzate in CodePipeline.

  Pubblicazione: gennaio 2016

- Fornitura con CodePipeline AWS CloudFormation

  Scopri come fornire una pipeline di distribuzione continua di base in CodePipeline uso. AWS CloudFormation

  Pubblicazione: dicembre 2015

- Distribuzione da CodePipeline a AWS OpsWorks Utilizzo di un'azione personalizzata e AWS Lambda

  Scopri come configurare la tua pipeline e la AWS Lambda funzione da implementare. AWS OpsWorks CodePipeline

  Pubblicazione: luglio 2015

CodePipeline tutorial

Dopo aver completato i passaggi indicatiGuida introduttiva con CodePipeline, puoi provare uno dei AWS CodePipeline tutorial di questa guida per l'utente:

Desidero utilizzare la procedura guidata per creare una pipeline da utilizzare per distribui re un'applicazione di esempio da un bucket Amazon S3 a istanze Amazon EC2 che eseguono Amazon Linux. CodeDeploy Una volta utilizzata la procedura guidata per creare una pipeline a due fasi, voglio aggiungere una terza fase.	Per informazioni, consulta Tutorial: creazione di una semplice pipeline (bucket S3).
Voglio creare una pipeline in due fasi che utilizzi CodeDeploy per distribuire un'applic azione di esempio da un CodeCommit repositor y a un'istanza Amazon EC2 che esegue Amazon Linux.	Per informazioni, consulta Tutorial: crea una pipeline semplice (CodeCommitrepository).
Desidero aggiungere una fase di compilazi one per la pipeline a tre fasi che ho creato nel primo tutorial. La nuova fase utilizza Jenkins per compilare l'applicazione.	Per informazioni, consulta Tutorial: creazione di una pipeline a quattro fasi.
Voglio configurare una regola CloudWatch Events che invii notifiche ogni volta che ci sono modifiche allo stato di esecuzione della mia pipeline, fase o azione.	Per informazioni, consulta Tutorial: imposta una regola CloudWatch Events per ricevere notifiche e-mail per le modifiche allo stato della pipeline.
Voglio creare una pipeline con una GitHub fonte che crei e testa un'app Android con e. CodeBuild AWS Device Farm	Per informazioni, consulta Tutorial: crea una pipeline con cui creare e testare la tua app Android AWS Device Farm.
Voglio creare una pipeline con una fonte Amazon S3 con cui testare un'app iOS. AWS Device Farm	Per informazioni, consulta Tutorial: crea una pipeline con cui testare la tua app iOS AWS Device Farm.

Voglio creare una pipeline che distribuisca il mio modello di prodotto su Service Catalog.	Per informazioni, consulta Tutorial: crea una pipeline da distribuire su Service Catalog.
Voglio utilizzare modelli di esempio per creare una pipeline semplice (con Amazon S3 CodeCommit GitHub o una fonte) utilizzando AWS CloudFormation la console.	Per informazioni, consulta Tutorial: Creare una pipeline con AWS CloudFormation.
Voglio creare una pipeline in due fasi che utilizzi CodeDeploy Amazon ECS per la distribuzione blu/verde di un'immagine da un repository Amazon ECR a un cluster e servizio Amazon ECS.	Per informazioni, consulta Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy .
Desidero creare una pipeline che pubblichi in modo continuo la mia applicazione serverless su AWS Serverless Application Repository.	Per informazioni, consulta Tutorial: crea una pipeline che pubblichi la tua applicazione serverless su AWS Serverless Application Repository.

I seguenti tutorial in altre guide per l'utente forniscono indicazioni per l'integrazione di altri nelle tue pipeline: Servizi AWS

- Crea una pipeline che utilizzi nella Guida per l'utente CodeBuild AWS CodeBuild
- Utilizzo di CodePipeline with AWS OpsWorks Stacks nella Guida per AWS OpsWorks l'utente
- Distribuzione continua con CodePipeline guida AWS CloudFormation per l'utente
- Guida introduttiva a Elastic Beanstalk nella Developer AWS Elastic Beanstalk Guide
- Configura una pipeline di distribuzione continua utilizzando CodePipeline

Tutorial: usa i tag Git per avviare la tua pipeline

In questo tutorial, creerai una pipeline che si connette al tuo GitHub repository in cui l'azione source è configurata per il tipo di trigger dei tag Git. Quando viene creato un tag Git su un commit, viene avviata la pipeline. Questo esempio mostra come creare una pipeline che consenta il filtraggio dei tag in base alla sintassi del nome del tag. Per ulteriori informazioni sul filtraggio con pattern a glob, consulta. Lavorare con i modelli a globo nella sintassi

Questo tutorial si collega a GitHub tramite il tipo di `CodeStarSourceConnection` azione.

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hong Kong), Africa (Città del Capo), Medio Oriente (Bahrein) o Europa (Zurigo). Per fare riferimento ad altre azioni disponibili, consultaIntegrazioni di prodotti e servizi con CodePipeline. Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Argomenti

Prerequisiti

Prima di iniziare è necessario:

- Crea un GitHub repository con il tuo GitHub account.
- Tieni a portata di mano GitHub le tue credenziali. Quando utilizzi il AWS Management Console per configurare una connessione, ti viene chiesto di accedere con GitHub le tue credenziali.

Passaggio 1: apri CloudShell e clona il tuo repository

Puoi usare un'interfaccia a riga di comando per clonare il tuo repository, fare commit e aggiungere tag. Questo tutorial avvia un' CloudShell istanza per l'interfaccia a riga di comando.

1. Accedi alla AWS Management Console.
2. Nella barra di navigazione in alto, scegli l' AWS icona. La pagina principale dei AWS Management Console display.

3. Nella barra di navigazione in alto, scegli l' AWS CloudShell icona. CloudShell si apre. Attendi che l' CloudShell ambiente venga creato.

   > **Note**
   >
   > Se non vedi l' CloudShell icona, assicurati di trovarti in una regione supportata da CloudShell. Questo tutorial presuppone che ti trovi nella regione degli Stati Uniti occidentali (Oregon).

4. In GitHub, accedi al tuo repository. Scegli Codice, quindi scegli HTTPS. Copia il percorso. L'indirizzo per clonare il repository Git viene copiato negli Appunti.
5. Esegui il comando seguente per clonare il repository.

   ```
   git clone https://github.com/<account>/MyGitHubRepo.git
   ```

6. Inserisci il tuo GitHub account `Username` e `Password` quando richiesto. Per l'`Password`dimmissione, è necessario utilizzare un token creato dall'utente anziché la password dell'account.

Passaggio 2: creare una pipeline da attivare sui tag Git

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase sorgente con una connessione al tuo GitHub repository e all'azione.
- Una fase di compilazione con un'azione di AWS CodeBuild compilazione.

Per creare una pipeline con la procedura guidata

1. Accedi alla CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyGitHubTagsPipeline`**.
4. Nel tipo Pipeline, mantieni la selezione predefinita su V2. I tipi di tubazioni differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Service Role (Ruolo del servizio), scegliere New service role (Nuovo ruolo del servizio).

> **Note**
>
> Se invece scegli di utilizzare il tuo ruolo di CodePipeline servizio esistente, assicurati di aver aggiunto l'autorizzazione `codestar-connections:UseConnection` IAM alla tua politica del ruolo di servizio. Per istruzioni sul ruolo CodePipeline di servizio, consulta Aggiungere autorizzazioni al ruolo CodePipeline di servizio.

6. In Impostazioni avanzate non modificare le impostazioni predefinite. In Artifact store (Archivio artefatti), seleziona Default location (Posizione predefinita) per utilizzare l'archivio artefatti predefinito, ad esempio il bucket Amazon S3 dedicato agli artefatti designato come predefinito, per la pipeline nella regione selezionata.

   > **Note**
   >
   > Non si tratta del bucket di origine per il codice sorgente, ma dell'archivio artefatti per la pipeline. È richiesto un archivio artefatti separato, ad esempio un bucket S3, per ogni pipeline.

   Seleziona Next (Successivo).

7. Nella Fase 2: Aggiungi una fase di origine, aggiungi una fase di origine:

   a. In Provider di origine, scegli GitHub (versione 2).

   b. In Connessione, scegli una connessione esistente o creane una nuova. Per creare o gestire una connessione per l'azione GitHub sorgente, consultaGitHub connessioni.

   c. In Nome archivio, scegli il nome del tuo GitHub repository.

   d. In Pipeline trigger, scegli Tag Git.

      Nel campo Includi, inserisci`release*`.

      Nel ramo predefinito, scegli il ramo che desideri specificare quando la pipeline viene avviata manualmente o con un evento di origine che non sia un tag Git. Se l'origine della modifica non è il trigger o se l'esecuzione di una pipeline è stata avviata manualmente, la modifica utilizzata sarà il commit HEAD del ramo predefinito.

> **Important**
>
> Le pipeline che iniziano con un tipo di trigger di tag Git verranno configurate per gli eventi WebHookV2 e non utilizzeranno l'evento Webhook (rilevamento delle modifiche su tutti gli eventi push) per avviare la pipeline.

Seleziona Next (Successivo).

8. In Add build stage (Aggiungi fase di compilazione), aggiungi una fase di compilazione:

   a. In Build provider (Provider compilazione), scegli AWS CodeBuild. Consenti a Region (Regione) di preimpostarsi sulla regione della pipeline.

   b. Seleziona Crea progetto.

   c. In Project name (Nome progetto) immettere un nome per questo progetto di compilazione.

   d. In Environment image (Immagine ambiente), scegli Managed image (Immagine gestita). In Operating system (Sistema operativo), seleziona Ubuntu.

   e. In Runtime, seleziona Standard. Per Immagine, scegli aws/codebuild/standard:5.0.

   f. Per Service Role (Ruolo del servizio), scegli New service role (Nuovo ruolo del servizio).

   > **Note**
   >
   > Annota il nome del tuo ruolo di servizio. CodeBuild Avrai bisogno del nome del ruolo per l'ultimo passaggio di questo tutorial.

   g. In Buildspec, per Build specifications (Specifiche di compilazione), scegli Insert build commands (Inserisci comandi di compilazione). Scegli Passa all'editor e incolla quanto segue nei comandi di compilazione.

```
version: 0.2
#env:
  #variables:
     # key: "value"
     # key: "value"
  #parameter-store:
     # key: "value"
     # key: "value"
  #git-credential-helper: yes
```

```
phases:
  install:
    #If you use the Ubuntu standard image 2.0 or later, you must specify
 runtime-versions.
    #If you specify runtime-versions and use an image other than Ubuntu
 standard image 2.0, the build fails.
    runtime-versions:
      nodejs: 12
    #commands:
      # - command
      # - command
  #pre_build:
    #commands:
      # - command
      # - command
  build:
    commands:
      -
  #post_build:
    #commands:
      # - command
      # - command
artifacts:
  files:
     - '*'
    # - location
  name: $(date +%Y-%m-%d)
  #discard-paths: yes
  #base-directory: location
#cache:
  #paths:
    # - paths
```

h. Scegli Continua a CodePipeline. Questo ritorna alla CodePipeline console e crea un CodeBuild progetto che utilizza i comandi di compilazione per la configurazione. Il progetto di compilazione utilizza un ruolo di servizio per gestire le Servizio AWS autorizzazioni. Questa operazione potrebbe richiedere un paio di minuti.

i. Seleziona Next (Successivo).

9. Nella pagina Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione), scegli Skip deploy stage (Ignora fase di distribuzione), quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Next (Successivo).

10. Nella Step 5: Review (Fase 5: revisione), scegliere Create pipeline (Crea pipeline).

Passaggio 3: tagga i tuoi commit per il rilascio

Dopo aver creato la pipeline e specificato i tag Git, puoi taggare i commit nel tuo GitHub repository. In questi passaggi, taggherai un commit con il tag. `release-1` Ogni commit in un repository Git deve avere un tag Git univoco. Quando scegli il commit e lo tagghi, ciò ti consente di incorporare le modifiche provenienti da diversi rami nella distribuzione della pipeline. Nota che il nome del tag release non si applica al concetto di release in GitHub.

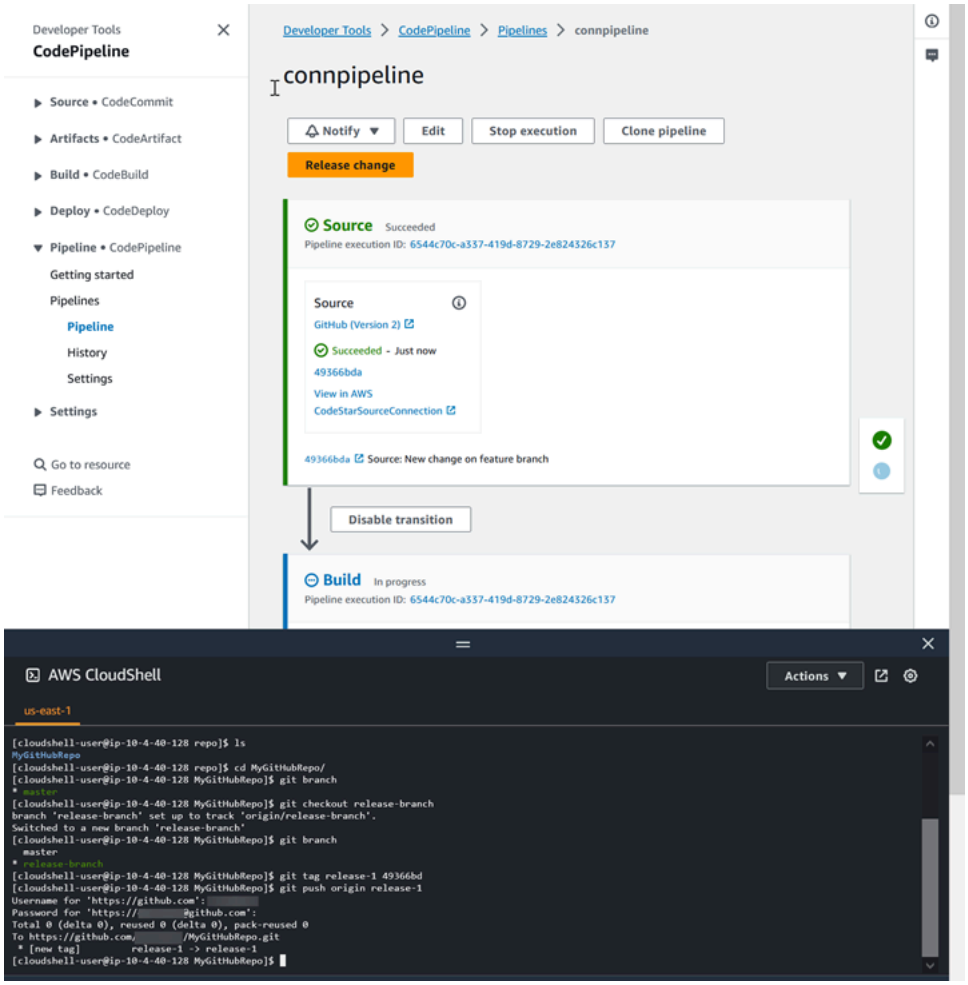
1. Fai riferimento agli ID di commit copiati che desideri taggare. Per visualizzare i commit in ogni ramo, nel CloudShell terminale, inserisci il seguente comando per acquisire gli ID di commit che desideri taggare:

   ```
   git log
   ```

2. Nel CloudShell terminale, inserisci il comando per etichettare il tuo commit e invialo a origin. Dopo aver taggato il tuo commit, usi il comando git push per spingere il tag all'origine. Nell'esempio seguente, inserisci il comando seguente per utilizzare il `release-1` tag per il secondo commit con ID49366bd. Questo tag verrà filtrato dal filtro dei `release*` tag pipeline e avvierà la pipeline.

   ```
   git tag release-1 49366bd
   ```

   ```
   git push origin release-1
   ```



Passaggio 4: Rilascia le modifiche e visualizza i log

1. Dopo che la pipeline è stata eseguita correttamente, nella fase di compilazione corretta, scegli Visualizza registro.

   In Log, visualizza l'output della CodeBuild build. I comandi restituiscono il valore della variabile immessa.

2. Nella pagina Cronologia, visualizza la colonna Trigger. Visualizza il tipo di trigger GitTag : release-1.

Tutorial: filtra i nomi delle filiali per le richieste pull per avviare la tua pipeline

In questo tutorial, creerai una pipeline che si collega al tuo repository GitHub .com in cui l'azione source è configurata per avviare la pipeline con una configurazione di trigger che filtra le richieste pull. Quando si verifica uno specifico evento di pull request per un ramo specifico, viene avviata la pipeline. Questo esempio mostra come creare una pipeline che consenta il filtraggio dei nomi delle filiali. Per ulteriori informazioni sull'utilizzo dei trigger, vedere. Filtraggio dei trigger nella pipeline JSON (CLI) Per ulteriori informazioni sul filtraggio con pattern regex in formato glob, consulta. Lavorare con i modelli a globo nella sintassi

Questo tutorial si collega a GitHub .com tramite il tipo di azione`CodeStarSourceConnection`.

Argomenti

- Prerequisiti
- Passaggio 1: crea una pipeline da avviare su richiesta pull per rami specifici
- Passaggio 2: crea e unisci una richiesta pull in GitHub .com per avviare le esecuzioni della pipeline

Prerequisiti

Prima di iniziare è necessario:

- Crea un GitHub repository.com con il tuo account GitHub .com.
- Tieni a portata di mano GitHub le tue credenziali. Quando utilizzi il AWS Management Console per configurare una connessione, ti viene chiesto di accedere con GitHub le tue credenziali.

Passaggio 1: crea una pipeline da avviare su richiesta pull per rami specifici

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con una connessione al repository e all'azione GitHub .com.
- Una fase di compilazione con un'azione di AWS CodeBuild compilazione.

Per creare una pipeline con la procedura guidata

1. Accedi alla CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyFilterBranchesPipeline`**.
4. Nel tipo Pipeline, mantieni la selezione predefinita su V2. I tipi di tubazioni differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Service Role (Ruolo del servizio), scegliere New service role (Nuovo ruolo del servizio).

   > **Note**
   >
   > Se invece scegli di utilizzare il tuo ruolo di CodePipeline servizio esistente, assicurati di aver aggiunto l'autorizzazione `codeconnections:UseConnection` IAM alla tua politica del ruolo di servizio. Per istruzioni sul ruolo CodePipeline di servizio, consulta Aggiungere autorizzazioni al ruolo CodePipeline di servizio.

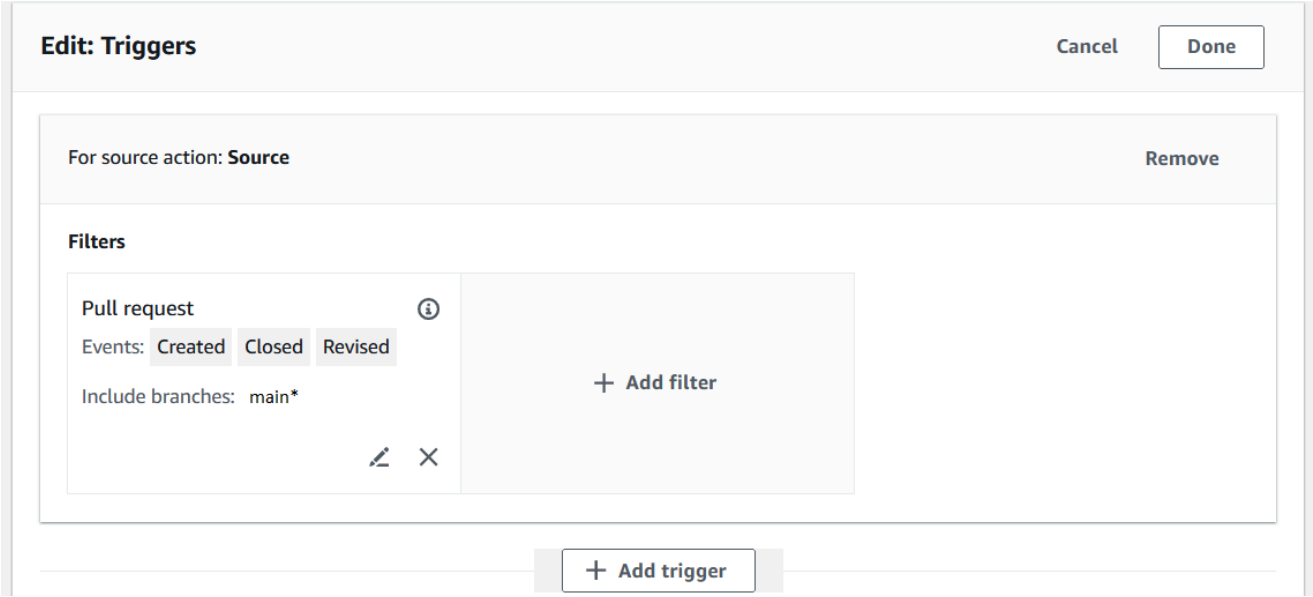
6. In Impostazioni avanzate non modificare le impostazioni predefinite. In Artifact store (Archivio artefatti), seleziona Default location (Posizione predefinita) per utilizzare l'archivio artefatti predefinito, ad esempio il bucket Amazon S3 dedicato agli artefatti designato come predefinito, per la pipeline nella regione selezionata.

   > **Note**
   >
   > Non si tratta del bucket di origine per il codice sorgente, ma dell'archivio artefatti per la pipeline. È richiesto un archivio artefatti separato, ad esempio un bucket S3, per ogni pipeline.

   Seleziona Next (Successivo).
7. Nella Fase 2: Aggiungi una fase di origine, aggiungi una fase di origine:
   a. In Provider di origine, scegli GitHub (versione 2).
   b. In Connessione, scegli una connessione esistente o creane una nuova. Per creare o gestire una connessione per l'azione GitHub sorgente, consultaGitHub connessioni.
   c. In Nome repository, scegli il nome del tuo repository GitHub .com.
   d. In Tipo di trigger, scegli Specificare filtro.

In Tipo di evento, scegli Pull request. Seleziona tutti gli eventi sotto richiesta pull in modo che l'evento si verifichi per le richieste pull create, aggiornate o chiuse.

In Branches, nel campo Includi, inserisci`main*`.



> **Important**
>
> Le pipeline che iniziano con questo tipo di trigger verranno configurate per gli eventi WebHookV2 e non utilizzeranno l'evento Webhook (rilevamento delle modifiche su tutti gli eventi push) per avviare la pipeline.

Seleziona Successivo.

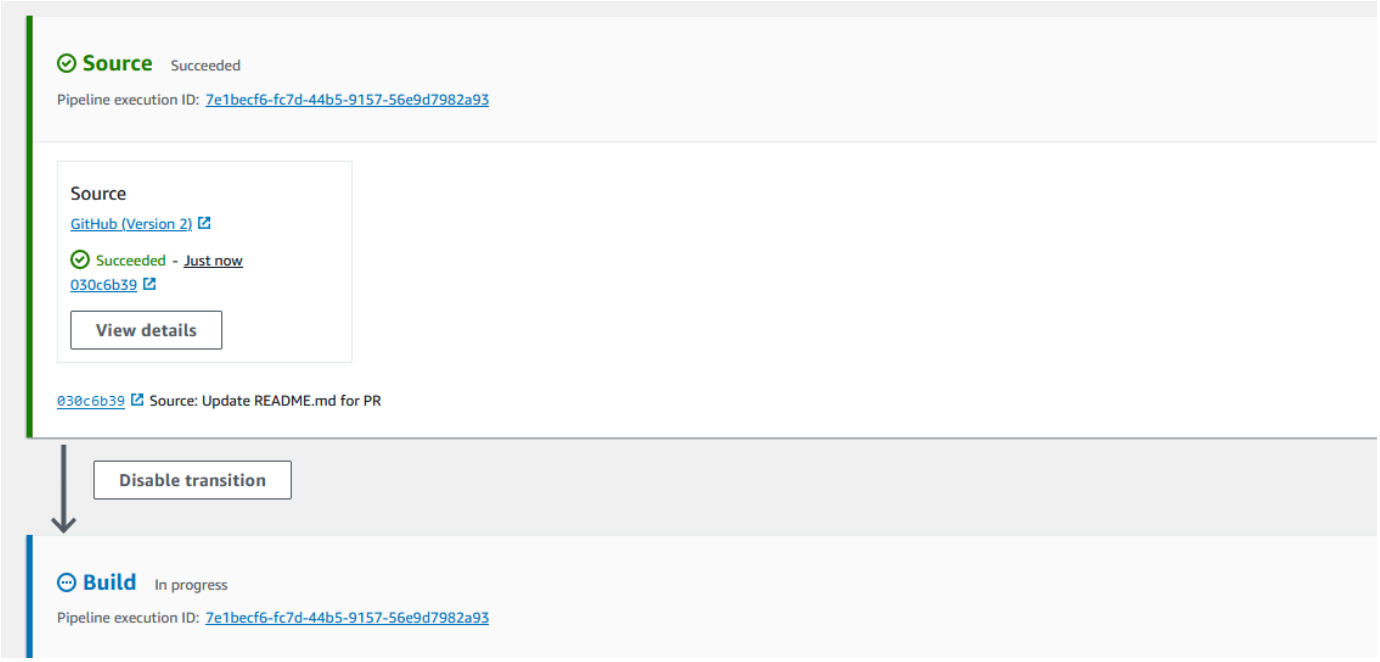
8. Nella fase Aggiungi compilazione, in Build provider, scegli. AWS CodeBuild Consenti a Region (Regione) di preimpostarsi sulla regione della pipeline. Scegli o crea il progetto di compilazione come indicato in[Tutorial: usa i tag Git per avviare la tua pipeline](). Questa azione verrà utilizzata in questo tutorial solo come seconda fase necessaria per creare la pipeline.
9. Nella pagina Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione), scegli Skip deploy stage (Ignora fase di distribuzione), quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Next (Successivo).
10. Nella Step 5: Review (Fase 5: revisione), scegliere Create pipeline (Crea pipeline).

Passaggio 2: crea e unisci una richiesta pull in GitHub .com per avviare le esecuzioni della pipeline

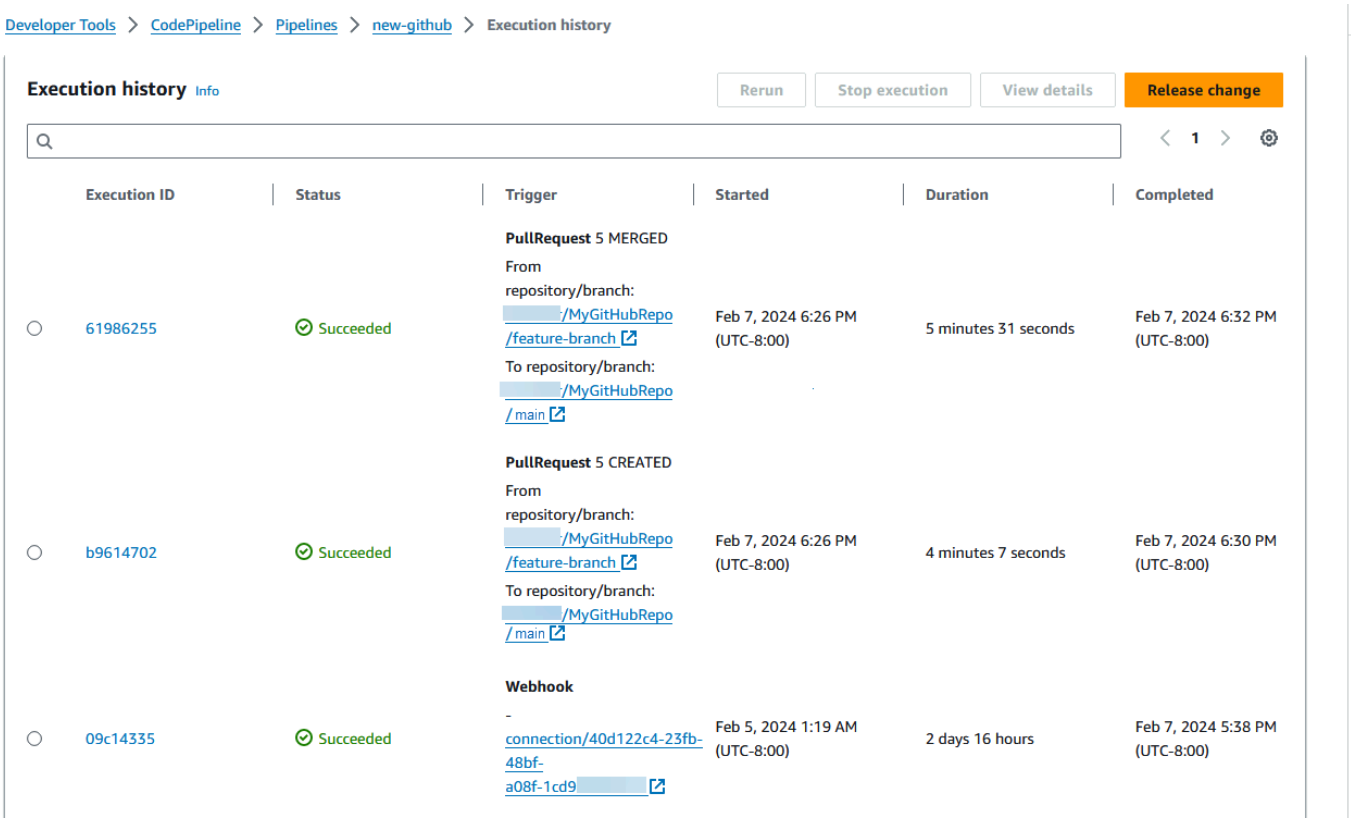
In questa sezione, crei e unisci una pull request. Questo avvia la pipeline, con un'esecuzione per la pull request aperta e un'esecuzione per la pull request chiusa.

Per creare una pull request e avviare la pipeline

1. In GitHub .com, crea una richiesta pull apportando una modifica a README.md su un ramo di funzionalità e inviando una richiesta pull al ramo. `main` Conferma la modifica con un messaggio del tipo. `Update README.md for PR`
2. La pipeline inizia con la revisione del codice sorgente che mostra il messaggio Source per la pull request come Update README.md for PR.



3. Scegliere History (Cronologia). Nella cronologia di esecuzione della pipeline, visualizza gli eventi di stato delle richieste pull CREATED e MERGED che hanno avviato le esecuzioni della pipeline.



Tutorial: utilizzare le variabili a livello di pipeline

In questo tutorial, creerai una pipeline in cui aggiungi una variabile a livello di pipeline ed eseguirai un'azione di CodeBuild compilazione che restituisce il valore della variabile.

Argomenti

Prerequisiti

Prima di iniziare è necessario:

- Crea un repository. CodeCommit
- Aggiungi un file.txt al repository.

Passaggio 1: crea la tua pipeline e crea il progetto

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con una connessione al tuo CodeCommit repository.
- Una fase di compilazione con un'azione di AWS CodeBuild compilazione.

Per creare una pipeline con la procedura guidata

1. Accedi alla CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyVariablesPipeline`**.
4. Nel tipo Pipeline, mantieni la selezione predefinita su V2. I tipi di tubazioni differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Service Role (Ruolo del servizio), scegliere New service role (Nuovo ruolo del servizio).

   > **Note**
   >
   > Se invece scegli di utilizzare il tuo ruolo di CodePipeline servizio esistente, assicurati di aver aggiunto l'autorizzazione `codeconnections:UseConnection` IAM alla tua politica del ruolo di servizio. Per istruzioni sul ruolo CodePipeline di servizio, consulta Aggiungere autorizzazioni al ruolo CodePipeline di servizio.

6. In Variabili, scegli Aggiungi variabile. In Nome, inserisci `timeout`. In Predefinito, inserisci 1000. Nella descrizione, immettere la seguente descrizione:**`Timeout`**.

   Questo creerà una variabile in cui è possibile dichiarare il valore all'avvio dell'esecuzione della pipeline. I nomi delle variabili devono corrispondere `[A-Za-z0-9@\-_]+` e possono essere qualsiasi cosa tranne una stringa vuota.
7. In Impostazioni avanzate non modificare le impostazioni predefinite. In Artifact store (Archivio artefatti), seleziona Default location (Posizione predefinita) per utilizzare l'archivio artefatti predefinito, ad esempio il bucket Amazon S3 dedicato agli artefatti designato come predefinito, per la pipeline nella regione selezionata.

> **Note**
>
> Non si tratta del bucket di origine per il codice sorgente, ma dell'archivio artefatti per la pipeline. È richiesto un archivio artefatti separato, ad esempio un bucket S3, per ogni pipeline.

Seleziona Next (Successivo).

8. Nella Fase 2: Aggiungi una fase di origine, aggiungi una fase di origine:

   a. In Source provider (Provider origine), scegliere AWS CodeCommit.

   b. In Nome del repository e Nome del ramo, scegli il tuo repository e il tuo ramo.

   Seleziona Next (Successivo).

9. In Add build stage (Aggiungi fase di compilazione), aggiungi una fase di compilazione:

   a. In Build provider (Provider compilazione), scegli AWS CodeBuild. Consenti a Region (Regione) di preimpostarsi sulla regione della pipeline.

   b. Seleziona Crea progetto.

   c. In Project name (Nome progetto) immettere un nome per questo progetto di compilazione.

   d. In Environment image (Immagine ambiente), scegli Managed image (Immagine gestita). In Operating system (Sistema operativo), seleziona Ubuntu.

   e. In Runtime, seleziona Standard. Per Immagine, scegli aws/codebuild/standard:5.0.

   f. Per Service Role (Ruolo del servizio), scegli New service role (Nuovo ruolo del servizio).

   > **Note**
   >
   > Annota il nome del tuo ruolo di CodeBuild servizio. Avrai bisogno del nome del ruolo per l'ultimo passaggio di questo tutorial.

   g. In Buildspec, per Build specifications (Specifiche di compilazione), scegli Insert build commands (Inserisci comandi di compilazione). Scegli Passa all'editor e incolla quanto segue nei comandi di compilazione. In buildspec, la variabile customer `$CUSTOM_VAR1` verrà utilizzata per generare la variabile pipeline nel registro di compilazione. Creerai la variabile `$CUSTOM_VAR1` di output come variabile di ambiente nel passaggio successivo.

```
version: 0.2
#env:
  #variables:
     # key: "value"
     # key: "value"
  #parameter-store:
     # key: "value"
     # key: "value"
  #git-credential-helper: yes
phases:
  install:
    #If you use the Ubuntu standard image 2.0 or later, you must specify
 runtime-versions.
    #If you specify runtime-versions and use an image other than Ubuntu
 standard image 2.0, the build fails.
    runtime-versions:
      nodejs: 12
    #commands:
      # - command
      # - command
  #pre_build:
    #commands:
      # - command
      # - command
  build:
    commands:
      - echo $CUSTOM_VAR1
  #post_build:
    #commands:
      # - command
      # - command
artifacts:
  files:
     - '*'
    # - location
  name: $(date +%Y-%m-%d)
  #discard-paths: yes
  #base-directory: location
#cache:
  #paths:
    # - paths
```

h. Scegli Continua a CodePipeline. Questo ritorna alla CodePipeline console e crea un CodeBuild progetto che utilizza i comandi di compilazione per la configurazione. Il progetto di compilazione utilizza un ruolo di servizio per gestire le Servizio AWS autorizzazioni. Questa operazione potrebbe richiedere un paio di minuti.

i. In Variabili di ambiente - facoltativo, per creare una variabile di ambiente come variabile di input per l'azione di compilazione che verrà risolta dalla variabile a livello di pipeline, scegli Aggiungi variabile di ambiente. Questo creerà la variabile specificata in buildspec as. `$CUSTOM_VAR1` In Nome, inserisci `CUSTOM_VAR1`. In Valore, immetti `#{variables.timeout}`. In Tipo, scegliete. `Plaintext`

   Il `#{variables.timeout}` valore della variabile di ambiente si basa sullo spazio dei nomi della variabile a livello di pipeline `variables` e sulla variabile a livello di pipeline creata per la pipeline nel passaggio 5. `timeout`

j. Seleziona Next (Successivo).

10. Nella pagina Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione), scegli Skip deploy stage (Ignora fase di distribuzione), quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Next (Successivo).
11. Nella Step 5: Review (Fase 5: revisione), scegliere Create pipeline (Crea pipeline).

Fase 2: Rilasciare le modifiche e visualizzare i log

1. Dopo che la pipeline è stata eseguita correttamente, nella fase di compilazione corretta, scegli Visualizza dettagli.

   Nella pagina dei dettagli, scegli la scheda Registri. Visualizza l'output della CodeBuild build. I comandi restituiscono il valore della variabile inserita.

2. Nel navigatore a sinistra, scegli Cronologia.

   Scegli l'esecuzione recente, quindi scegli la scheda Variabili. Visualizzate il valore risolto per la variabile pipeline.

Tutorial: creazione di una semplice pipeline (bucket S3)

Il modo più semplice per creare una pipeline consiste nell'utilizzare la procedura guidata Crea pipeline nella console. AWS CodePipeline

In questo tutorial, creerai una pipeline a due fasi che utilizza un bucket S3 con versione diversa e rilascerai un'applicazione di esempio. CodeDeploy

> **Note**
>
> Se Amazon S3 è il fornitore di origine per la tua pipeline, puoi comprimere il file o i file sorgente in un unico .zip e caricare il file.zip nel tuo bucket di origine. È inoltre possibile caricare un singolo file decompresso; tuttavia, le operazioni a valle che si aspettano un file con estensione .zip avranno esito negativo.

Una volta creata questa pipeline semplice, puoi aggiungere un'altra fase e quindi disabilitare e abilitare la transizione tra fasi.

> **Important**
>
> Molte delle azioni che aggiungi alla pipeline in questa procedura coinvolgono AWS risorse che devi creare prima di creare la pipeline. AWS le risorse per le tue azioni di origine devono sempre essere create nella stessa AWS regione in cui crei la pipeline. Ad esempio, se crei la pipeline nella regione Stati Uniti orientali (Ohio), il tuo CodeCommit repository deve trovarsi nella regione Stati Uniti orientali (Ohio).
>
> Puoi aggiungere azioni interregionali quando crei la pipeline. AWS le risorse per le azioni interregionali devono trovarsi nella stessa AWS regione in cui intendi eseguire l'azione. Per ulteriori informazioni, consulta Aggiungere un'azione interregionale in CodePipeline.

Prima di iniziare, occorre completare i prerequisiti in Guida introduttiva con CodePipeline.

Argomenti

Fase 1: creazione di un bucket S3 per l'applicazione

Puoi archiviare i file di origine o le applicazioni in qualsiasi percorso con versioni multiple. In questo tutorial, crei un bucket S3 per i file dell'applicazione di esempio e abiliti il controllo delle versioni su quel bucket. Dopo aver abilitato la funzione Versioni multiple, puoi copiare le applicazioni di esempio nel bucket.

Per creare un bucket S3

1. Accedi alla console all'indirizzo. AWS Management Console Apri la console S3.
2. Seleziona Crea bucket.
3. Per Bucket Name (Nome bucket), immettere un nome per il bucket, ad esempio **`awscodepipeline-demobucket-example-date`**.

   > **Note**
   >
   > Poiché tutti i nomi dei bucket in Amazon S3 devono essere univoci, usane uno personalizzato, non il nome mostrato nell'esempio. È possibile modificare il nome dell'esempio semplicemente aggiungendo la data. Prendere nota di questo nome perché sarà necessario nel resto di questo tutorial.

   In Regione, scegli la regione in cui intendi creare la pipeline, ad esempio Stati Uniti occidentali (Oregon), quindi scegli Crea bucket.
4. Dopo aver creato il bucket, viene visualizzato un banner di successo. Scegliere Go to bucket details (Vai ai dettagli del bucket).
5. Nella scheda Properties (Proprietà) scegliere Versioning (Funzione Versioni multiple). Scegliere Enable versioning (Abilita funzione Versioni multiple), quindi scegliere Save (Salva).

   Quando il controllo delle versioni è abilitato, Amazon S3 salva ogni versione di ogni oggetto nel bucket.
6. Nella scheda Permissions (Autorizzazioni), lasciare le impostazioni predefinite. Per ulteriori informazioni sul bucket S3 e le autorizzazioni oggetto, consulta Specificare le autorizzazioni in una policy.
7. Quindi, scaricare un esempio e salvarlo in una cartella o directory sul computer locale.

a. Scegliere una delle seguenti opzioni. Scegliere `SampleApp_Windows.zip` se si desidera seguire la procedura descritta in questo tutorial per le istanze di Windows Server.

   - Se desideri eseguire la distribuzione su istanze Amazon Linux utilizzando CodeDeploy, scarica l'applicazione di esempio qui: SampleApp _Linux.zip.
   - Se desideri eseguire la distribuzione su istanze di Windows Server utilizzando CodeDeploy, scarica l'applicazione di esempio qui: _Windows.zip. SampleApp

   L'applicazione di esempio contiene i seguenti file con cui eseguire la distribuzione: CodeDeploy

   - `appspec.yml`— Il file delle specifiche dell'applicazione (AppSpecfile) è un file in formato YAML utilizzato da CodeDeploy per gestire una distribuzione. Per ulteriori informazioni sul AppSpec file, vedere CodeDeploy AppSpec File reference nella Guida per l'utente.AWS CodeDeploy
   - `index.html`— Il file indice contiene la home page dell'applicazione di esempio distribuita.
   - `LICENSE.txt`— Il file di licenza contiene informazioni sulla licenza per l'applicazione di esempio.
   - File per script: l'applicazione di esempio utilizza script per scrivere file di testo in una posizione sull'istanza. Un file viene scritto per ciascuno dei diversi eventi del ciclo di vita della CodeDeploy distribuzione nel modo seguente:
     - `scripts`Cartella (solo esempio Linux): la cartella contiene i seguenti script di shell per installare le dipendenze e avviare e arrestare l'applicazione di esempio per la distribuzione automatizzata:`install_dependencies`, e. `start_server` `stop_server`
     - (Solo esempio per Windows)`before-install.bat`: si tratta di uno script batch per l'evento del ciclo di vita della `BeforeInstall` distribuzione, che verrà eseguito per rimuovere i vecchi file scritti durante le distribuzioni precedenti di questo esempio e creare una posizione sull'istanza in cui scrivere i nuovi file.

b. Scarica il file compresso. Non decomprimere il file.

8. Nella console Amazon S3, per il tuo bucket, carica il file:

   a. Scegli Carica.

   b. Trascina il file oppure scegli Add files (Aggiungi file) e individua il file.

c. Scegli Carica.

Fase 2: creare istanze Amazon EC2 Windows e installare l'agente CodeDeploy

> **Note**
>
> Questo tutorial fornisce passaggi di esempio per creare istanze Windows di Amazon EC2. Per esempi di passaggi per creare istanze Amazon EC2 Linux, consulta. Fase 3: creare un'istanza Amazon EC2 Linux e installare l'agente CodeDeploy Quando viene richiesto il numero di istanze da creare, specificare 2 istanze.

In questo passaggio, creerai le istanze Windows Server Amazon EC2 su cui distribuirai un'applicazione di esempio. Come parte di questo processo, crei un ruolo dell'istanza con politiche che consentono l'installazione e la gestione dell' CodeDeploy agente sulle istanze. L' CodeDeploy agente è un pacchetto software che consente di utilizzare un'istanza nelle CodeDeploy distribuzioni. È inoltre possibile allegare politiche che consentono all'istanza di recuperare i file utilizzati dall' CodeDeploy agente per distribuire l'applicazione e di consentire la gestione dell'istanza da parte di SSM.

Per creare un ruolo dell'istanza

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/).
2. Dal pannello di controllo della console, scegli Roles (Ruoli).
3. Scegli Crea ruolo.
4. In Seleziona il tipo di entità affidabile, seleziona Servizio AWS. In Choose a use case (Scegliere un caso d'uso), selezionare EC2, quindi scegliere Next: Permissions (Successivo: autorizzazioni).
5. Cerca e seleziona la politica denominata **`AmazonEC2RoleforAWSCodeDeploy`**.
6. Cerca e seleziona la politica denominata **`AmazonSSMManagedInstanceCore`**. Scegliere Next: Tags (Successivo: Tag).
7. Scegliere Next:Review (Successivo: Rivedi). Immettere un nome per il ruolo (ad esempio **`EC2InstanceRole`**).

> **Note**
>
> Prendere nota del nome del ruolo per la fase successiva. È possibile scegliere questo ruolo quando si crea l'istanza.

Scegli Crea ruolo.

Per avviare istanze

1. Apri la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.
2. Dalla barra di navigazione laterale, scegli Istanze e seleziona Avvia istanze nella parte superiore della pagina.
3. In Nome e tag, in Nome, inserisci. **`MyCodePipelineDemo`** Questo assegna alle istanze un tag Key di **`Name`** e un tag Value di. **`MyCodePipelineDemo`** Successivamente, si crea un' CodeDeploy applicazione che distribuisce l'applicazione di esempio alle istanze. CodeDeployseleziona le istanze da distribuire in base ai tag.
4. In Immagini dell'applicazione e del sistema operativo (Amazon Machine Image), scegli l'opzione Windows. (Questa AMI è descritta come Microsoft Windows Server 2019 Base ed è etichettata come «Idonea per il livello gratuito» e può essere trovata in Quick Start.)
5. In Tipo di istanza, scegli il `t2.micro` tipo idoneo al piano gratuito come configurazione hardware per la tua istanza.
6. In Coppia di chiavi (login), scegli una coppia di chiavi o creane una.

   Puoi anche scegliere Proceed without a key pair.

   > **Note**
   >
   > Ai fini di questo tutorial, è possibile procedere senza una coppia di chiavi. Per utilizzare SSH per connettersi alle istanze, creare o utilizzare una coppia di chiavi.

7. In Impostazioni di rete, procedi come segue.

   In Assegna automaticamente un IP pubblico, assicurati che lo stato sia Abilita.

- Accanto a Assign a security group (Assegna un gruppo di sicurezza), scegliere Create a new security group (Crea un nuovo gruppo di sicurezza).
- Nella riga SSH, in Tipo di sorgente, scegli Il mio IP.
- Scegli Aggiungi gruppo di sicurezza, scegli HTTP, quindi in Tipo di origine, scegli Il mio IP.

8. Espandi Advanced details (Dettagli avanzati). Nel profilo dell'istanza IAM, scegli il ruolo IAM che hai creato nella procedura precedente (ad esempio,**`EC2InstanceRole`**).
9. In Riepilogo, in Numero di istanze, inserisci.. 2
10. Scegliere Launch Instance (Avvia istanza).
11. Scegli View Instances (Visualizza istanze) per chiudere la pagina di conferma e tornare alla console.
12. È possibile visualizzare lo stato dell'avvio nella pagina Instances (Istanze). Quando avvii un'istanza, il suo stato iniziale è `pending`. Una volta avviata l'istanza, il suo stato passa a `running` e riceve un nome DNS pubblico. Se la colonna Public DNS (DNS pubblico) non è visualizzata, scegliere l'icona Show/Hide (Mostra/Nascondi) quindi selezionare Public DNS (DNS pubblico).
13. Possono essere necessari alcuni minuti prima che sia possibile connettersi all'istanza. Verifica che l'istanza abbia superato i controlli dello stato. Queste informazioni sono disponibili nella colonna Status Checks (Verifiche di stato).

Passaggio 3: Creare un'applicazione in CodeDeploy

In CodeDeploy, un'applicazione è un identificatore, sotto forma di nome, per il codice che si desidera distribuire. CodeDeploy utilizza questo nome per garantire che durante una distribuzione venga fatto riferimento alla combinazione corretta di revisione, configurazione di distribuzione e gruppo di distribuzione. Il nome dell' CodeDeploy applicazione da creare in questo passaggio viene selezionato quando si crea la pipeline più avanti in questo tutorial.

Per prima cosa crei un ruolo di servizio CodeDeploy da utilizzare. Se è già stato creato un ruolo di servizio, non è necessario crearne un altro.

Per creare un ruolo CodeDeploy di servizio

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/).
2. Dal pannello di controllo della console, scegli Roles (Ruoli).

3. Scegli Crea ruolo.
4. In Seleziona un'entità affidabile, scegli Servizio AWS. In Use case (Caso d'uso), scegli CodeDeploy. Scegli CodeDeploytra le opzioni elencate. Seleziona Successivo. La policy gestita `AWSCodeDeployRole` è già collegata al ruolo.
5. Seleziona Successivo.
6. Immetti un nome per il ruolo, ad esempio **`CodeDeployRole`**, quindi seleziona Create role (Crea ruolo).

Per creare un'applicazione in CodeDeploy

1. Apri la CodeDeploy console all'indirizzo https://console.aws.amazon.com/codedeploy.
2. Se la pagina Applicazioni non viene visualizzata, nel AWS CodeDeploy menu, scegli Applicazioni.
3. Scegli Crea applicazione.
4. In Application name (Nome applicazione), immettere `MyDemoApplication`.
5. In Compute Platform (Piattaforma di calcolo), scegliere EC2/On-premises (EC2/Locale).
6. Scegli Crea applicazione.

Per creare un gruppo di distribuzione in CodeDeploy

1. Nella pagina in cui è visualizzata l'applicazione, scegliere Create deployment group (Crea gruppo di distribuzione).
2. In Deployment group name (Nome del gruppo di distribuzione), immettere **`MyDemoDeploymentGroup`**.
3. In Ruolo di servizio, scegli il ruolo di servizio che hai creato in precedenza. È necessario utilizzare un ruolo di servizio che AWS CodeDeploy garantisca almeno la fiducia e le autorizzazioni descritte in Creare un ruolo di servizio per. CodeDeploy Per ottenere l'ARN del ruolo di servizio, consulta Ottenere l'ARN del ruolo di servizio (console).
4. In Deployment type (Tipo di distribuzione), scegliere In-place (In loco).
5. In Environment configuration (Configurazione dell'ambiente), scegliere Amazon EC2 Instances (Istanze Amazon EC2). Scegliere Name (Nome) nel campo Key (Chiave) e nel campo Value (Valore), immettere **`MyCodePipelineDemo`**.

> **Important**
>
> Scegliere qui lo stesso valore per la chiave Name (Nome) che è stato assegnato all'istanza EC2 al momento della creazione. Se l'istanza è stata taggata in maniera diversa da **`MyCodePipelineDemo`**, assicurarsi di utilizzare tale valore qui.

6. In Configurazione dell'agente con AWS Systems Manager, scegli Ora e pianifica gli aggiornamenti. Questo installa l'agente sull'istanza. L'istanza Windows è già configurata con l'agente SSM e verrà ora aggiornata con l' CodeDeploy agente.
7. In Impostazioni di distribuzione, scegli`CodeDeployDefault.OneAtaTime`.
8. In Load Balancer, assicurati che la casella Enable load balancing non sia selezionata. Non è necessario configurare un sistema di bilanciamento del carico o scegliere un gruppo di destinazione per questo esempio. Dopo aver deselezionato la casella di controllo, le opzioni di bilanciamento del carico non vengono visualizzate.
9. Nella sezione Advanced (Impostazioni avanzate) mantenere le impostazioni predefinite.
10. Scegliere Create deployment group (Crea gruppo di distribuzione).

Passaggio 4: Crea la tua prima pipeline in CodePipeline

In questa parte del tutorial, verrà creata la pipeline. L'esempio viene eseguito automaticamente tramite la pipeline.

Per creare un CodePipeline processo di rilascio automatico

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegliere Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyFirstPipeline`**.

> **Note**
>
> Se si sceglie un altro nome per la pipeline, accertarsi di utilizzarlo al posto di **`MyFirstPipeline`** nella parte restante di questo tutorial. Non è possibile modificare il

nome di una pipeline dopo che è stata creata. I nomi di pipeline sono soggetti ad alcune limitazioni. Per ulteriori informazioni, consulta Quote in AWS CodePipeline.

4. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Service role (Ruolo del servizio), procedere in uno dei seguenti modi:
   - Scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un nuovo ruolo di servizio in IAM.
   - Scegli Existing service role (Ruolo di servizio esistente) per usare un ruolo del servizio già creato in IAM. In Role name (Nome ruolo), scegli il ruolo del servizio dall'elenco.
6. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).
7. In Step 2: Add source stage (Fase 2: aggiunta della fase di origine), in Source provider (Provider di origine), scegli Amazon S3. In Bucket, immettere il nome del bucket S3 creato in Fase 1: creazione di un bucket S3 per l'applicazione. Nella chiave oggetto S3, immettere la chiave oggetto con o senza un percorso file e ricordarsi di includere l'estensione del file. Ad esempio, per `SampleApp_Windows.zip`, immettere il nome del file di esempio come illustrato in questo esempio:

   ```
   SampleApp_Windows.zip
   ```

   Seleziona Fase successiva.

   In Change detection options (Opzioni di rilevamento delle modifiche), lasciare le impostazioni predefinite. Ciò consente di CodePipeline utilizzare Amazon CloudWatch Events per rilevare le modifiche nel bucket di origine.

   Seleziona Successivo.
8. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Successivo.
9. Nel passaggio 4: aggiungi la fase di distribuzione, in Deploy provider, scegli. CodeDeploy Per impostazione predefinita, il campo Regione è lo stesso Regione AWS della pipeline. In Application name (Nome applicazione), immettere `MyDemoApplication` oppure scegliere

il pulsante Refresh (Aggiorna) e quindi selezionare il nome dell'applicazione dall'elenco. In Deployment group (Gruppo di distribuzione), immettere **`MyDemoDeploymentGroup`** o selezionarlo dall'elenco, quindi scegliere Next (Successivo).

> **Note**
>
> Il nome "Deploy (Distribuzione)" è il nome assegnato per impostazione predefinita alla fase creata in Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione), così come "Source (Origine)" è il nome assegnato alla prima fase della pipeline.

10. In Step 5: Review (Fase 5: revisione), esaminare le informazioni e quindi scegliere Create pipeline (Crea pipeline).
11. La pipeline inizia l'esecuzione. Puoi visualizzare lo stato di avanzamento e i messaggi di successo e di fallimento mentre l' CodePipeline esempio distribuisce una pagina Web su ciascuna delle istanze Amazon EC2 della distribuzione. CodeDeploy

Complimenti! Hai appena creato una semplice pipeline in. CodePipeline La pipeline è composta da due fasi:

- Una fase di origine denominata Source (Origine), che rileva le modifiche nell'applicazione di esempio con versioni multiple archiviate nel bucket S3 ed esegue il push di tali modifiche nella pipeline.
- Una fase di implementazione che implementa tali modifiche alle istanze EC2 con. CodeDeploy

Verifica ora i risultati.

Per verificare che la pipeline è stata eseguita correttamente

1. Visualizzare l'avanzamento iniziale della pipeline. Lo stato di ciascuna fase cambia da No executions yet (Ancora nessun esecuzione) a In Progress (In corso) e quindi in Succeeded (Riuscito) o Failed (Non riuscito). L'esecuzione della pipeline richiede qualche minuto.
2. Quando lo stato dell'operazione indica Succeeded (Riuscito), nell'area di stato per la fase Deploy (Distribuzione), scegliere Details (Dettagli). Verrà aperta la console. CodeDeploy
3. Nella scheda Deployment group (Gruppo di distribuzione), in Deployment lifecycle events (Eventi del ciclo di vita di distribuzione), scegliere un ID dell'istanza. Si apre la console EC2.

4. Nella scheda Description (Descrizione), in Public DNS (DNS pubblico), copiare l'indirizzo e incollarlo nella barra degli indirizzi del browser Web. Visualizzare la pagina di indice per l'applicazione di esempio caricata nel bucket S3.

   Viene visualizzata la pagina Web dell'applicazione di esempio che hai caricato nel tuo bucket S3.

Per ulteriori informazioni sulle fasi, sulle operazioni e sul funzionamento delle pipeline, consulta CodePipeline concetti .

(Opzionale) Fase 5: aggiunta di un'altra fase alla pipeline

Ora aggiungi un'altra fase della pipeline da distribuire dai server di staging ai server di produzione utilizzando. CodeDeploy Innanzitutto, crei un altro gruppo di distribuzione all'interno. CodePipelineDemoApplication CodeDeploy Quindi, aggiungi una fase che include un'operazione che utilizza questo gruppo di distribuzione. Per aggiungere un'altra fase, è necessario utilizzare la CodePipeline console o AWS CLI recuperare e modificare manualmente la struttura della pipeline in un file JSON, quindi eseguire il update-pipeline comando per aggiornare la pipeline con le modifiche.

Argomenti

- Crea un secondo gruppo di distribuzione in CodeDeploy
- Aggiunta del gruppo di distribuzione come un'altra fase nella pipeline

Crea un secondo gruppo di distribuzione in CodeDeploy

> **Note**
>
> In questa parte del tutorial, crei un secondo gruppo di distribuzione, ma esegui la distribuzione sulle stesse istanze Amazon EC2 di prima. Questo esempio è solo a scopo dimostrativo. È stato progettato appositamente per non mostrarti come vengono visualizzati gli errori. CodePipeline

Per creare un secondo gruppo di distribuzione in CodeDeploy

1. Apri la CodeDeploy console all'indirizzo https://console.aws.amazon.com/codedeploy.
2. Scegliere Applications (Applicazioni) e selezionare `MyDemoApplication` nell'elenco di applicazioni.

3. Scegliere la scheda Deployment groups (Gruppi di distribuzione), quindi scegliere Create deployment group (Crea gruppo di distribuzione).
4. Nella pagina Create deployment group (Crea gruppo di distribuzione), in Deployment group name (Nome del gruppo di distribuzione), immettere un nome per il secondo gruppo di distribuzione, ad esempio **`CodePipelineProductionFleet`**.
5. In Service Role, scegli lo stesso ruolo di CodeDeploy servizio che hai usato per la distribuzione iniziale (non il ruolo CodePipeline di servizio).
6. In Deployment type (Tipo di distribuzione), scegliere In-place (In loco).
7. In Environment configuration (Configurazione dell'ambiente), scegliere Amazon EC2 Instances (Istanze Amazon EC2). Scegliere Nome nella casella Key (Chiave) e nella casella Value (Valore) scegliere `MyCodePipelineDemo` dall'elenco. Lasciare la configurazione predefinita per Deployment settings (Impostazioni di distribuzione).
8. In Deployment configuration (Configurazione della distribuzione), scegliere `CodeDeployDefault.OneAtaTime`.
9. In Load Balancer (Sistema di bilanciamento del carico), deselezionare Enable load balancing (Abilita il bilanciamento del carico).
10. Scegliere Create deployment group (Crea gruppo di distribuzione).

Aggiunta del gruppo di distribuzione come un'altra fase nella pipeline

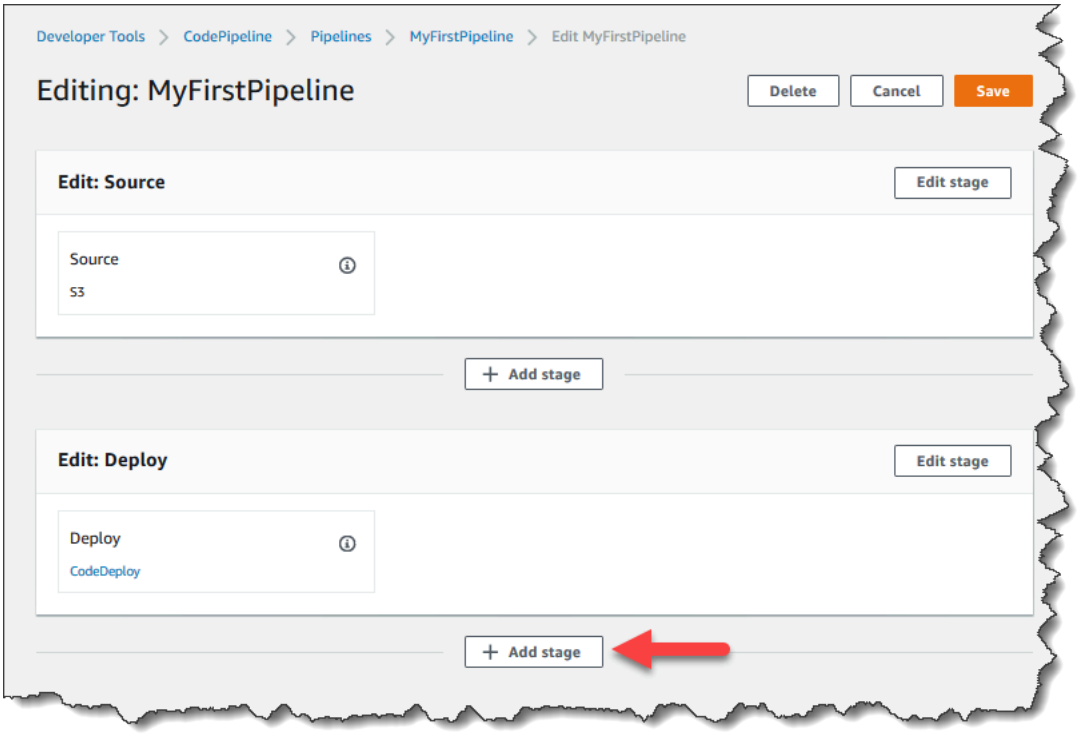
Ora che disponi di un altro gruppo di distribuzione, puoi aggiungere una fase che utilizza questo gruppo di distribuzione per la distribuzione alle stesse istanze EC2 utilizzate in precedenza. Puoi usare la CodePipeline console o AWS CLI aggiungere questa fase.

Argomenti

Creazione di una terza fase (console)

È possibile utilizzare la CodePipeline console per aggiungere una nuova fase che utilizza il nuovo gruppo di distribuzione. Poiché questo gruppo di distribuzione sta eseguendo la distribuzione alle istanze EC2 che hai già utilizzato, l'operazione di distribuzione in questa fase non va a buon fine.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. In Nome, scegli il nome della pipeline che hai creato, MyFirstPipeline.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. Nella pagina Edit (Modifica), scegli + Add stage (Aggiungi fase) per aggiungere una fase subito dopo la fase Deploy (Distribuzione).



5. In Aggiungi fase, in Nome fase, immettete**`Production`**. Selezionare Add stage (Aggiungi fase).
6. Nella nuova fase, scegliere + Add action group (Aggiungi gruppo di operazioni).
7. In Modifica azione, in Nome azione, inserisci**`Deploy-Second-Deployment`**. In Action provider, in Deploy, scegli CodeDeploy.
8. Nella CodeDeploy sezione, in Nome applicazione, scegli `MyDemoApplication` dall'elenco a discesa, come hai fatto quando hai creato la pipeline. In Deployment group (Gruppo di distribuzione), scegli il gruppo di distribuzione appena creato **`CodePipelineProductionFleet`**. In Artefatti di input, scegliere l'artefatto di input dall'azione sorgente. Selezionare Salva.
9. Nella pagina Edit (Modifica), scegliere Save (Salva). In Save pipeline changes (Salva modifiche alla pipeline), scegliere Save (Salva).

10. Anche se la nuova fase è stata aggiunta alla pipeline, viene visualizzato lo stato No executions yet (Ancora nessun esecuzione) perché le modifiche non hanno attivato un'altra esecuzione della pipeline. Per vedere come viene eseguita la pipeline modificata, è necessario rieseguire manualmente la revisione più recente. Nella pagina dei dettagli della pipeline, scegli Release change, quindi scegli Release quando richiesto. In questo modo viene eseguita la revisione più recente disponibile in ogni percorso di origine specificato in un'operazione origine tramite la pipeline.

    In alternativa, per utilizzare il AWS CLI per rieseguire la pipeline, da un terminale sul computer Linux, macOS o Unix locale o da un prompt dei comandi sul computer Windows locale, esegui il comando, start-pipeline-execution specificando il nome della pipeline. In questo modo l'applicazione viene eseguita nel bucket di origine tramite la pipeline per una seconda volta.

    ```
    aws codepipeline start-pipeline-execution --name MyFirstPipeline
    ```

    Questo comando restituisce un oggetto `pipelineExecutionId`.

11. Tornate alla CodePipeline console e nell'elenco delle pipeline, scegliete di aprire la pagina di visualizzazione. MyFirstPipeline

    La pipeline mostra tre fasi e lo stato dell'artefatto in esecuzione attraverso queste tre fasi. L'esecuzione in tutte le fasi della pipeline potrebbe richiedere fino a cinque minuti. La distribuzione va a buon fine nelle prime due fasi, come in precedenza, ma nella fase Production (Produzione) l'operazione Deploy-Second-Deployment (Distribuisci-seconda-distribuzione) viene mostrata come non riuscita.

12. Nell'operazione Deploy-Second-Deployment (Distribuisci-seconda-distribuzione), scegliere Details (Dettagli). Verrai reindirizzato alla pagina per la CodeDeploy distribuzione. In questo caso, l'errore è il risultato della distribuzione del primo gruppo di istanze a tutte le istanze EC2, senza lasciare alcuna istanza per il secondo gruppo di distribuzione.

    > **Note**
    >
    > Questo errore è un comportamento predefinito per dimostrare cosa accade quando si verifica un errore in una fase della pipeline.

Creazione di una terza fase (CLI)

Sebbene l'utilizzo AWS CLI di per aggiungere una fase alla pipeline sia più complesso rispetto all'utilizzo della console, offre una maggiore visibilità sulla struttura della pipeline.

Per creare una terza fase per la pipeline

1. Apri una sessione terminale sul tuo computer Linux, macOS o Unix locale o un prompt dei comandi sul tuo computer Windows locale ed esegui il get-pipeline comando per visualizzare la struttura della pipeline appena creata. Per **`MyFirstPipeline`**, digitare il comando seguente:

   ```
   aws codepipeline get-pipeline --name "MyFirstPipeline"
   ```

   Questo comando restituisce la struttura di. MyFirstPipeline L'aspetto della prima parte dell'output è simile al seguente:

   ```
   {
       "pipeline": {
           "roleArn": "arn:aws:iam::80398EXAMPLE:role/AWS-CodePipeline-Service",
           "stages": [
       ...
   ```

   La parte finale dell'output include i metadati della pipeline e l'aspetto è simile al seguente:

   ```
       ...
           ],
           "artifactStore": {
               "type": "S3"
               "location": "codepipeline-us-east-2-250656481468",
           },
           "name": "MyFirstPipeline",
           "version": 4
       },
       "metadata": {
           "pipelineArn": "arn:aws:codepipeline:us-
   east-2:80398EXAMPLE:MyFirstPipeline",
           "updated": 1501626591.112,
           "created": 1501626591.112
       }
   }
   ```

2. Copiare e incollare questa struttura in un editor di testo semplice e salvare il file come **`pipeline.json`**. Per comodità, salvare il file nella stessa directory in cui si eseguono i comandi aws codepipeline.

   > **Note**
   >
   > È possibile eseguire il piping di JSON direttamente in un file con il comando get-pipeline come segue:
   >
   > ```
   > aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   > ```

3. Copiare la sezione della fase Deploy (Distribuzione) e incollarla dopo le prime due fasi. Poiché si tratta di una fase di distribuzione, analogamente alla fase Deploy (Distribuzione), utilizzarla come un modello per la terza fase.

4. Cambia il nome della fase e i dettagli del gruppo di distribuzione.

   L'esempio seguente mostra il codice JSON aggiunto al file pipeline.json dopo la fase di distribuzione. Modificare gli elementi sottolineati con nuovi valori. Ricordare di includere una virgola per separare le definizioni delle fasi Deploy (Distribuzione) e Production (Produzione).

```
,
{
    "name": "Production",
     "actions": [
        {
         "inputArtifacts": [
             {
              "name": "MyApp"
             }
          ],
          "name": "Deploy-Second-Deployment",
          "actionTypeId": {
              "category": "Deploy",
              "owner": "AWS",
              "version": "1",
              "provider": "CodeDeploy"
              },
          "outputArtifacts": [],
          "configuration": {
              "ApplicationName": "CodePipelineDemoApplication",
```

```
                "DeploymentGroupName": "CodePipelineProductionFleet"
                 },
             "runOrder": 1
            }
        ]
}
```

5. Se stai utilizzando la struttura della pipeline recuperata tramite il comando get-pipeline, devi rimuovere le righe `metadata` dal file JSON. In caso contrario, il comando update-pipeline non è in grado di utilizzarlo. Rimuovi le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`.

   Ad esempio, rimuovere dalla struttura le seguenti righe:

   ```
   "metadata": {
     "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
     "created": "date",
     "updated": "date"
     }
   ```

   Salvare il file.

6. Eseguire il comando update-pipeline, specificando il file JSON della pipeline, in modo analogo al seguente:

   ```
   aws codepipeline update-pipeline --cli-input-json file://pipeline.json
   ```

   Questo comando restituisce l'intera struttura della pipeline aggiornata.

   > **Important**
   >
   > Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

7. Eseguire il comando start-pipeline-execution, specificando il nome della pipeline. In questo modo l'applicazione viene eseguita nel bucket di origine tramite la pipeline per una seconda volta.

   ```
   aws codepipeline start-pipeline-execution --name MyFirstPipeline
   ```

   Questo comando restituisce un oggetto `pipelineExecutionId`.

8. Apri la CodePipeline console e scegli MyFirstPipelinedall'elenco delle pipeline.

   La pipeline mostra tre fasi e lo stato dell'artefatto in esecuzione attraverso queste tre fasi. L'esecuzione in tutte le fasi della pipeline potrebbe richiedere fino a cinque minuti. Anche se la distribuzione va buon fine sulle prime due fasi, come in precedenza, la fase Production (Produzione) mostra che l'operazione Deploy-Second-Deployment (Distribuisci-seconda-distribuzione) non è riuscita.

9. Nell'operazione Deploy-Second-Deployment (Distribuisci-seconda-distribuzione), scegliere Details (Dettagli) per visualizzare i dettagli dell'errore. Verrai reindirizzato alla pagina dei dettagli della CodeDeploy distribuzione. In questo caso, l'errore è il risultato della distribuzione del primo gruppo di istanze a tutte le istanze EC2, senza lasciare alcuna istanza per il secondo gruppo di distribuzione.

   > **Note**
   >
   > Questo errore è un comportamento predefinito per dimostrare cosa accade quando si verifica un errore in una fase della pipeline.

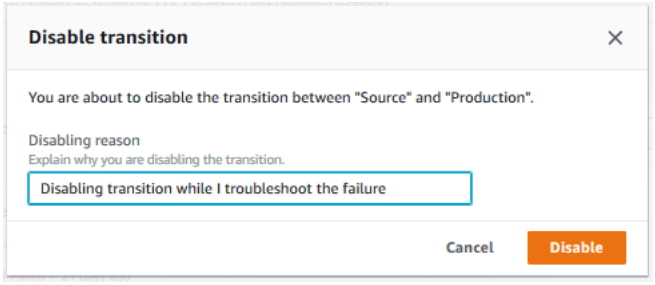
(Facoltativo) Fase 6: Disabilitare e abilitare le transizioni tra le fasi in CodePipeline

Puoi abilitare o disabilitare la transizione tra fasi in una pipeline. La disabilitazione della transizione tra fasi consente di controllare manualmente le transizioni tra una fase e un'altra. Ad esempio, potrebbe essere necessario eseguire le prime due fasi di una pipeline, ma disabilitare le transizioni alla terza fase finché non si è pronti per distribuire in produzione, oppure durante la risoluzione di un problema o errore relativo a tale fase.

Per disabilitare e abilitare le transizioni tra le fasi di una pipeline CodePipeline

1. Apri la CodePipeline console e scegli MyFirstPipelinedall'elenco delle pipeline.
2. Nella pagina dei dettagli per la pipeline, scegliere il pulsante Disable transition (Disabilita transizione) tra la seconda fase, (Deploy (Distribuzione)) e la terza fase aggiunta nella sezione precedente, (Production (Produzione)).
3. In Disable transition (Disabilita transizione), specificare il motivo della disabilitazione della transizione tra le fasi, quindi scegliere Disable (Disabilita).

La freccia tra le fasi visualizza un'icona e una modifica del colore e viene visualizzato il pulsante Enable transition (Abilita transizione).



4. Caricare nuovamente l'esempio nel bucket S3. Poiché il bucket è con versioni multiple, questa modifica avvia la pipeline.
5. Tornare alla pagina dei dettagli per la pipeline e osservare lo stato delle fasi. La visualizzazione della pipeline cambia per mostrare l'avanzamento e i messaggi di esito positivo sulle prime due fasi, ma nessuna modifica viene apportata sulla terza fase. Questo processo potrebbe richiedere alcuni minuti.
6. Abilitare la transizione selezionando il pulsante Enable transition (Abilita transazione) tra le due fasi. Nella finestra di dialogo Enable transition (Abilita transazione), scegliere Enable (Abilita). La fase inizia l'esecuzione entro pochi minuti e tenta di elaborare l'artefatto che è già stato eseguito nelle prime due fasi della pipeline.

> **Note**
>
> Se vuoi che questa terza fase abbia successo, modifica il gruppo di CodePipelineProductionFleet distribuzione prima di abilitare la transizione e specifica un set diverso di istanze EC2 in cui viene distribuita l'applicazione. Per ulteriori informazioni su come eseguire questa operazione, consulta Modifica delle impostazioni del gruppo di distribuzione. Se si creano più istanze EC2, è possibile che vengano addebitati costi aggiuntivi.

Fase 7: eliminazione delle risorse

Puoi utilizzare alcune delle risorse create in questo tutorial per il Tutorial: creazione di una pipeline a quattro fasi. Ad esempio, puoi riutilizzare l'applicazione e la CodeDeploy distribuzione. Puoi configurare un'azione di compilazione con un provider come CodeBuild, che è un servizio di

compilazione completamente gestito nel cloud. È inoltre possibile configurare un'operazione di compilazione che utilizza un provider con un server o sistema di compilazione, ad esempio Jenkins.

Tuttavia, dopo aver completato questo ed eventuali altri tutorial, devi eliminare la pipeline e le risorse utilizzate, per evitare che ti venga addebitato un costo per l'utilizzo continuo di tali risorse. Innanzitutto, elimina la pipeline, quindi l' CodeDeploy applicazione e le istanze Amazon EC2 associate e infine il bucket S3.

Per eliminare le risorse utilizzate in questo tutorial

1. Per ripulire CodePipeline le risorse, segui le istruzioni riportate in Eliminare una pipeline in. AWS CodePipeline
2. Per ripulire le CodeDeploy risorse, segui le istruzioni in Per pulire le risorse (console).
3. Per eliminare il bucket S3, segui le istruzioni in Eliminazione o svuotamento di un bucket S3. Se non si prevede di creare altre pipeline, eliminare il bucket S3 creato per l'archiviazione degli artefatti pipeline. Per ulteriori informazioni su questo bucket, consulta CodePipeline concetti .

Tutorial: crea una pipeline semplice (CodeCommitrepository)

In questo tutorial, lo utilizzerai CodePipeline per distribuire il codice gestito in un CodeCommit repository su una singola istanza Amazon EC2. La tua pipeline viene attivata quando invii una modifica al repository. CodeCommit La pipeline distribuisce le modifiche a un'istanza Amazon EC2 CodeDeploy utilizzandola come servizio di distribuzione.

La pipeline è composta da due fasi:

- Una fase di origine (Source) per la tua CodeCommit azione di origine.
- Una fase di distribuzione (Deploy) per l'azione CodeDeploy di distribuzione.

Il modo più semplice per iniziare AWS CodePipeline è utilizzare la procedura guidata Create Pipeline nella console. CodePipeline

> **Note**
>
> Prima di iniziare, assicurati di aver configurato il tuo client Git per utilizzarlo CodeCommit. Per istruzioni, vedi Configurazione per CodeCommit.

Fase 1: Creare un CodeCommit repository

Innanzitutto, crei un repository in. CodeCommit La pipeline riceve il codice sorgente da questo repository quando viene eseguita. Inoltre, crei un repository locale in cui conservare e aggiornare il codice prima di inviarlo al CodeCommit repository.

Per creare un repository CodeCommit

1. Apri la CodeCommit console all'indirizzo https://console.aws.amazon.com/codecommit/.
2. Nel selettore della regione, scegli Regione AWS dove vuoi creare il repository e la pipeline. Per ulteriori informazioni, consulta Regioni AWS ed endpoint.
3. Nella pagina Repositories (Repository), scegli Create repository (Crea repository).
4. Nella pagina Create repository (Crea repository), in Repository name (Nome repository), immetti un nome per il repository (ad esempio **MyDemoRepo**).
5. Scegli Crea.

> **Note**
>
> I passaggi rimanenti di questo tutorial vengono utilizzati **MyDemoRepo** per il nome del repository. CodeCommit Se scegli un nome differente, assicurati di utilizzarlo in tutto il tutorial.

Per configurare un repository locale

In questo passaggio, configurerai un repository locale per connetterti al tuo repository remoto CodeCommit.

> **Note**
>
> Non è necessario configurare un repository locale. È inoltre possibile utilizzare la console per caricare file come descritto inPassaggio 2: aggiungi codice di esempio al tuo CodeCommit repository.

1. Con il nuovo repository aperto nella console, scegli Clone URL (Clona URL) nella parte superiore destra della pagina, quindi seleziona Clone SSH (Clona SSH). L'indirizzo per clonare il repository Git viene copiato negli Appunti.
2. Nel terminale o nella riga di comando, passare a una directory locale in cui si desidera archiviare il repository locale. In questo tutorial, utilizziamo `/tmp`.
3. Esegui il comando seguente per clonare il repository, sostituendo l'indirizzo SSH con quello copiato nella fase precedente. Questo comando crea una directory denominata `MyDemoRepo`. Copia un'applicazione di esempio in questa directory.

```
git clone ssh://git-codecommit.us-west-2.amazonaws.com/v1/repos/MyDemoRepo
```

Passaggio 2: aggiungi codice di esempio al tuo CodeCommit repository

In questo passaggio, scarichi il codice per un'applicazione di esempio creata per una procedura dettagliata di CodeDeploy esempio e lo aggiungi al tuo repository. CodeCommit

1. Quindi, scaricare un esempio e salvarlo in una cartella o directory sul computer locale.

   a. Scegliere una delle seguenti opzioni. Scegli `SampleApp_Linux.zip` se vuoi seguire i passaggi di questo tutorial per le istanze Linux.

   - Se desideri eseguire la distribuzione su istanze Amazon Linux utilizzando CodeDeploy, scarica l'applicazione di esempio qui: SampleApp _Linux.zip.
   - Se desideri eseguire la distribuzione su istanze di Windows Server utilizzando CodeDeploy, scarica l'applicazione di esempio qui: _Windows.zip. SampleApp

   L'applicazione di esempio contiene i seguenti file con cui eseguire la distribuzione: CodeDeploy

   - `appspec.yml`— Il file delle specifiche dell'applicazione (AppSpecfile) è un file in formato YAML utilizzato da CodeDeploy per gestire una distribuzione. Per ulteriori informazioni sul AppSpec file, vedere CodeDeploy AppSpec File reference nella Guida per l'utente.AWS CodeDeploy
   - `index.html`— Il file indice contiene la home page dell'applicazione di esempio distribuita.

- `LICENSE.txt`— Il file di licenza contiene informazioni sulla licenza per l'applicazione di esempio.
- File per script: l'applicazione di esempio utilizza script per scrivere file di testo in una posizione sull'istanza. Un file viene scritto per ciascuno dei diversi eventi del ciclo di vita della CodeDeploy distribuzione nel modo seguente:
  - `scripts`Cartella (solo esempio Linux): la cartella contiene i seguenti script di shell per installare le dipendenze e avviare e arrestare l'applicazione di esempio per la distribuzione automatizzata:`install_dependencies`, e. `start_server` `stop_server`
  - (Solo esempio per Windows)`before-install.bat`: si tratta di uno script batch per l'evento del ciclo di vita della `BeforeInstall` distribuzione, che verrà eseguito per rimuovere i vecchi file scritti durante le distribuzioni precedenti di questo esempio e creare una posizione sull'istanza in cui scrivere i nuovi file.

b. Scarica il file compresso.

2. Decomprimi i file da SampleApp_Linux.zip nella directory locale creata in precedenza (ad esempio o). `/tmp/MyDemoRepo` `c:\temp\MyDemoRepo`

   Assicurati di inserire i file direttamente nel repository locale. Non includere una cartella `SampleApp_Linux`. Sul computer locale Linux, macOS o Unix, ad esempio, la gerarchia delle cartelle e dei file dovrebbe avere il seguente aspetto:

```
/tmp
   #-- MyDemoRepo
       #-- appspec.yml
       #-- index.html
       #-- LICENSE.txt
       #-- scripts
           #-- install_dependencies
           #-- start_server
           #-- stop_server
```

3. Per caricare file nel tuo repository, usa uno dei seguenti metodi.

   a. Per utilizzare la CodeCommit console per caricare i file:

      i. Apri la CodeCommit console e scegli il tuo repository dall'elenco Repository.

      ii. Seleziona Add file (Aggiungi file), quindi scegli Upload file (Carica file).

iii. Seleziona Choose file (Scegli file) e vai al file. Per aggiungere un file in una cartella, scegli Crea file, quindi inserisci il nome della cartella con il nome del file, ad esempio. `scripts/install_dependencies` Incolla il contenuto del file nel nuovo file.

Conferma la modifica inserendo il tuo nome utente e indirizzo e-mail.

Scegliere Commit changes (Applica modifiche).

iv. Ripetere questo passaggio per ogni file.

Il contenuto del repository dovrebbe avere il seguente aspetto:

```
#-- appspec.yml
#-- index.html
#-- LICENSE.txt
#-- scripts
    #-- install_dependencies
    #-- start_server
    #-- stop_server
```

b. Per usare i comandi git per caricare i tuoi file:

i. Cambiare le directory nel repository locale:

```
(For Linux, macOS, or Unix) cd /tmp/MyDemoRepo
(For Windows) cd c:\temp\MyDemoRepo
```

ii. Esegui il comando seguente per posizionare tutti i file contemporaneamente:

```
git add -A
```

iii. Esegui il comando seguente per eseguire il commit dei file con un messaggio di commit:

```
git commit -m "Add sample application files"
```

iv. Esegui il comando seguente per inviare i file dal repository locale al tuo CodeCommit repository:

```
git push
```

4. I file scaricati e aggiunti al repository locale sono stati ora aggiunti al `main` ramo del CodeCommit `MyDemoRepo` repository e sono pronti per essere inclusi in una pipeline.

Fase 3: creare un'istanza Amazon EC2 Linux e installare l'agente CodeDeploy

In questo passaggio, crei l'istanza Amazon EC2 in cui distribuisci un'applicazione di esempio. Come parte di questo processo, crea un ruolo dell'istanza che consenta l'installazione e la gestione dell' CodeDeploy agente sull'istanza. L' CodeDeploy agente è un pacchetto software che consente di utilizzare un'istanza nelle CodeDeploy distribuzioni. È inoltre possibile allegare politiche che consentono all'istanza di recuperare i file utilizzati dall' CodeDeploy agente per distribuire l'applicazione e di consentire la gestione dell'istanza da parte di SSM.

Per creare un ruolo dell'istanza

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/).
2. Dal pannello di controllo della console, scegli Roles (Ruoli).
3. Scegli Crea ruolo.
4. In Seleziona il tipo di entità affidabile, seleziona Servizio AWS. In Scegli un caso d'uso, seleziona EC2. In Select your use case (Seleziona il tuo caso d'uso) selezionare EC2. Scegli Successivo: autorizzazioni.
5. Cerca e seleziona la politica denominata **`AmazonEC2RoleforAWSCodeDeploy`**.
6. Cerca e seleziona la politica denominata **`AmazonSSMManagedInstanceCore`**. Scegliere Next: Tags (Successivo: Tag).
7. Scegliere Next:Review (Successivo: Rivedi). Immettere un nome per il ruolo (ad esempio **`EC2InstanceRole`**).

   > **Note**
   >
   > Prendere nota del nome del ruolo per la fase successiva. È possibile scegliere questo ruolo quando si crea l'istanza.

   Scegli Crea ruolo.

Per avviare un'istanza

1. Aprire la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.
2. Dalla barra di navigazione laterale, scegli Istanze e seleziona Avvia istanze nella parte superiore della pagina.
3. In Nome, inserisci **MyCodePipelineDemo**. Questo assegna all'istanza un tag Key di **Name** e un tag Value di. **MyCodePipelineDemo** Successivamente, si crea un' CodeDeploy applicazione che distribuisce l'applicazione di esempio su questa istanza. CodeDeployseleziona le istanze da distribuire in base ai tag.
4. In Immagini dell'applicazione e del sistema operativo (Amazon Machine Image), individua l'opzione AMI Amazon Linux con il AWS logo e assicurati che sia selezionata. (Questa AMI è descritta come AMI Amazon Linux 2 (HVM) ed è etichettata come «Idoneo al piano gratuito».)
5. In Tipo di istanza, scegli il `t2.micro` tipo idoneo al piano gratuito come configurazione hardware per la tua istanza.
6. In Coppia di chiavi (login), scegli una coppia di chiavi o creane una.

   Puoi anche scegliere Proceed without a key pair.

   > **Note**
   >
   > Ai fini di questo tutorial, è possibile procedere senza una coppia di chiavi. Per utilizzare SSH per connettersi alle istanze, creare o utilizzare una coppia di chiavi.

7. In Impostazioni di rete, procedi come segue.

   In Assegna automaticamente un IP pubblico, assicurati che lo stato sia Abilita.

   - Accanto a Assign a security group (Assegna un gruppo di sicurezza), scegliere Create a new security group (Crea un nuovo gruppo di sicurezza).
   - Nella riga SSH, in Tipo di sorgente, scegli Il mio IP.
   - Scegli Aggiungi gruppo di sicurezza, scegli HTTP, quindi in Tipo di origine, scegli Il mio IP.
8. Espandi Advanced details (Dettagli avanzati). Nel profilo dell'istanza IAM, scegli il ruolo IAM che hai creato nella procedura precedente (ad esempio,**EC2InstanceRole**).
9. In Riepilogo, in Numero di istanze, inserisci.. 1
10. Scegliere Launch Instance (Avvia istanza).

11. È possibile visualizzare lo stato dell'avvio nella pagina Instances (Istanze). Quando avvii un'istanza, il suo stato iniziale è `pending`. Una volta avviata l'istanza, il suo stato passa a `running` e riceve un nome DNS pubblico. Se la colonna Public DNS (DNS pubblico) non è visualizzata, scegliere l'icona Show/Hide (Mostra/Nascondi) quindi selezionare Public DNS (DNS pubblico).

Passaggio 4: Creare un'applicazione in CodeDeploy

In CodeDeploy, un'applicazione è una risorsa che contiene l'applicazione software che si desidera distribuire. Successivamente, utilizzerai questa applicazione CodePipeline per automatizzare le distribuzioni dell'applicazione di esempio sulla tua istanza Amazon EC2.

Innanzitutto, crei un ruolo che CodeDeploy consenta di eseguire le distribuzioni. Quindi, crea un'applicazione CodeDeploy .

Per creare un ruolo di servizio CodeDeploy

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/).
2. Dal pannello di controllo della console, scegli Roles (Ruoli).
3. Scegli Crea ruolo.
4. In Seleziona un'entità affidabile, scegli Servizio AWS. In Use case (Caso d'uso), scegli CodeDeploy. Scegli CodeDeploytra le opzioni elencate. Seleziona Successivo. La policy gestita `AWSCodeDeployRole` è già collegata al ruolo.
5. Seleziona Successivo.
6. Immetti un nome per il ruolo, ad esempio **`CodeDeployRole`**, quindi seleziona Create role (Crea ruolo).

Per creare un'applicazione in CodeDeploy

1. Apri la CodeDeploy console all'indirizzo https://console.aws.amazon.com/codedeploy.
2. Se la pagina Applicazioni non viene visualizzata, nel menu, scegli Applicazioni.
3. Scegli Crea applicazione.
4. In Application name (Nome applicazione), immettere **`MyDemoApplication`**.
5. In Compute Platform (Piattaforma di calcolo), scegliere EC2/On-premises (EC2/Locale).
6. Scegli Crea applicazione.

Per creare un gruppo di distribuzione in CodeDeploy

Un gruppo di distribuzione è una risorsa che definisce le impostazioni correlate alla distribuzione, come le istanze a cui distribuire e la velocità di distribuzione.

1. Nella pagina in cui è visualizzata l'applicazione, scegliere Create deployment group (Crea gruppo di distribuzione).
2. In Deployment group name (Nome del gruppo di distribuzione), immettere **`MyDemoDeploymentGroup`**.
3. In Ruolo di servizio, scegli l'ARN del ruolo di servizio creato in precedenza (ad esempio,**`arn:aws:iam::`*`account_ID`*`:role/CodeDeployRole`**).
4. In Deployment type (Tipo di distribuzione), scegliere In-place (In loco).
5. In Environment configuration (Configurazione dell'ambiente), scegliere Amazon EC2 Instances (Istanze Amazon EC2). Nel campo Chiave, immettete**`Name`**. Nel campo Valore, inserisci il nome che hai usato per etichettare l'istanza (ad esempio,**`MyCodePipelineDemo`**).
6. In Configurazione dell'agente con AWS Systems Manager, scegli Ora e pianifica gli aggiornamenti. Questo installa l'agente sull'istanza. L'istanza Linux è già configurata con l'agente SSM e verrà ora aggiornata con l' CodeDeploy agente.
7. In Deployment configuration (Configurazione della distribuzione), scegliere `CodeDeployDefault.OneAtaTime`.
8. In Load Balancer, assicurati che l'opzione Abilita bilanciamento del carico non sia selezionata. Non è necessario configurare un sistema di bilanciamento del carico o scegliere un gruppo di destinazione per questo esempio.
9. Scegliere Create deployment group (Crea gruppo di distribuzione).

Fase 5: Crea la tua prima pipeline in CodePipeline

È ora possibile creare ed eseguire la prima pipeline. In questo passaggio, crei una pipeline che viene eseguita automaticamente quando il codice viene inviato al tuo repository. CodeCommit

Per creare una pipeline CodePipeline

1. Accedere a AWS Management Console e aprire la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

2. Scegliere Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyFirstPipeline`**.
4. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Ruolo di servizio, scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.
6. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).
7. Nel passaggio 2: Aggiungi la fase di origine, in Provider di origine, scegli CodeCommit. In Nome archivio, scegli il nome del CodeCommit repository in cui hai creato. Fase 1: Creare un CodeCommit repository In Branch name (Nome ramo), scegliere `main`, quindi selezionare Next step (Fase successiva).

   Dopo aver selezionato il nome del repository e il ramo, un messaggio mostra la regola Amazon CloudWatch Events da creare per questa pipeline.

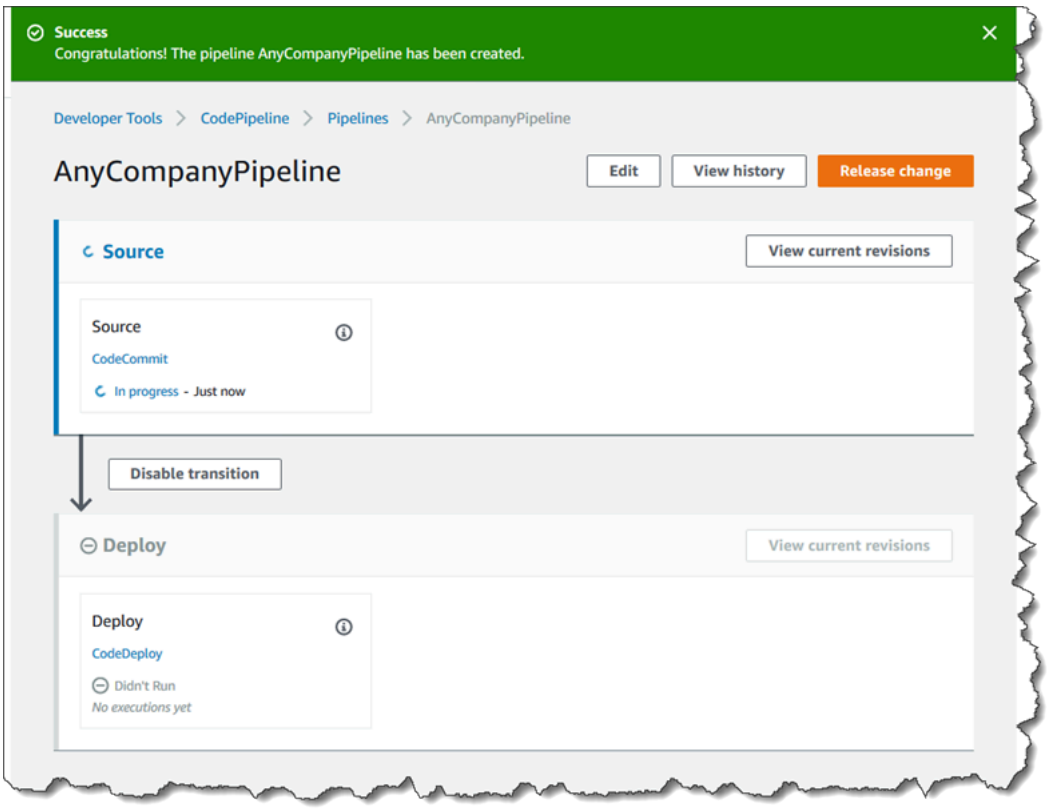
   In Change detection options (Opzioni di rilevamento delle modifiche), lasciare le impostazioni predefinite. Ciò consente di CodePipeline utilizzare Amazon CloudWatch Events per rilevare le modifiche nel tuo repository di origine.

   Seleziona Successivo.
8. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Successivo.

   > **Note**
   >
   > In questo tutorial, viene distribuito codice che non richiede alcun servizio di compilazione, in modo da poter ignorare questa fase. Tuttavia, se il codice sorgente deve essere creato prima di essere distribuito sulle istanze, puoi configurarlo CodeBuildin questo passaggio.

9. Nel passaggio 4: aggiungi la fase di distribuzione, in Deploy provider, scegli. CodeDeploy In Application name (Nome applicazione), scegliere **`MyDemoApplication`**. In Deployment group (Gruppo di distribuzione), scegliere **`MyDemoDeploymentGroup`**, quindi Next step (Fase successiva).

10. In Step 5: Review (Fase 5: revisione), esaminare le informazioni e quindi scegliere Create pipeline (Crea pipeline).
11. La pipeline viene avviata dopo la creazione. Scarica il codice dal tuo CodeCommit repository e crea una CodeDeploy distribuzione nella tua istanza EC2. Puoi visualizzare lo stato di avanzamento e i messaggi di successo e di errore mentre l' CodePipeline esempio distribuisce la pagina Web sull'istanza Amazon EC2 nella distribuzione. CodeDeploy



Complimenti! Hai appena creato una pipeline semplice in. CodePipeline

Viene quindi eseguita la verifica dei risultati.

Per verificare che la pipeline è stata eseguita correttamente

1. Visualizzare l'avanzamento iniziale della pipeline. Lo stato di ciascuna fase cambia da No executions yet (Ancora nessun esecuzione) a In Progress (In corso) e quindi in Succeeded (Riuscito) o Failed (Non riuscito). L'esecuzione della pipeline richiede qualche minuto.
2. Dopo aver visualizzato Succeeded per lo stato della pipeline, nell'area di stato della fase di distribuzione, scegliete. CodeDeploy Verrà aperta la console. CodeDeploy Se Succeeded (Riuscito) non viene visualizzato, consulta Risoluzione dei problemi CodePipeline.

3. Nella scheda Deployments (Distribuzioni), scegliere, l'ID della distribuzione. Nella pagina relativa alla distribuzione, in Deployment lifecycle events (Eventi del ciclo di vita di distribuzione), scegliere l'ID dell'istanza. Si apre la console EC2.
4. Nella scheda Description (Descrizione), in Public DNS (DNS pubblico), copiare l'indirizzo (ad esempio, `ec2-192-0-2-1.us-west-2.compute.amazonaws.com`) e incollarlo nella barra degli indirizzi del browser Web.

   Viene visualizzata la pagina Web dell'applicazione di esempio che hai scaricato e inviato al tuo CodeCommit repository.

Per ulteriori informazioni sulle fasi, sulle operazioni e sul funzionamento delle pipeline, consulta CodePipeline concetti .

Passaggio 6: Modifica il codice nel tuo repository CodeCommit

La pipeline è configurata per essere eseguita ogni volta che vengono apportate modifiche al codice del CodeCommit repository. In questo passaggio, si apportano modifiche al file HTML che fa parte dell' CodeDeployapplicazione di esempio nel CodeCommit repository. Quando esegui il push di queste modifiche, la pipeline viene eseguita nuovamente e le modifiche apportate sono visibili all'indirizzo Web a cui hai effettuato l'accesso in precedenza.

1. Cambiare le directory nel repository locale:

```
(For Linux, macOS, or Unix) cd /tmp/MyDemoRepo
(For Windows) cd c:\temp\MyDemoRepo
```

2. Utilizzare un editor di testo per modificare il file `index.html`:

```
(For Linux or Unix)gedit index.html
(For OS X)open -e index.html
(For Windows)notepad index.html
```

3. Rivedere il contenuto del file `index.html` per modificare il colore di sfondo e parte del testo sulla pagina Web, quindi salvare il file.

```
<!DOCTYPE html>
<html>
<head>
  <title>Updated Sample Deployment</title>
```

```
  <style>
    body {
      color: #000000;
      background-color: #CCFFCC;
      font-family: Arial, sans-serif;
      font-size:14px;
    }

    h1 {
      font-size: 250%;
      font-weight: normal;
      margin-bottom: 0;
    }

    h2 {
      font-size: 175%;
      font-weight: normal;
      margin-bottom: 0;
    }
  </style>
</head>
<body>
  <div align="center"><h1>Updated Sample Deployment</h1></div>
  <div align="center"><h2>This application was updated using CodePipeline,
 CodeCommit, and CodeDeploy.</h2></div>
  <div align="center">
    <p>Learn more:</p>
    <p><a href="https://docs.aws.amazon.com/codepipeline/latest/
userguide/">CodePipeline User Guide</a></p>
    <p><a href="https://docs.aws.amazon.com/codecommit/latest/
userguide/">CodeCommit User Guide</a></p>
    <p><a href="https://docs.aws.amazon.com/codedeploy/latest/
userguide/">CodeDeploy User Guide</a></p>
  </div>
</body>
</html>
```

4. Conferma e invia le modifiche al tuo CodeCommit repository eseguendo i seguenti comandi, uno alla volta:

```
git commit -am "Updated sample application files"
```

```
git push
```

Per verificare che la pipeline è stata eseguita correttamente

1. Visualizzare l'avanzamento iniziale della pipeline. Lo stato di ciascuna fase cambia da No executions yet (Ancora nessun esecuzione) a In Progress (In corso) e quindi in Succeeded (Riuscito) o Failed (Non riuscito). L'esecuzione della pipeline dovrebbe essere completata entro pochi minuti.
2. Dopo aver visualizzato Succeeded (Riuscito) per lo stato dell'operazione, aggiorna la pagina dimostrativa che hai consultato in precedenza nel browser.

   Viene visualizzata la pagina Web aggiornata.

Fase 7: eliminazione delle risorse

Puoi utilizzare alcune delle risorse create in questo tutorial per altri tutorial in questa guida. Ad esempio, è possibile riutilizzare l' CodeDeploy applicazione e la distribuzione. Tuttavia, dopo aver completato questo ed eventuali altri tutorial, devi eliminare la pipeline e le risorse utilizzate, per evitare che ti venga addebitato un costo per l'utilizzo continuo di tali risorse. Innanzitutto, elimina la pipeline, quindi l' CodeDeploy applicazione e l'istanza Amazon EC2 associata e infine il repository. CodeCommit

Per eliminare le risorse utilizzate in questo tutorial

1. Per ripulire CodePipeline le tue risorse, segui le istruzioni riportate in Eliminare una pipeline in. AWS CodePipeline
2. Per ripulire CodeDeploy le risorse, segui le istruzioni riportate in Clean Up Deployment Walkthrough Resources.
3. Per eliminare il CodeCommit repository, segui le istruzioni in Eliminare un repository. CodeCommit

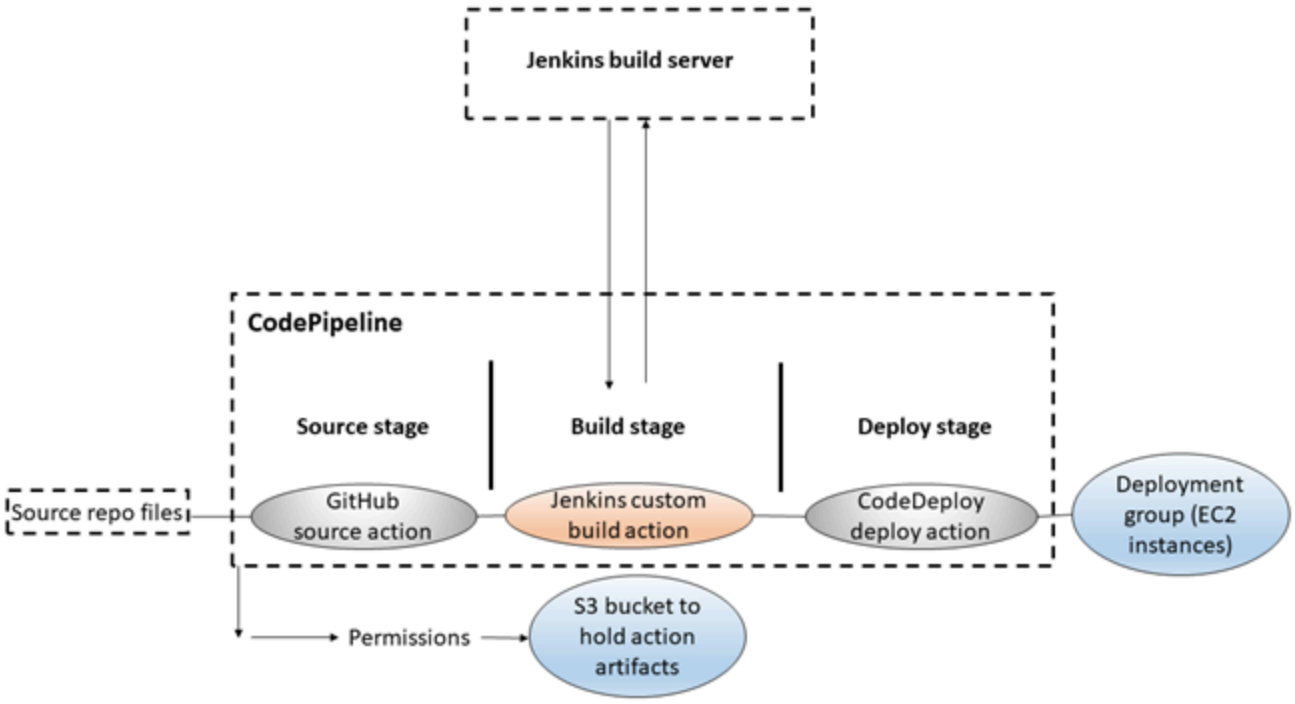
Fase 8: approfondimenti

Scopri di più su come funziona CodePipeline :

- Per ulteriori informazioni sulle fasi, sulle operazioni e sul funzionamento delle pipeline, consulta CodePipeline concetti .
- Per informazioni sulle azioni che è possibile eseguire utilizzando CodePipeline, vedereIntegrazioni con tipi di CodePipeline azioni.
- Prova questo tutorial più avanzato, Tutorial: creazione di una pipeline a quattro fasi. Crea una pipeline a più fasi che include una fase che compila il codice prima che venga distribuito.

Tutorial: creazione di una pipeline a quattro fasi

Ora che hai creato la tua prima pipeline in Tutorial: creazione di una semplice pipeline (bucket S3) o in Tutorial: crea una pipeline semplice (CodeCommitrepository), puoi iniziare a creare pipeline più complesse. Questo tutorial ti guiderà attraverso la creazione di una pipeline in quattro fasi che utilizza un GitHub repository per il codice sorgente, un server di build Jenkins per creare il progetto e un' CodeDeploy applicazione per distribuire il codice creato su un server di staging. Il diagramma seguente mostra la pipeline iniziale in tre fasi.



Dopo aver creato la pipeline, dovrai modificarla aggiungendo una fase con un'operazione di test per provare il codice, sempre tramite Jenkins.

Prima di poter creare la pipeline, devi configurare le risorse necessarie. Ad esempio, se si desidera utilizzare un GitHub repository per il codice sorgente, è necessario creare il repository prima di poterlo aggiungere a una pipeline. Nell'ambito della configurazione, il tutorial illustra i passaggi per configurare Jenkins su un'istanza EC2 a scopo dimostrativo.

> **Important**
>
> Molte delle azioni che aggiungi alla pipeline in questa procedura coinvolgono AWS risorse che devi creare prima di creare la pipeline. AWS le risorse per le tue azioni di origine devono sempre essere create nella stessa AWS regione in cui crei la pipeline. Ad esempio, se crei la pipeline nella regione Stati Uniti orientali (Ohio), il tuo CodeCommit repository deve trovarsi nella regione Stati Uniti orientali (Ohio).
>
> Puoi aggiungere azioni interregionali quando crei la pipeline. AWS le risorse per le azioni interregionali devono trovarsi nella stessa AWS regione in cui intendi eseguire l'azione. Per ulteriori informazioni, consulta Aggiungere un'azione interregionale in CodePipeline.

Prima di iniziare il tutorial, devi aver già completato i prerequisiti generali indicati in Guida introduttiva con CodePipeline.

Argomenti

Fase 1: completamento dei prerequisiti

Per l'integrazione con Jenkins, è AWS CodePipeline necessario installare il CodePipeline Plugin for Jenkins su qualsiasi istanza di Jenkins con cui si desidera utilizzare. CodePipeline Dovresti anche configurare un utente o un ruolo IAM dedicato da utilizzare per le autorizzazioni tra il tuo progetto Jenkins e. CodePipeline Il modo più semplice per integrare Jenkins CodePipeline consiste nell'installare Jenkins su un'istanza EC2 che utilizza un ruolo di istanza IAM creato per l'integrazione con Jenkins. Per un'efficace connessione dei collegamenti della pipeline per le operazioni Jenkins, occorre configurare le impostazioni del proxy e del firewall sul server; oppure dovrai configurare l'istanza EC2 in modo che autorizzi le connessioni in entrata alla porta utilizzata dal progetto Jenkins.

Verifica che la configurazione di Jenkins consenta di autenticare gli utenti e di applicare il controllo degli accessi prima di autorizzare le connessioni su queste porte (ad esempio, 443 e 8443 se hai protetto Jenkins in modo che utilizzi solo connessioni HTTPS o 80 e 8080 se autorizzi le connessioni HTTP). Per ulteriori informazioni, consulta l'articolo Garantire la sicurezza di Jenkins.

> **Note**
>
> Questo tutorial utilizza un codice di esempio e configura le fasi di compilazione che convertono il codice dal formato Haml al formato HTML. Puoi scaricare il codice di esempio open source dal GitHub repository seguendo la procedura riportata di seguito. Copia o clona il campione in un repository GitHub Avrai bisogno dell'intero campione nel tuo GitHub repository, non solo del file.zip.
> Il tutorial presuppone anche che:
>
> - Tu abbia familiarità con l'installazione e l'amministrazione di Jenkins e con la creazione di progetti Jenkins.
> - Tu abbia installato Rake e il gem Haml per Ruby sullo stesso computer o sulla stessa istanza che ospita il progetto Jenkins.
> - Tu abbia impostato le variabili di ambiente di sistema necessarie affinché i comandi Rake possano essere eseguiti dal terminale o dalla riga di comando (ad esempio, sui sistemi Windows, modificando la variabile PATH in modo da includere la directory dove è installato Rake).

Argomenti

Copia o clona il campione in un repository GitHub

Per clonare il campione e inviarlo a un repository GitHub

1. Scarica il codice di esempio dal GitHub repository o clona i repository sul tuo computer locale. Il codice di esempio è disponibile in due pacchetti:

- Se intendi distribuire il tuo campione su istanze di Amazon Linux, RHEL o Ubuntu Server, scegli _linux.zip. codepipeline-jenkins-aws-codedeploy
- Se intendi distribuire il campione su istanze di Windows Server, scegli -Jenkins- .zip. CodePipeline AWSCodeDeploy_Windows

2. Dal repository, scegliere Fork per copiare il repository di esempio in un repository dell'account Github. Per ulteriori informazioni, consulta la documentazione. GitHub

Crea un ruolo IAM da utilizzare per l'integrazione con Jenkins

Come best practice, prendi in considerazione l'avvio di un'istanza EC2 per ospitare il tuo server Jenkins e l'utilizzo di un ruolo IAM per concedere all'istanza le autorizzazioni necessarie per interagire con. CodePipeline

1. Accedi AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel riquadro di navigazione della console IAM, scegli Ruoli, quindi Crea ruolo.
3. In Select type of trusted entity (Seleziona tipo di entità attendibile), scegli Servizio AWS. In Choose the service that will use this role (Scegli il servizio che utilizzerà questo ruolo) scegliere EC2. In Select your use case (Seleziona il tuo caso d'uso) selezionare EC2.
4. Scegli Successivo: autorizzazioni. Nella pagina Collega policy di autorizzazione, seleziona la policy gestita `AWSCodePipelineCustomActionAccess`, quindi scegli Next: Tags. Scegli Prossimo: Rivedi.
5. Nella pagina Revisione, in Nome ruolo, inserisci il nome del ruolo da creare specificamente per l'integrazione con Jenkins (ad esempio, *`JenkinsAccess`*), quindi scegli Crea ruolo.

Quando crei l'istanza EC2 in cui installerai Jenkins, nel Passaggio 3: Configurazione dei dettagli dell'istanza, assicurati di scegliere il ruolo dell'istanza (ad esempio,). *`JenkinsAccess`*

Per ulteriori informazioni sui ruoli delle istanze e Amazon EC2, consulta Ruoli IAM per Amazon EC2, Utilizzo dei ruoli IAM per concedere autorizzazioni alle applicazioni in esecuzione su istanze Amazon EC2 e Creazione di un ruolo per delegare le autorizzazioni a un. Servizio AWS

Installa e configura Jenkins e il plugin per Jenkins CodePipeline

Per installare Jenkins e il plugin per Jenkins CodePipeline

1. Crea un'istanza EC2 in cui installare Jenkins e, nel Passaggio 3: Configurazione dei dettagli dell'istanza, assicurati di scegliere il ruolo dell'istanza che hai creato (ad esempio,). *JenkinsAccess* Per ulteriori informazioni sulla creazione di istanze EC2, consulta Launch an Amazon EC2 istanza nella Amazon EC2 User Guide.

   > **Note**
   >
   > Se disponi già di risorse Jenkins che desideri utilizzare, puoi farlo, ma devi creare un utente IAM speciale, applicare la policy `AWSCodePipelineCustomActionAccess` gestita a quell'utente e quindi configurare e utilizzare le credenziali di accesso per quell'utente sulla tua risorsa Jenkins. Se vuoi utilizzare l'interfaccia utente Jenkins per fornire le credenziali, configura Jenkins in modo che autorizzi solo il protocollo HTTPS. Per ulteriori informazioni, consulta Risoluzione dei problemi CodePipeline.

2. Installare Jenkins sull'istanza EC2. Per ulteriori informazioni, consulta la documentazione di Jenkins relativa all'installazione di Jenkins e all'avvio e all'accesso a Jenkins, oltre a details of integration with Jenkins in Integrazioni di prodotti e servizi con CodePipeline.
3. Avviare Jenkins e sulla home page scegliere Manage Jenkins (Gestisci Jenkins).
4. Nella pagina Manage Jenkins (Gestisci Jenkins), scegliere Manage Plugins (Gestisci plugin).
5. Scegliere la scheda Available (Disponibile) e nella casella di ricerca Filter (Filtra), immettere **`AWS CodePipeline`**. Scegli CodePipeline Plugin for Jenkins dall'elenco e scegli Scarica ora e installa dopo il riavvio.
6. Nella pagina Installing Plugins/Upgrades (Installazione plugin/Aggiornamenti), selezionare Restart Jenkins when installation is complete and no jobs are running (Riavvia Jenkins al termine dell'installazione e quando non ci sono processi in esecuzione).
7. Scegliere Back to Dashboard (Torna al pannello di controllo).
8. Nella pagina principale, scegliere New Item (Nuova voce).
9. In Item Name, inserisci un nome per il progetto Jenkins (ad esempio, *MyDemoProject*). Scegliere Freestyle project (Progetto freestyle), quindi OK.

> **Note**
>
> Assicurati che il nome del tuo progetto soddisfi i requisiti per CodePipeline. Per ulteriori informazioni, consulta Quote in AWS CodePipeline.

10. Nella pagina di configurazione del progetto, selezionare la casella di controllo Execute concurrent builds if necessary (Esegui compilazioni simultanee in caso di necessità). In Source Code Management (Gestione codice sorgente), scegli AWS CodePipeline. Se hai installato Jenkins su un'istanza EC2 e l'hai configurata AWS CLI con il profilo per l'utente IAM che hai creato per l'integrazione tra CodePipeline e Jenkins, lascia tutti gli altri campi vuoti.
11. Scegli Avanzato e, in Provider, inserisci un nome per il fornitore dell'azione così come verrà visualizzata CodePipeline (ad esempio,). *MyJenkinsProviderName* Verifica che il nome sia univoco e facile da ricordare. Verrà utilizzato più avanti nel tutorial quando si aggiunge un'operazione di compilazione e ancora quando si aggiunge un'operazione di test.

> **Note**
>
> Questo nome di azione deve soddisfare i requisiti di denominazione per le azioni in CodePipeline. Per ulteriori informazioni, consulta Quote in AWS CodePipeline.

12. In Build Triggers (Crea trigger), deselezionare tutte le caselle di controllo, quindi selezionare Poll SCM (Polling SCM). In Schedule (Pianificazione), immettere cinque asterischi separati da spazi, nel seguente modo:

```
* * * * *
```

Questo sondaggio viene effettuato CodePipeline ogni minuto.

13. In Build (Compila), scegliere Add build step (Aggiungi fase di compilazione). Scegli Esegui shell (Amazon Linux, RHEL o Ubuntu Server) Esegui comando batch (Windows Server), quindi inserisci quanto segue:

```
rake
```

> **Note**
>
> Verifica che il tuo ambiente sia configurato con le variabili e le impostazioni richieste per eseguire rake; in caso contrario, la compilazione non andrà a buon fine.

14. Scegli Aggiungi azione post-compilazione, quindi scegli AWS CodePipeline Publisher. Scegliere Add (Aggiungi) e in Build Output Locations (Posizioni di output della compilazione), lasciare vuota la posizione. Questa è la configurazione predefinita. Al termine del processo di compilazione verrà creato un file compresso.
15. Scegliere Save (Salva) per salvare il progetto Jenkins.

Passaggio 2: crea una pipeline in CodePipeline

In questa parte del tutorial viene creata una pipeline utilizzando la proceduta guidata Create Pipeline (Crea pipeline).

Per creare un CodePipeline processo di rilascio automatico

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Se necessario, utilizzare il selettore delle regioni per modificare la regione in quella in cui si trovano le risorse della pipeline. Ad esempio, se hai creato risorse per il tutorial precedente in `us-east-2`, assicurati che il selettore della regione sia impostato su Stati Uniti orientali (Ohio).

   Per ulteriori informazioni sulle regioni e gli endpoint disponibili CodePipeline, consulta AWS CodePipeline endpoint e quote.
3. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegliere Create pipeline (Crea pipeline).
4. Nella pagina Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere il nome della pipeline.
5. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
6. In Ruolo di servizio, scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.

7. Lascia i valori predefiniti delle impostazioni nella pagina Advanced settings (Impostazioni avanzate) e scegli Next (Successivo).
8. Nella pagina Passaggio 2: Aggiungi fase di origine, in Provider di origine, scegli GitHub.
9. In Connessione, scegli una connessione esistente o creane una nuova. Per creare o gestire una connessione per l'azione GitHub sorgente, consultaGitHub connessioni.
10. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione), scegliere Add Jenkins (Aggiungi Jenkins). In Nome del provider, inserisci il nome dell'azione che hai fornito nel CodePipeline Plugin per Jenkins (ad esempio *`MyJenkinsProviderName`*). Questo nome deve corrispondere esattamente al nome nel CodePipeline Plugin for Jenkins. In Server URL (URL del server), immettere l'URL dell'istanza EC2 in cui è installato Jenkins. In Nome progetto, inserisci il nome del progetto che hai creato in Jenkins, ad esempio ***`MyDemoProject`***, e quindi scegli Avanti.
11. Nel passaggio 4: aggiungi la fase di distribuzione, riutilizza l' CodeDeploy applicazione e il gruppo di distribuzione in cui hai creato. Tutorial: creazione di una semplice pipeline (bucket S3) In Deploy provider (Provider di distribuzione), scegliere CodeDeploy. In Application name (Nome applicazione), immettere **`CodePipelineDemoApplication`** oppure scegliere il pulsante di aggiornamento e quindi selezionare il nome dell'applicazione dall'elenco. In Deployment group (Gruppo di distribuzione), immettere **`CodePipelineDemoFleet`** o selezionarlo dall'elenco, quindi scegliere Next (Successivo).

    > **Note**
    >
    > Puoi utilizzare CodeDeploy le tue risorse o crearne di nuove, ma potresti incorrere in costi aggiuntivi.

12. In Step 5: Review (Fase 5: revisione), esaminare le informazioni e quindi scegliere Create pipeline (Crea pipeline).
13. La pipeline viene avviata automaticamente ed esegue il codice di esempio. Puoi visualizzare i messaggi di avanzamento e di successo e di fallimento mentre la pipeline crea l'esempio Haml in HTML e lo distribuisce su una pagina Web su ciascuna istanza Amazon EC2 nella distribuzione. CodeDeploy

Fase 3: aggiunta di un'altra fase alla pipeline

Ora aggiungerai una fase di test e un'operazione di test alla fase che utilizza il test Jenkins incluso nel codice di esempio per stabilire se la pagina Web è dotata di contenuti. Il test è solo a scopo dimostrativo.

> **Note**
>
> Se non volevi aggiungere un'altra fase alla pipeline, avresti potuto aggiungere un'operazione di test alla fase temporanea della pipeline, prima o dopo l'operazione di distribuzione.

Aggiunta di una fase di test alla pipeline

Argomenti

- Ricerca dell'indirizzo IP di un'istanza
- Creazione di un progetto Jenkins per testare la distribuzione
- Creazione di una quarta fase

Ricerca dell'indirizzo IP di un'istanza

Per verificare l'indirizzo IP di un'istanza in cui hai distribuito il codice

1. Quando lo stato della pipeline indica Succeeded (Riuscito), nell'area dello stato della fase Staging (Gestione temporanea) scegliere Details (Dettagli).
2. Nella sezione Deployment Details (Dettagli distribuzione), in Instance ID (ID istanza), scegliere l'ID di una delle istanze correttamente distribuite.
3. Copiare l'indirizzo IP dell'istanza (ad esempio `192.168.0.4`). Questo indirizzo IP verrà utilizzato nel test Jenkins.

Creazione di un progetto Jenkins per testare la distribuzione

Per creare il progetto Jenkins

1. Nell'istanza in cui è installato Jenkins, aprire Jenkins e dalla pagina principale scegliere New Item (Nuova voce).

2. In Item Name, inserisci un nome per il progetto Jenkins (ad esempio,). *MyTestProject* Scegliere Freestyle project (Progetto freestyle), quindi OK.

   > **Note**
   >
   > Assicurati che il nome del progetto soddisfi i CodePipeline requisiti. Per ulteriori informazioni, consulta Quote in AWS CodePipeline.

3. Nella pagina di configurazione del progetto, selezionare la casella di controllo Execute concurrent builds if necessary (Esegui compilazioni simultanee in caso di necessità). In Source Code Management (Gestione codice sorgente), scegli AWS CodePipeline. Se hai installato Jenkins su un'istanza EC2 e l'hai configurata AWS CLI con il profilo per l'utente IAM che hai creato per l'integrazione tra CodePipeline e Jenkins, lascia tutti gli altri campi vuoti.

   > **Important**
   >
   > Se stai configurando un progetto Jenkins che non è installato su un'istanza Amazon EC2 o è installato su un'istanza EC2 che esegue un sistema operativo Windows, completa i campi come richiesto dalle impostazioni dell'host e della porta del proxy e fornisci le credenziali dell'utente o del ruolo IAM che hai configurato per l'integrazione tra Jenkins e. CodePipeline

4. Scegliere Advanced (Avanzate) e in Category (Categoria), scegliere Test.

5. In Provider, inserisci lo stesso nome che hai usato per il progetto di compilazione (ad esempio,). *MyJenkinsProviderName* Questo nome verrà utilizzato più avanti nel tutorial per aggiungere l'operazione di test alla pipeline.

   > **Note**
   >
   > Questo nome deve soddisfare i requisiti di CodePipeline denominazione per le azioni. Per ulteriori informazioni, consulta Quote in AWS CodePipeline.

6. In Build Triggers (Crea trigger), deselezionare tutte le caselle di controllo, quindi selezionare Poll SCM (Polling SCM). In Schedule (Pianificazione), immettere cinque asterischi separati da spazi, nel seguente modo:

   ```
   * * * * *
   ```

Questo sondaggio viene fatto CodePipeline ogni minuto.

7. In Build (Compila), scegliere Add build step (Aggiungi fase di compilazione). Se esegui la distribuzione su istanze Amazon Linux, RHEL o Ubuntu Server, scegli Execute shell. Immetti quindi quanto segue, dove l'indirizzo IP è l'indirizzo dell'istanza EC2 copiata in precedenza:

   ```
   TEST_IP_ADDRESS=192.168.0.4 rake test
   ```

   Se esegui la distribuzione su istanze di Windows Server, scegli il comando Esegui batch, quindi inserisci quanto segue, dove l'indirizzo IP è l'indirizzo dell'istanza EC2 che hai copiato in precedenza:

   ```
   set TEST_IP_ADDRESS=192.168.0.4 rake test
   ```

   > **Note**
   >
   > Il test presuppone la porta predefinita 80. Se vuoi specificare una porta diversa, aggiungi un'istruzione di test della porta, come segue:
   >
   > ```
   > TEST_IP_ADDRESS=192.168.0.4 TEST_PORT=8000 rake test
   > ```

8. Scegli Aggiungi azione post-compilazione, quindi scegli Publisher.AWS CodePipeline Non scegliere Add (Aggiungi).
9. Scegliere Save (Salva) per salvare il progetto Jenkins.

Creazione di una quarta fase

Per aggiungere una fase alla pipeline che include l'operazione di test Jenkins

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. In Nome, scegli il nome della pipeline che hai creato, MySecondPipeline.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. Nella pagina Edit (Modifica) scegliere + Stage (+ Fase) per aggiungere una fase immediatamente dopo quella di compilazione.

5. Nel campo del nome della nuova fase, immetti un nome (ad esempio **`Testing`**), quindi scegli + Add Action (+ Aggiungi operazione).
6. In Nome azione, immettete *`MyJenkinsTest-Action`*. In Test provider, scegli il nome del provider specificato in Jenkins (ad esempio, *`MyJenkinsProviderName`*). In Nome progetto, inserisci il nome del progetto che hai creato in Jenkins (ad esempio, *`MyTestProject`*). In Input artifacts, scegli l'artefatto dalla build Jenkins il cui nome predefinito è ***`BuildArtifact`***, quindi scegli Fine.

   > **Note**
   >
   > Poiché l'operazione di test Jenkins funziona sull'applicazione creata nella fase di compilazione di Jenkins, utilizza l'artefatto di compilazione per l'artefatto di input all'operazione di test.

   Per ulteriori informazioni sugli artefatti di input e di output e sulla struttura delle pipeline, consulta CodePipeline riferimento alla struttura della tubazione.
7. Nella pagina Edit (Modifica), scegliere Save pipeline changes (Salva le modifiche alla pipeline) Nella casella di dialogo Save pipeline changes (Salva modifiche della pipeline), scegliere Save and continue (Salva e continua).
8. Anche se la nuova fase è stata aggiunta alla pipeline, viene visualizzato lo stato No executions yet (Ancora nessun esecuzione) per la nuova fase, perché le modifiche non hanno attivato un'altra esecuzione della pipeline. Per eseguire l'esempio nella pipeline rivista, nella pagina dei dettagli della pipeline, scegliete Release change.

   La visualizzazione della pipeline mostra le fasi e le operazioni nella pipeline e lo stato della revisione attraverso le quattro fasi. Il tempo necessario per l'esecuzione di tutte le fasi della pipeline dipende dalle dimensioni degli artefatti, dalla complessità delle operazioni di compilazione e test e da altri fattori.

Fase 4: Eliminazione delle risorse

Dopo aver completato questo tutorial, devi eliminare la pipeline e le risorse utilizzate per evitare che ti venga addebitato un costo per l'utilizzo continuo di tali risorse. Se non intendi continuare a utilizzare CodePipeline, elimina la pipeline, quindi l' CodeDeploy applicazione e le istanze Amazon

EC2 associate e infine il bucket Amazon S3 utilizzato per archiviare gli artefatti. Dovresti anche valutare se eliminare altre risorse, come il GitHub repository, se non intendi continuare a utilizzarle.

Per eliminare le risorse utilizzate in questo tutorial

1. Apri una sessione terminale sul tuo computer Linux, macOS o Unix locale o un prompt dei comandi sul tuo computer Windows locale ed esegui il delete-pipeline comando per eliminare la pipeline che hai creato. Per **`MySecondPipeline`**, immettere il comando seguente:

   ```
   aws codepipeline delete-pipeline --name "MySecondPipeline"
   ```

   Questo comando non restituisce alcun risultato.
2. Per ripulire CodeDeploy le risorse, segui le istruzioni in Pulizia.
3. Per eliminare le risorse dell'istanza, eliminare l'istanza EC2 in cui è installato Jenkins. Per ulteriori informazioni, consulta Pulizia di un'istanza.
4. Se non intendi creare altre pipeline o CodePipeline riutilizzarle, elimina il bucket Amazon S3 usato per archiviare gli elementi per la tua pipeline. Per eliminare il bucket, segui le istruzioni relative all'eliminazione di un bucket.
5. Se non si desidera riutilizzare le altre risorse per questa pipeline, è consigliabile eliminarle seguendo le istruzioni per quella particolare risorsa. Ad esempio, se desideri eliminare il repository, segui le istruzioni in Eliminazione di un GitHub repository sul sito Web. GitHub

Tutorial: imposta una regola CloudWatch Events per ricevere notifiche e-mail per le modifiche allo stato della pipeline

Dopo aver configurato una pipeline in AWS CodePipeline, puoi impostare una regola CloudWatch Events per inviare notifiche ogni volta che ci sono modifiche allo stato di esecuzione delle tue pipeline o nelle fasi o nelle azioni nelle tue pipeline. Per ulteriori informazioni sull'utilizzo CloudWatch degli eventi per impostare le notifiche per le modifiche dello stato della pipeline, consulta. Monitoraggio CodePipeline degli eventi

In questo tutorial, puoi impostare una notifica per l'invio di un'e-mail quando lo stato di una pipeline diventa FAILED (NON RIUSCITO). Questo tutorial utilizza un metodo di trasformazione di input durante la creazione della regola CloudWatch Events. Trasforma i dettagli dello schema del messaggio per consegnare il messaggio in testo leggibile.

> **Note**
>
> Mentre crei le risorse per questo tutorial, come la notifica Amazon SNS e la regola CloudWatch Events, assicurati che le risorse vengano create nella stessa AWS regione della pipeline.

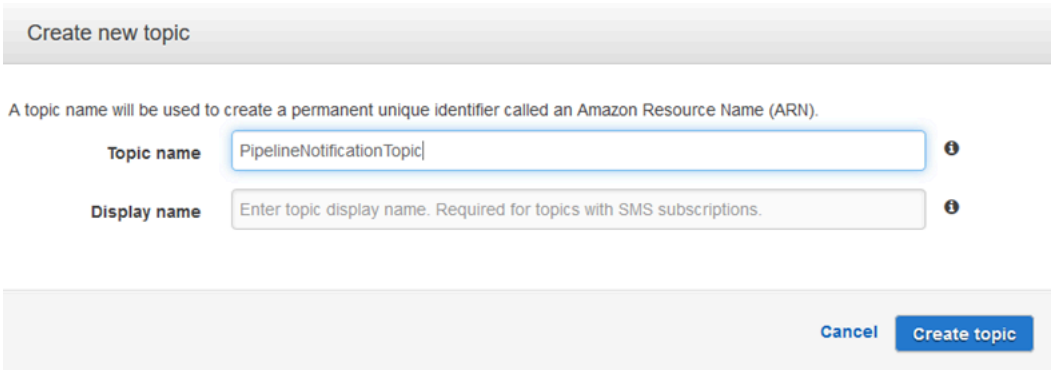
Argomenti

Fase 1: configurare una notifica e-mail utilizzando Amazon SNS

Amazon SNS coordina l'uso degli argomenti per recapitare messaggi agli endpoint o ai clienti abbonati. Usa Amazon SNS per creare un argomento di notifica e poi iscriviti all'argomento utilizzando il tuo indirizzo e-mail. L'argomento Amazon SNS verrà aggiunto come obiettivo alla tua regola CloudWatch Events. Per ulteriori informazioni, consulta la Guida per gli sviluppatori di Amazon Simple Notification Service.

Crea o identifica un argomento in Amazon SNS. CodePipeline utilizzerà CloudWatch Events per inviare notifiche su questo argomento tramite Amazon SNS. Per creare un argomento:

1. Apri la console Amazon SNS all'indirizzo https://console.aws.amazon.com/sns.
2. Scegli Create topic (Crea argomento).
3. Nella finestra di dialogo Create new topic (Crea nuovo argomento) per Topic name (Nome argomento), digitare un nome per l'argomento, ad esempio **`PipelineNotificationTopic`**.



4. Scegli Create topic (Crea argomento).

Per ulteriori informazioni, consulta Create a Topic nella Amazon SNS Developer Guide.

Abbonare uno o più destinatari all'argomento per ricevere notifiche e-mail. Per sottoscrivere un destinatario a un argomento:

1. Nella console Amazon SNS, dall'elenco Argomenti, seleziona la casella di controllo accanto al nuovo argomento. Scegliere Actions, Subscribe to topic (Operazioni, Effettua sottoscrizione all'argomento).
2. Nella finestra di dialogo Create subscription (Crea sottoscrizione), verificare che un ARN sia presente in Topic ARN (ARN dell'argomento).
3. Per Protocollo, scegli E-mail.
4. Per Endpoint (Endpoint), digitare l'indirizzo e-mail completo del destinatario.
5. Scegli Create Subscription (Crea sottoscrizione).
6. Amazon SNS invia un'e-mail di conferma dell'abbonamento al destinatario. Il destinatario deve scegliere il collegamento Confirm subscription (Conferma sottoscrizione) nell'email di conferma per iniziare a ricevere le notifiche. Dopo che il destinatario ha fatto clic sul link, se l'iscrizione è avvenuta correttamente, Amazon SNS visualizza un messaggio di conferma nel browser Web del destinatario.

   Per ulteriori informazioni, consulta Abbonarsi a un argomento nella Amazon SNS Developer Guide.

Fase 2: creazione di una regola e aggiunta dell'argomento SNS come destinazione

Crea una regola di notifica CloudWatch degli eventi con CodePipeline come origine dell'evento.

1. Apri la CloudWatch console all'indirizzo https://console.aws.amazon.com/cloudwatch/.
2. Nel pannello di navigazione seleziona Events (Eventi).
3. Scegli Crea regola. In Event source (Origine eventi), scegli AWS CodePipeline. Per Event Type (Tipo di evento), scegliere Pipeline Execution State Change (Modifica dello stato di esecuzione della pipeline).
4. Selezionare Specific state(s) (Stati specifici), quindi scegliere **`FAILED`**.

5. Scegliere Edit (Modifica) per aprire l'editor JSON per il riquadro Event Pattern Preview (Anteprima modello di eventi). Aggiungere il parametro **`pipeline`** con il nome della pipeline come nell'esempio seguente per la pipeline denominata "myPipeline."

   Puoi copiare il modello di evento qui e incollarlo nella console:

   ```
   {
     "source": [
       "aws.codepipeline"
     ],
     "detail-type": [
       "CodePipeline Pipeline Execution State Change"
     ],
     "detail": {
       "state": [
         "FAILED"
       ],
       "pipeline": [
         "myPipeline"
       ]
     }
   }
   ```

6. In Targets (Destinazioni), seleziona Add target (Aggiungi destinazione).
7. Nell'elenco degli obiettivi, selezionare SNS topic (Argomento SNS); Per Topic (Argomento), immettere l'argomento creato.
8. Espandere Configure input (Configura input), quindi selezionare Input Transformer (Trasformatore di input).
9. Nella casella Input Path (Percorso input) digitare le seguenti coppie chiave-valore.

   ```
   { "pipeline" : "$.detail.pipeline" }
   ```

   Nella casella Input Template (Modello di input) digitare quanto segue:

   ```
   "The Pipeline <pipeline> has failed."
   ```

10. Scegli Configura dettagli.
11. Nella pagina Configure rule details (Configura i dettagli della regola), digitare un nome e una descrizione facoltativa. Lasciare la casella Enabled (Abilitato) selezionata per State (Stato).

12. Scegli Crea regola.
13. Conferma che ora CodePipeline sta inviando le notifiche di build. Ad esempio, verificare se le e-mail di notifica di compilazione si trovano nella casella di posta.
14. Per modificare il comportamento di una regola, nella CloudWatch console, scegli la regola, quindi scegli Azioni, Modifica. Modificare la regola, scegliere Configure details (Configura dettagli) e quindi selezionare Update rule (Aggiorna regola).

    Per smettere di usare una regola per inviare notifiche di build, nella CloudWatch console, scegli la regola, quindi scegli Azioni, Disabilita.

    Per eliminare una regola, nella CloudWatch console, scegli la regola, quindi scegli Azioni, Elimina.

Fase 3: Eliminazione delle risorse

Dopo aver completato questo tutorial, devi eliminare la pipeline e le risorse utilizzate per evitare che ti venga addebitato un costo per l'utilizzo continuo di tali risorse.

Per informazioni su come ripulire la notifica SNS ed eliminare la regola Amazon CloudWatch Events, consulta Clean Up (Unsubscribe from an Amazon SNS Topic) e consulta il riferimento `DeleteRule` nell'Amazon Events API Reference CloudWatch .

Tutorial: crea una pipeline con cui creare e testare la tua app Android AWS Device Farm

Puoi utilizzarla AWS CodePipeline per configurare un flusso di integrazione continuo in cui la tua app viene creata e testata ogni volta che viene inviato un commit. Questo tutorial mostra come creare e configurare una pipeline per creare e testare la tua app Android con codice sorgente in un GitHub repository. La pipeline rileva l'arrivo di un nuovo GitHub commit e quindi la utilizza CodeBuildper creare l'app e Device Farm per testarlo.

> **Important**
>
> Molte delle azioni che aggiungi alla pipeline in questa procedura coinvolgono AWS risorse che devi creare prima di creare la pipeline. AWS le risorse per le tue azioni di origine devono sempre essere create nella stessa AWS regione in cui crei la pipeline. Ad esempio, se crei la

pipeline nella regione Stati Uniti orientali (Ohio), il tuo CodeCommit repository deve trovarsi nella regione Stati Uniti orientali (Ohio).
Puoi aggiungere azioni interregionali quando crei la pipeline. AWS le risorse per le azioni interregionali devono trovarsi nella stessa AWS regione in cui intendi eseguire l'azione. Per ulteriori informazioni, consulta Aggiungere un'azione interregionale in CodePipeline.

Puoi provarlo utilizzando l'app Android esistente e le definizioni di test oppure puoi utilizzare l'app di esempio e le definizioni di test fornite da Device Farm.

Note

Prima di iniziare

1. Accedi alla AWS Device Farm console e scegli Crea un nuovo progetto.
2. Scegliere il progetto. Nel browser, copiare l'URL del nuovo progetto. L'URL contiene l'ID del progetto.
3. Copiare e conservare questo ID del progetto. Lo usi quando crei la tua pipeline in CodePipeline.

   Ecco un URL di esempio per un progetto. Per estrarre l'ID del progetto, copiare il valore dopo `projects/`. In questo esempio l'ID del prodotto è `eec4905f-98f8-40aa-9afc-4c1cfexample`.

   ```
   https://<region-URL>/devicefarm/home?region=us-west-2#/projects/
   eec4905f-98f8-40aa-9afc-4c1cfexample/runs
   ```

Configura CodePipeline per utilizzare i test Device Farm

1. Aggiungi e esegui il commit di un file chiamato `buildspec.yml` nella radice del codice dell'app e invialo al tuo repository. CodeBuild utilizza questo file per eseguire comandi e accedere agli artefatti necessari per creare l'app.

   ```
   version: 0.2

   phases:
   ```

```
  build:
    commands:
      - chmod +x ./gradlew
      - ./gradlew assembleDebug
artifacts:
  files:
     - './android/app/build/outputs/**/*.apk'
  discard-paths: yes
```

2. (Facoltativo) Se utilizzi Calabash o Appium per testare la tua applicazione, aggiungi il file di definizione del test al repository. In una fase successiva, puoi configurare Device Farm per utilizzare le definizioni per eseguire la tua suite di test.

   Se utilizzi i test integrati di Device Farm, puoi saltare questo passaggio.

3. Per creare la pipeline e aggiungere una fase di origine, procedi nel seguente modo:

   a. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

   b. Scegliere Create pipeline (Crea pipeline). Nella pagina Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere il nome della pipeline.

   c. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.

   d. Alla voce Service role (Ruolo del servizio), lasciare selezionato New service role (Nuovo ruolo del servizio) e lasciare immutata la voce Role name (Nome ruolo). È anche possibile scegliere di utilizzare un ruolo di servizio esistente, se disponibile.

   > **Note**
   >
   > Se utilizzi un ruolo CodePipeline di servizio creato prima di luglio 2018, devi aggiungere le autorizzazioni per Device Farm. A tale scopo, apri la console IAM, trova il ruolo, quindi aggiungi le seguenti autorizzazioni alla policy del ruolo. Per ulteriori informazioni, consulta Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline.
   >
   > ```
   > {
   >      "Effect": "Allow",
   >      "Action": [
   > ```

```
            "devicefarm:ListProjects",
            "devicefarm:ListDevicePools",
            "devicefarm:GetRun",
            "devicefarm:GetUpload",
            "devicefarm:CreateUpload",
            "devicefarm:ScheduleRun"
        ],
        "Resource": "*"
}
```

e. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).

f. Nella pagina Step 2: Aggiungi fase di origine, in Source provider, scegli GitHub.

g. In Connessione, scegli una connessione esistente o creane una nuova. Per creare o gestire una connessione per l'azione GitHub sorgente, consultaGitHub connessioni.

h. In Repository, scegliere il repository di origine.

i. In Branch (Ramificazione), scegliere la ramificazione che si desidera utilizzare.

j. Lascia le impostazioni predefinite rimanenti per l'azione di origine. Seleziona Next (Successivo).

4. In Add build stage (Aggiungi fase di compilazione), aggiungi una fase di compilazione:

a. In Build provider (Provider compilazione), scegli AWS CodeBuild. Consenti a Region (Regione) di preimpostarsi sulla regione della pipeline.

b. Seleziona Crea progetto.

c. In Project name (Nome progetto) immettere un nome per questo progetto di compilazione.

d. In Environment image (Immagine ambiente), scegli Managed image (Immagine gestita). In Operating system (Sistema operativo), seleziona Ubuntu.

e. In Runtime, seleziona Standard. Per Immagine, scegli aws/codebuild/standard:5.0.

CodeBuild utilizza questa immagine del sistema operativo, su cui è installato Android Studio, per creare l'app.

f. Per Ruolo di servizio, scegli il ruolo CodeBuild di servizio esistente o creane uno nuovo.

g. Per Build specifications (Compila specifiche), scegli Use a buildspec file (Usa un file buildspec).

h. Scegli Continua a CodePipeline. Questo ritorna alla CodePipeline console e crea un CodeBuild progetto che utilizza il file `buildspec.yml` nel tuo repository per la

configurazione. Il progetto di compilazione utilizza un ruolo di servizio per gestire le Servizio AWS autorizzazioni. Questa operazione potrebbe richiedere un paio di minuti.

i. Seleziona Next (Successivo).

5. Nella pagina Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione), scegli Skip deploy stage (Ignora fase di distribuzione), quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Next (Successivo).
6. Nella Step 5: Review (Fase 5: revisione), scegliere Create pipeline (Crea pipeline). Dovresti visualizzare uno schema che mostra le fasi di origine e compilazione.
7. Aggiungi un'azione di test di Device Farm alla tua pipeline:

   a. In alto a destra, scegli Edit (Modifica).

   b. In fondo al diagramma, scegliere + Add stage (+ Aggiungi fase) In Nome fase immetti un nome, ad esempio **`Test`**.

   c. Scegliere + Add action group (+ Aggiungi gruppo di operazioni).

   d. Alla voce Action name (Nome operazione), inserire un nome.

   e. In Action provider, scegli AWS Device Farm. Consenti a Region (Regione) di preimpostarsi sulla regione della pipeline.

   f. In Input artifacts (Artefatti di input), scegli l'artefatto di input che corrisponde all'artefatto di output della fase precedente alla fase di test, ad esempio `BuildArtifact`.

      Nella AWS CodePipeline console, puoi trovare il nome dell'elemento di output per ogni fase passando il mouse sull'icona delle informazioni nel diagramma della pipeline. Se la pipeline verifica l'app direttamente dalla fase Source, scegli. SourceArtifact Se la pipeline include una fase di creazione, scegli. BuildArtifact

   g. Nel ProjectId, inserisci l'ID del tuo progetto Device Farm. Utilizzare la procedura all'inizio di questo tutorial per recuperare l'ID del progetto.

   h. In DevicePoolArn, inserisci l'ARN per il pool di dispositivi. Per ottenere gli ARN del pool di dispositivi disponibili per il progetto, incluso l'ARN per i dispositivi principali, utilizza la AWS CLI per immettere il seguente comando:

      ```
      aws devicefarm list-device-pools --arn arn:aws:devicefarm:us-
      west-2:account_ID:project:project_ID
      ```

   i. In, inserisci AppTypeAndroid.

      Di seguito è riportato un elenco di valori validi per AppType:

- iOS
- Android
- App

j. In App, inserisci il percorso del pacchetto dell'applicazione compilata. Il percorso è relativo alla cartella principale dell'artefatto di input della fase di sviluppo. Di solito, questo percorso è simile a `app-release.apk`.

k. In TestType, inserisci il tipo di test, quindi in Test, inserisci il percorso del file di definizione del test. Il percorso è relativo alla cartella principale dell'artefatto di input del test.

Di seguito è riportato un elenco di valori validi per TestType:

- APPIUM_JAVA_JUNIT
- APPIUM_JAVA_TESTNG
- NODO APPIUM
- APPIUM_RUBY
- APPIUM_PYTHON
- APPIUM_WEB_JAVA JUNIT
- APPIUM_WEB_JAVA_TESTNG
- NODO APPIUM_WEB
- APPIUM_WEB_RUBY
- APPIUM_WEB_PYTHON
- FUZZ INTEGRATO
- INSTRUMENTATION
- XCTEST
- XCTEST_UI

> **Note**
>
> I nodi di ambiente personalizzati non sono supportati.

l. Nei campi rimanenti, inserire la configurazione appropriata per il test e il tipo di applicazione.

m. (Facoltativo) IN Advanced (Avanzate), fornire informazioni di configurazione per la sessione di test.

n. Selezionare Salva.

o. Nella fase che stai modificando, scegli Done (Fatto). Nel riquadro AWS CodePipeline , scegli Save (Salva) e quindi scegli Save (Salva) sul messaggio di avviso.

p. Per inviare le modifiche e avviare una compilazione tramite pipeline, scegliere Release change (Rilascia modifica) e quindi scegliere Release (Rilascia).

Tutorial: crea una pipeline con cui testare la tua app iOS AWS Device Farm

Puoi usarlo AWS CodePipeline per configurare facilmente un flusso di integrazione continuo in cui la tua app viene testata ogni volta che il bucket di origine cambia. Questo tutorial illustra come creare e configurare una pipeline per testare l'app iOS compilata da un bucket S3. La pipeline rileva l'arrivo di una modifica salvata tramite Amazon CloudWatch Events, quindi utilizza Device Farm per testare l'applicazione creata.

> **Important**
>
> Molte delle azioni che aggiungi alla pipeline in questa procedura coinvolgono AWS risorse che devi creare prima di creare la pipeline. AWS le risorse per le tue azioni di origine devono sempre essere create nella stessa AWS regione in cui crei la pipeline. Ad esempio, se crei la pipeline nella regione Stati Uniti orientali (Ohio), il tuo CodeCommit repository deve trovarsi nella regione Stati Uniti orientali (Ohio).
>
> Puoi aggiungere azioni interregionali quando crei la pipeline. AWS le risorse per le azioni interregionali devono trovarsi nella stessa AWS regione in cui intendi eseguire l'azione. Per ulteriori informazioni, consulta Aggiungere un'azione interregionale in CodePipeline.

Puoi sperimentare utilizzando una tua app iOS preesistente oppure puoi utilizzare l'app iOS di esempio.

> **Note**

Prima di iniziare

1. Accedi alla AWS Device Farm console e scegli Crea un nuovo progetto.

2. Scegliere il progetto. Nel browser, copiare l'URL del nuovo progetto. L'URL contiene l'ID del progetto.
3. Copiare e conservare questo ID del progetto. Lo usi quando crei la tua pipeline in CodePipeline.

   Ecco un URL di esempio per un progetto. Per estrarre l'ID del progetto, copiare il valore dopo `projects/`. In questo esempio l'ID del prodotto è `eec4905f-98f8-40aa-9afc-4c1cfexample`.

   ```
   https://<region-URL>/devicefarm/home?region=us-west-2#/projects/
   eec4905f-98f8-40aa-9afc-4c1cfexample/runs
   ```

Configura CodePipeline per utilizzare i test Device Farm (esempio Amazon S3)

1. Creare o utilizzare un bucket S3 con la funzione versioni multiple attivata. Per creare un bucket S3, segui le istruzioni contenute in Fase 1: creazione di un bucket S3 per l'applicazione.
2. Nella console Amazon S3 del tuo bucket, scegli Carica e segui le istruzioni per caricare il tuo file.zip.

   L'applicazione di esempio deve essere compressa in un file .zip.
3. Per creare la pipeline e aggiungere una fase di origine, procedi nel seguente modo:

   a. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

   b. Scegliere Create pipeline (Crea pipeline). Nella pagina Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere il nome della pipeline.

   c. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.

   d. Alla voce Service role (Ruolo del servizio), lasciare selezionato New service role (Nuovo ruolo del servizio) e lasciare immutata la voce Role name (Nome ruolo). È anche possibile scegliere di utilizzare un ruolo di servizio esistente, se disponibile.

> **Note**
>
> Se si utilizza un ruolo CodePipeline di servizio creato prima di luglio 2018, è necessario aggiungere le autorizzazioni per Device Farm. A tale scopo, apri la console IAM, trova il ruolo e aggiungi le seguenti autorizzazioni alla policy del ruolo. Per ulteriori informazioni, consulta Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline.
>
> ```
> {
>      "Effect": "Allow",
>      "Action": [
>         "devicefarm:ListProjects",
>         "devicefarm:ListDevicePools",
>         "devicefarm:GetRun",
>         "devicefarm:GetUpload",
>         "devicefarm:CreateUpload",
>         "devicefarm:ScheduleRun"
>      ],
>      "Resource": "*"
> }
> ```

e. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).

f. Nella pagina Fase 2: aggiunta fase di origine in Source provider (Provider origine) scegliere Amazon S3.

g. Nella posizione Amazon S3, inserisci il bucket, ad esempio, e la chiave oggettomy-`storage-bucket`, ad esempio `s3-ios-test-1.zip` per il tuo file.zip.

h. Seleziona Successivo.

4. Alla voce Build (Compila), creare un segnaposto per la fase di compilazione della pipeline. In questo modo è possibile creare la pipeline nella procedura guidata. Dopo aver utilizzato la procedura guidata per creare la pipeline a due fasi, questo segnaposto per la fase di compilazione non è più necessario. Una volta completata la creazione della pipeline, questa seconda fase viene eliminata e viene aggiunta la nuova fase di test nel passaggio 5.

   a. Alla voce Build provider (Provider di compilazione), scegliere Add Jenkins (Aggiungi Jenkins). Questa selezione per la compilazione è un segnaposto. Non viene utilizzato.

b. Alla voce Provider name (Nome provider), inserire un nome. Il nome è un segnaposto. Non viene utilizzato.

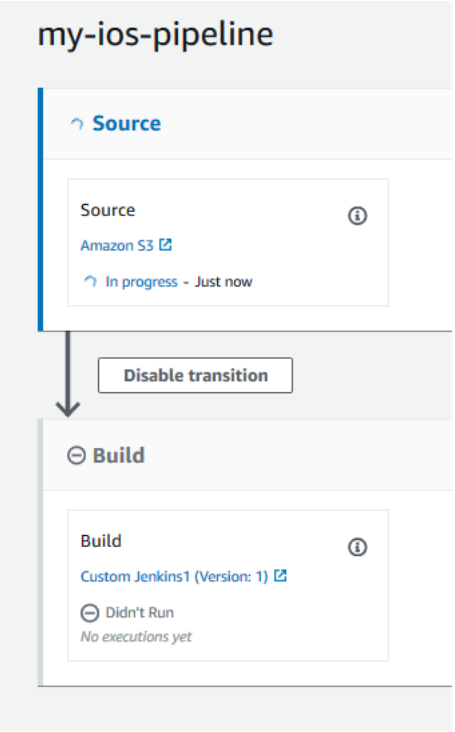
c. Alla voce Server URL (URL server), inserire il testo. Il testo è un segnaposto. Non viene utilizzato.

d. Alla voce Project name (Nome progetto), inserire un nome. Il nome è un segnaposto. Non viene utilizzato.

e. Seleziona Next (Successivo).

f. Nella pagina Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione), scegli Skip deploy stage (Ignora fase di distribuzione), quindi accetta il messaggio di avviso scegliendo Skip (Ignora).

g. Nella Step 5: Review (Fase 5: revisione), scegliere Create pipeline (Crea pipeline). Dovresti visualizzare uno schema che mostra le fasi di origine e compilazione.



5. Aggiungi un'azione di test di Device Farm alla tua pipeline come segue:

a. In alto a destra, scegli Edit (Modifica).

b. Scegli Edit stage (Modifica fase). Scegli Elimina. Ciò elimina il segnaposto della fase ora che non è più necessaria per la creazione della pipeline.

c. In fondo al diagramma, scegliere + Add stage (+ Aggiungi fase)

d. In "Nome fase", immetti un nome per la fase, ad esempio Test, quindi scegli Add stage (Aggiungi fase).

e. Scegliere + Add action group (+ Aggiungi gruppo di operazioni).

f. In Nome azione, inserisci un nome, ad esempio DeviceFarmTest.

g. In Action provider, scegli AWS Device Farm. Consenti a Region (Regione) di preimpostarsi sulla regione della pipeline.

h. In Input artifacts (Artefatti di input), scegli l'artefatto di input che corrisponde all'artefatto di output della fase precedente alla fase di test, ad esempio `SourceArtifact`.

   Nella AWS CodePipeline console, puoi trovare il nome dell'elemento di output per ogni fase passando il mouse sull'icona delle informazioni nel diagramma della pipeline. Se la tua pipeline testa la tua app direttamente dalla fase Source, scegli. SourceArtifact Se la pipeline include una fase di creazione, scegli. BuildArtifact

i. Nel ProjectId, scegli l'ID del tuo progetto Device Farm. Utilizzare la procedura all'inizio di questo tutorial per recuperare l'ID del progetto.

j. In DevicePoolArn, inserisci l'ARN per il pool di dispositivi. Per ottenere gli ARN del pool di dispositivi disponibili per il progetto, incluso l'ARN per i dispositivi principali, utilizza la AWS CLI per immettere il seguente comando:

```
aws devicefarm list-device-pools --arn arn:aws:devicefarm:us-
west-2:account_ID:project:project_ID
```

k. Nel AppType, inserisci iOS.

   Di seguito è riportato un elenco di valori validi per AppType:

   - iOS
   - Android
   - App

l. In App, inserisci il percorso del pacchetto dell'applicazione compilata. Il percorso è relativo alla cartella principale dell'artefatto di input della fase di sviluppo. Di solito, questo percorso è simile a `ios-test.ipa`.

m. In TestType, inserisci il tipo di test, quindi in Test, inserisci il percorso del file di definizione del test. Il percorso è relativo alla cartella principale dell'artefatto di input del test.

Se utilizzi uno dei test Device Farm integrati, inserisci il tipo di test configurato nel tuo progetto Device Farm, ad esempio BUILTIN_FUZZ. In FuzzEventCount, inserisci un tempo in millisecondi, ad esempio 6000. In FuzzEventThrottle, immettete un tempo in millisecondi, ad esempio 50.

Se non stai utilizzando uno dei test Device Farm integrati, inserisci il tipo di test, quindi in Test inserisci il percorso del file di definizione del test. Il percorso è relativo alla cartella principale dell'artefatto di input del test.

Di seguito è riportato un elenco di valori validi per TestType:

- APPIUM_JAVA_JUNIT
- APPIUM_JAVA_TESTNG
- NODO_APPIUM
- APPIUM_RUBY
- APPIUM_PYTHON
- APPIUM_WEB_JAVA_JUNIT
- APPIUM_WEB_JAVA_TESTNG
- NODO APPIUM_WEB
- APPIUM_WEB_RUBY
- APPIUM_WEB_PYTHON
- BUILTIN_FUZZ
- INSTRUMENTATION
- XCTEST
- XCTEST_UI

> **Note**
>
> I nodi di ambiente personalizzati non sono supportati.

n. Nei campi rimanenti, inserire la configurazione appropriata per il test e il tipo di applicazione.

o. (Facoltativo) IN Advanced (Avanzate), fornire informazioni di configurazione per la sessione di test.

p. Selezionare Salva.

q. Nella fase che stai modificando, scegli Done (Fatto). Nel AWS CodePipeline riquadro, scegli Salva, quindi scegli Salva nel messaggio di avviso.

r. Per inviare le modifiche e avviare l'esecuzione della pipeline, scegli Release change (Rilascia modifica) e quindi scegli Release (Rilascia).

Tutorial: crea una pipeline da distribuire su Service Catalog

Service Catalog consente di creare e fornire prodotti basati su AWS CloudFormation modelli. Questo tutorial mostra come creare e configurare una pipeline per distribuire il modello di prodotto su Service Catalog e fornire le modifiche apportate nel repository di origine (già creato in GitHub CodeCommit, o Amazon S3).

> **Note**
>
> Se Amazon S3 è il provider di origine per la tua pipeline, devi caricare nel tuo bucket tutti i file sorgente impacchettati come un unico file.zip. In caso contrario, l'azione di origine ha esito negativo.

Innanzitutto, crei un prodotto in Service Catalog, quindi crei una pipeline in AWS CodePipeline. Questo tutorial offre due opzioni per l'impostazione della configurazione della distribuzione:

- Crea un prodotto in Service Catalog e carica un file modello nel tuo repository di origine. Fornisci la versione del prodotto e la configurazione di distribuzione nella CodePipeline console (senza un file di configurazione separato). Per informazioni, consulta Opzione 1: distribuzione su Service Catalog senza un file di configurazione.

  > **Note**
  >
  > Il file di modello può essere creato in formato JSON o YAML.

- Crea un prodotto in Service Catalog e carica un file modello nel tuo repository di origine. Fornisci la versione del prodotto e la configurazione della distribuzione nella console (senza un file di configurazione separato). Per informazioni, consulta Opzione 2: eseguire la distribuzione su Service Catalog utilizzando un file di configurazione.

Opzione 1: distribuzione su Service Catalog senza un file di configurazione

In questo esempio, carichi il file AWS CloudFormation modello di esempio per un bucket S3, quindi crei il prodotto in Service Catalog. Successivamente, crei la pipeline e specifichi la configurazione di distribuzione nella console. CodePipeline

Fase 1: caricamento di file di modello di esempio nel repository di origine

1. Aprire un editor di testo. Creare un modello di esempio incollando quanto segue nel file. Salva il file con nome `S3_template.json`.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "CloudFormation Sample Template S3_Bucket: Sample template showing
 how to create a privately accessible S3 bucket. **WARNING** This template creates
 an S3 bucket. You will be billed for the resources used if you create a stack from
 this template.",
  "Resources": {
    "S3Bucket": {
      "Type": "AWS::S3::Bucket",
      "Properties": {}
    }
  },
  "Outputs": {
    "BucketName": {
      "Value": {
        "Ref": "S3Bucket"
      },
      "Description": "Name of Amazon S3 bucket to hold website content"
    }
  }
}
```

   Questo modello consente di AWS CloudFormation creare un bucket S3 che può essere utilizzato da Service Catalog.

2. Caricare il file `S3_template.json` nel repository AWS CodeCommit .

Fase 2: Creare un prodotto in Service Catalog

1. In qualità di amministratore IT, accedi alla console Service Catalog, vai alla pagina Prodotti, quindi scegli Carica nuovo prodotto.
2. Nella pagina Upload new product (Carica nuovo prodotto), procedere come segue:
   a. In Product name (Nome prodotto), immettere il nome da utilizzare per il nuovo prodotto.
   b. In Description (Descrizione), immettere la descrizione del catalogo dei prodotti. Questa descrizione è visualizzata nell'elenco dei prodotti per consentire all'utente di scegliere il prodotto corretto.
   c. In Provided by (Fornito da), immettere il nome dell'amministratore o del reparto IT.
   d. Seleziona Successivo.
3. (Facoltativo) In Enter support details (Inserisci i dettagli di supporto), immettere le informazioni di contatto per il supporto del prodotto e scegliere Next (Successivo).
4. In Version details (Dettagli versione), procedere come segue:
   a. Scegliere Upload a template file (Carica un file di modello). Individuare il file `S3_template.json` e caricarlo.
   b. In Version title (Titolo versione), immettere il nome della versione del prodotto (ad esempio **`devops S3 v2`**).
   c. In Description (Descrizione), immettere i dettagli che distinguono questa versione dalle altre versioni.
   d. Seleziona Successivo.
5. Nella pagina Review (Verifica), verificare che le informazioni siano corrette, quindi scegliere Create (Crea).
6. Nella pagina Products (Prodotti), copiare nel browser l'URL del nuovo prodotto. L'URL contiene l'ID del prodotto. Copiare e conservare questo ID del prodotto. Lo usi quando crei la tua pipeline in CodePipeline.

   Ecco l'URL per un prodotto denominato `my-product`. Per estrarre l'ID del prodotto, copiare il valore compreso tra il segno di uguale (=) e la E commerciale (&). In questo esempio l'ID del prodotto è `prod-example123456`.

```
https://<region-URL>/servicecatalog/home?region=<region>#/admin-products?
productCreated=prod-example123456&createdProductTitle=my-product
```

> **Note**
>
> Copiare l'URL per il prodotto prima di uscire da questa pagina. Una volta che si esce da questa pagina, è necessario utilizzare l'interfaccia a riga di comando per ottenere l'ID prodotto.

Dopo alcuni secondi, il prodotto viene visualizzato nella pagina Products (Prodotti). È possibile che sia necessario aggiornare il browser per visualizzare il prodotto nell'elenco.

Fase 3: creazione della pipeline

1. Per assegnare un nome alla pipeline e selezionare i parametri per la pipeline, procedere nel seguente modo:
   a. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
   b. Selezionare Getting started (Nozioni di base). Scegliere Create pipeline (Crea pipeline) e immettere un nome per la pipeline.
   c. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
   d. In Ruolo di servizio, scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.
   e. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).
2. Per aggiungere una fase di origine, procedere come segue:
   a. In Source provider (Provider origine), scegliere AWS CodeCommit.
   b. In Repository name (Nome del repository) e Branch name (Nome ramo), immettere il repository e il ramo da utilizzare per l'operazione di origine.
   c. Seleziona Successivo.
3. In Add build stage (Aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora).
4. In Aggiunta della fase di distribuzione, procedere come segue:

a. In Deploy provider (Provider di distribuzione), scegliere AWS Service Catalog.

b. Per la configurazione della distribuzione, scegli Enter deployment configuration (Inserisci configurazione distribuzione).

c. In Product ID, incolla l'ID del prodotto che hai copiato dalla console Service Catalog.

d. In Template file path (Percorso file di modello), immettere il percorso relativo dove è archiviato il file di modello.

e. In Tipo di prodotto, scegli AWS CloudFormation modello.

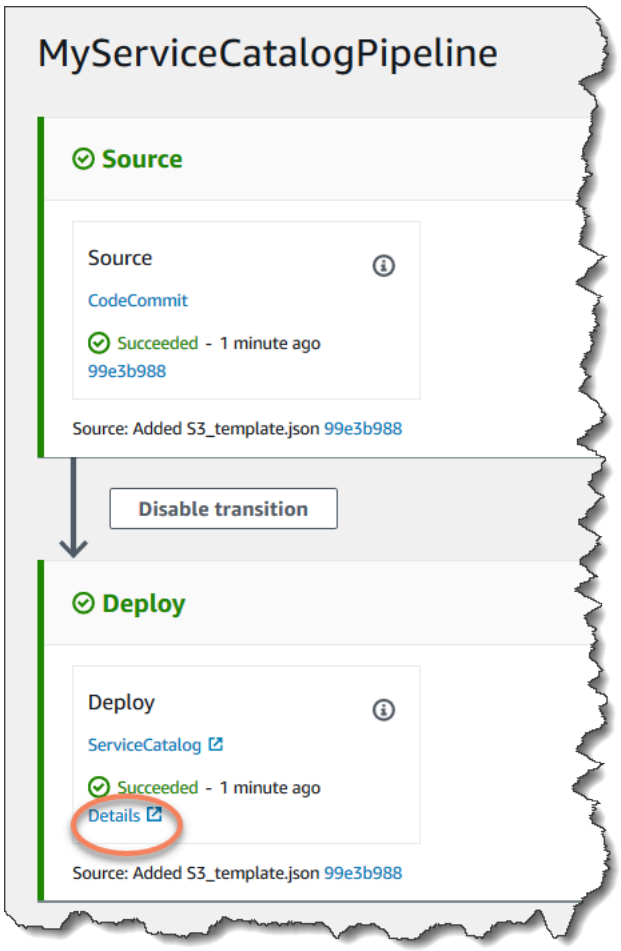
f. Nel Nome della versione del prodotto, inserisci il nome della versione del prodotto specificata in Service Catalog. Se si desidera che la modifica del modello venga distribuita in una nuova versione del prodotto, immettere un nome della versione del prodotto che non è stato utilizzato per una versione precedente dello stesso prodotto.

g. Per Input artifact (Artefatto di input), scegliere l'artefatto di origine di input.

h. Seleziona Successivo.

5. In Review (Verifica), esaminare le impostazioni della pipeline e selezionare Create (Crea).

6. Dopo che la pipeline è in esecuzione, nella fase di distribuzione, scegliere Details (Dettagli). Verrà aperto il prodotto in Service Catalog.



7. Nelle informazioni sul prodotto, scegliere il nome della versione per aprire il modello di prodotto. Visualizzare la distribuzione del modello.

Passaggio 4: invia una modifica e verifica il prodotto in Service Catalog

1. Visualizza la pipeline nella CodePipeline console e, nella fase di origine, scegli Dettagli. Il tuo AWS CodeCommit repository di origine si apre nella console. Scegliere Edit (Modifica) ed effettuare una modifica nel file (ad esempio la descrizione).

```
"Description": "Name of Amazon S3 bucket to hold and version website content"
```

2. Eseguire il commit e il push della modifica. La pipeline viene avviata dopo il push della modifica. Una volta completata l'esecuzione della pipeline, nella fase di implementazione, scegli Dettagli per aprire il prodotto in Service Catalog.
3. Nelle informazioni sul prodotto, scegliere il nome della nuova versione per aprire il modello di prodotto. Visualizzare la modifica del modello distribuito.

Opzione 2: eseguire la distribuzione su Service Catalog utilizzando un file di configurazione

In questo esempio, carichi il file AWS CloudFormation modello di esempio per un bucket S3, quindi crei il prodotto in Service Catalog. Inoltre carichi un file di configurazione separato che specifica la configurazione della distribuzione. Successivamente, crei la pipeline e specifichi il percorso del file di configurazione.

Fase 1: caricamento di file di modello di esempio nel repository di origine

1. Aprire un editor di testo. Creare un modello di esempio incollando quanto segue nel file. Salva il file con nome `S3_template.json`.

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "CloudFormation Sample Template S3_Bucket: Sample template showing
 how to create a privately accessible S3 bucket. **WARNING** This template creates
 an S3 bucket. You will be billed for the resources used if you create a stack from
 this template.",
  "Resources": {
    "S3Bucket": {
      "Type": "AWS::S3::Bucket",
      "Properties": {}
    }
  },
  "Outputs": {
    "BucketName": {
      "Value": {
        "Ref": "S3Bucket"
      },
      "Description": "Name of Amazon S3 bucket to hold website content"
    }
  }
}
```

   Questo modello consente di AWS CloudFormation creare un bucket S3 che può essere utilizzato da Service Catalog.

2. Caricare il file `S3_template.json` nel repository AWS CodeCommit .

Fase 2: creazione del file di configurazione della distribuzione del prodotto

1. Aprire un editor di testo. Creare il file di configurazione per il prodotto. Il file di configurazione viene utilizzato per definire i parametri/le preferenze di distribuzione del Service Catalog. Questo file viene utilizzato quando si crea la pipeline.

   In questo esempio viene usato il `ProductVersionName` "devops S3 v2" e la `ProductVersionDescription` `MyProductVersionDescription`. Se si desidera che la modifica del modello venga distribuita in una nuova versione del prodotto, è sufficiente immettere un nome della versione del prodotto che non è stato utilizzato per una versione precedente dello stesso prodotto.

   Salva il file con nome `sample_config.json`.

   ```
   {
       "SchemaVersion": "1.0",
       "ProductVersionName": "devops S3 v2",
       "ProductVersionDescription": "MyProductVersionDescription",
       "ProductType": "CLOUD_FORMATION_TEMPLATE",
       "Properties": {
           "TemplateFilePath": "/S3_template.json"
       }
   }
   ```

   Questo file crea le informazioni sulla versione del prodotto per ogni volta che si esegue la pipeline.

2. Caricare il file `sample_config.json` nel repository AWS CodeCommit . Assicurarsi di caricare questo file nel repository di origine.

Fase 3: Creare un prodotto in Service Catalog

1. In qualità di amministratore IT, accedi alla console Service Catalog, vai alla pagina Prodotti, quindi scegli Carica nuovo prodotto.
2. Nella pagina Upload new product (Carica nuovo prodotto), procedere come segue:

   a. In Product name (Nome prodotto), immettere il nome da utilizzare per il nuovo prodotto.

b. In Description (Descrizione), immettere la descrizione del catalogo dei prodotti. Questa descrizione è presente nell'elenco dei prodotti per consentire all'utente di scegliere il prodotto corretto.

c. In Provided by (Fornito da), immettere il nome dell'amministratore o del reparto IT.

d. Seleziona Successivo.

3. (Facoltativo) In Enter support details (Inserisci i dettagli di supporto), immettere le informazioni di contatto per il supporto del prodotto e scegliere Next (Successivo).

4. In Version details (Dettagli versione), procedere come segue:

a. Scegliere Upload a template file (Carica un file di modello). Individuare il file `S3_template.json` e caricarlo.

b. In Version title (Titolo versione), inserire il nome della versione del prodotto (ad esempio "devops S3 v2").

c. In Description (Descrizione), immettere i dettagli che distinguono questa versione dalle altre versioni.

d. Seleziona Successivo.

5. Nella pagina Review (Verifica), verificare che le informazioni siano corrette, quindi scegliere Confirm and upload (Conferma e carica).

6. Nella pagina Products (Prodotti), copiare nel browser l'URL del nuovo prodotto. L'URL contiene l'ID del prodotto. Copiare e conservare questo ID del prodotto. Lo usi quando crei la tua pipeline in CodePipeline.

   Ecco l'URL per un prodotto denominato `my-product`. Per estrarre l'ID del prodotto, copiare il valore compreso tra il segno di uguale (=) e la E commerciale (&). In questo esempio l'ID del prodotto è `prod-example123456`.

```
https://<region-URL>/servicecatalog/home?region=<region>#/admin-products?
productCreated=prod-example123456&createdProductTitle=my-product
```

> **Note**
>
> Copiare l'URL per il prodotto prima di uscire da questa pagina. Una volta che si esce da questa pagina, è necessario utilizzare l'interfaccia a riga di comando per ottenere l'ID prodotto.

Dopo alcuni secondi, il prodotto viene visualizzato nella pagina Products (Prodotti). È possibile che sia necessario aggiornare il browser per visualizzare il prodotto nell'elenco.

Fase 4: creazione della pipeline

1. Per assegnare un nome alla pipeline e selezionare i parametri per la pipeline, procedere nel seguente modo:

   a. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

   b. Selezionare Getting started (Nozioni di base). Scegliere Create pipeline (Crea pipeline) e immettere un nome per la pipeline.

   c. In Ruolo di servizio, scegli Nuovo ruolo di servizio CodePipeline per consentire la creazione di un ruolo di servizio in IAM.

   d. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).

2. Per aggiungere una fase di origine, procedere come segue:

   a. In Source provider (Provider origine), scegliere AWS CodeCommit.

   b. In Repository name (Nome del repository) e Branch name (Nome ramo), immettere il repository e il ramo da utilizzare per l'operazione di origine.

   c. Seleziona Successivo.

3. In Add build stage (Aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora).

4. In Aggiunta della fase di distribuzione, procedere come segue:

   a. In Deploy provider (Provider di distribuzione), scegliere AWS Service Catalog.

   b. Selezionare Use configuration file (Usa file di configurazione).

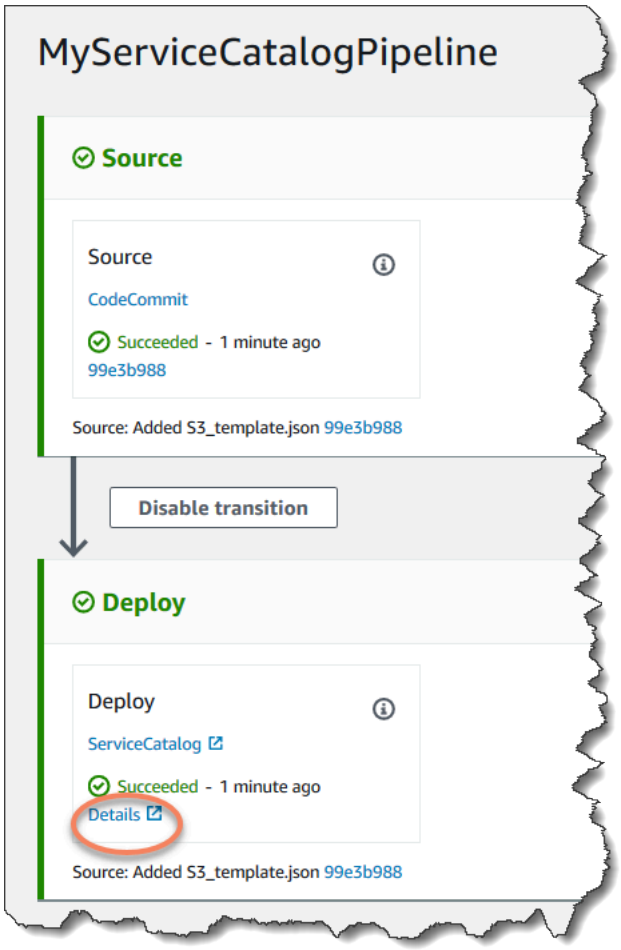
   c. In Product ID, incolla l'ID del prodotto che hai copiato dalla console Service Catalog.

   d. In Configuration file path (Percorso file di configurazione), immettere il percorso del file di configurazione nel repository.

   e. Seleziona Successivo.

5. In Review (Verifica), esaminare le impostazioni della pipeline e selezionare Create (Crea).

6. Dopo che la pipeline è stata eseguita correttamente, nella fase di implementazione, scegli Dettagli per aprire il prodotto in Service Catalog.



7. Nelle informazioni sul prodotto, scegliere il nome della versione per aprire il modello di prodotto. Visualizzare la distribuzione del modello.

Fase 5: push di una modifica e verifica del prodotto in Service Catalog

1. Visualizza la pipeline nella CodePipeline console e, nella fase di origine, scegli Dettagli. Il tuo AWS CodeCommit repository di origine si apre nella console. Scegliere Edit (Modifica) ed effettuare una modifica nel file (ad esempio la descrizione).

```
"Description": "Name of Amazon S3 bucket to hold and version website content"
```

2. Eseguire il commit e il push della modifica. La pipeline viene avviata dopo il push della modifica. Una volta completata l'esecuzione della pipeline, nella fase di implementazione, scegli Dettagli per aprire il prodotto in Service Catalog.

3. Nelle informazioni sul prodotto, scegliere il nome della nuova versione per aprire il modello di prodotto. Visualizzare la modifica del modello distribuito.

Tutorial: Creare una pipeline con AWS CloudFormation

Gli esempi forniscono modelli di esempio che ti consentono di AWS CloudFormation creare una pipeline che distribuisce l'applicazione sulle tue istanze ogni volta che il codice sorgente cambia. Il modello di esempio crea una pipeline che è possibile visualizzare in AWS CodePipeline. La pipeline rileva l'arrivo di una modifica salvata tramite Amazon CloudWatch Events.

Argomenti

Esempio 1: creare una AWS CodeCommit pipeline con AWS CloudFormation

Questa procedura dettagliata mostra come utilizzare la AWS CloudFormation console per creare un'infrastruttura che includa una pipeline connessa a un repository di origine. CodeCommit In questo tutorial, utilizzi il file modello di esempio fornito per creare il tuo stack di risorse, che include l'archivio degli artefatti, la pipeline e le risorse di rilevamento delle modifiche, come la regola Amazon Events. CloudWatch Dopo aver creato lo stack di risorse in AWS CloudFormation, puoi visualizzare la pipeline nella console. AWS CodePipeline La pipeline è una pipeline a due fasi con una fase di CodeCommit origine e una fase di distribuzione. CodeDeploy

Prerequisiti:

È necessario aver creato le seguenti risorse da utilizzare con il AWS CloudFormation modello di esempio:

- È necessario aver creato un repository di origine. È possibile utilizzare il AWS CodeCommit repository in Tutorial: crea una pipeline semplice (CodeCommitrepository) cui è stato creato.
- È necessario aver creato un' CodeDeploy applicazione e un gruppo di distribuzione. È possibile utilizzare le CodeDeploy risorse create inTutorial: crea una pipeline semplice (CodeCommitrepository).

- Scegli uno di questi link per scaricare il file AWS CloudFormation modello di esempio per la creazione di una pipeline: YAML | JSON

  Decomprimi il file e memorizzalo nel computer locale.

- Scaricate il file dell'applicazione di esempio SampleApp_Linux.zip.

Crea la tua pipeline in AWS CloudFormation

1. Decomprimi i file da SampleApp_Linux.zip e carica i file nel tuo AWS CodeCommit repository. È necessario caricare i file estratti nella directory radice del repository. È possibile seguire le istruzioni disponibili in Passaggio 2: aggiungi codice di esempio al tuo CodeCommit repository per inviare i file nel repository.
2. Apri la AWS CloudFormation console e scegli Create Stack. Scegliere Con nuove risorse (standard).
3. In Specificare modello, scegli Carica un modello. Seleziona Scegli file, quindi scegli il file modello dal tuo computer locale. Seleziona Successivo.
4. Nel campo Stack name (Nome stack), immetti un nome per la pipeline. Vengono visualizzati i parametri specificati dal modello di esempio. Immetti i seguenti parametri:

   a. In ApplicationName, inserisci il nome della tua CodeDeploy applicazione.

   b. In BetaFleet, inserisci il nome del tuo gruppo di CodeDeploy distribuzione.

   c. In BranchName, inserisci il ramo del repository che desideri utilizzare.

   d. In RepositoryName, inserisci il nome del tuo repository di CodeCommit origine.
5. Seleziona Successivo. Accetta i valori predefiniti nella pagina successiva e scegli Next (Successivo).
6. In Capacità, seleziona Riconosco che AWS CloudFormation potrebbe creare risorse IAM, quindi scegli Create stack.
7. Alla fine della creazione dello stack, visualizza l'elenco di eventi per verificare che non vi siano errori.

   Risoluzione dei problemi

   L'utente IAM che sta creando la pipeline in AWS CloudFormation potrebbe richiedere autorizzazioni aggiuntive per creare risorse per la pipeline. Le seguenti autorizzazioni sono

richieste nella policy AWS CloudFormation per consentire la creazione delle risorse Amazon CloudWatch Events richieste per la CodeCommit pipeline:

```
{
     "Effect": "Allow",
     "Action": [
        "events:PutRule",
        "events:PutEvents",
        "events:PutTargets",
        "events:DeleteRule",
        "events:RemoveTargets",
        "events:DescribeRule"
     ],
     "Resource": "resource_ARN"
}
```

8. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

   In Pipelines (Pipeline), seleziona la pipeline e scegli View (Visualizza). Il diagramma mostra le fasi di distribuzione e di origine della pipeline.

   > **Note**
   >
   > Per visualizzare la pipeline che è stata creata, trova la colonna Logical ID nella scheda Risorse relativa al tuo stack in. AWS CloudFormation Annota il nome nella colonna Physical ID della pipeline. In CodePipeline, puoi visualizzare la pipeline con lo stesso ID fisico (nome della pipeline) nella regione in cui hai creato lo stack.

9. Nel repository di origine, confermare e inviare una modifica. Le risorse con rilevamento delle modifiche raccolgono la modifica e la pipeline inizia.

Esempio 2: creare una pipeline Amazon S3 con AWS CloudFormation

Questa procedura dettagliata mostra come utilizzare la AWS CloudFormation console per creare un'infrastruttura che include una pipeline connessa a un bucket sorgente Amazon S3. In questo tutorial, utilizzi il file modello di esempio fornito per creare il tuo stack di risorse, che include il bucket di origine, l'artifact store, la pipeline e le risorse di rilevamento delle modifiche, come la regola e il percorso di Amazon Events. CloudWatch CloudTrail Dopo aver creato lo stack di risorse in AWS

CloudFormation, puoi visualizzare la pipeline nella console. AWS CodePipeline La pipeline è una pipeline a due fasi con una fase di origine di Amazon S3 e una fase di distribuzione. CodeDeploy

Prerequisiti:

È necessario disporre delle seguenti risorse da utilizzare con il modello di esempio: AWS CloudFormation

- Devi aver creato le istanze Amazon EC2, dove hai installato l' CodeDeploy agente sulle istanze. È necessario aver creato un' CodeDeploy applicazione e un gruppo di distribuzione. Usa Amazon EC2 e CodeDeploy le risorse in cui hai creato. Tutorial: crea una pipeline semplice (CodeCommitrepository)
- Scegli i seguenti link per scaricare i file AWS CloudFormation modello di esempio per la creazione di una pipeline con un sorgente Amazon S3:
  - Scaricare il modello di esempio per la pipeline: YAML | JSON
  - Scarica il modello di esempio per il tuo CloudTrail bucket and trail: YAML | JSON
  - Decomprimi i file e memorizzali nel computer locale.
- Scarica l'applicazione di esempio da _Linux.zip. SampleApp

  Salva il file .zip nel computer locale. Puoi caricare il file .zip dopo aver creato lo stack.

Crea la tua pipeline in AWS CloudFormation

1. Apri la AWS CloudFormation console e scegli Create Stack. Scegliere Con nuove risorse (standard).
2. In Scegli un modello, scegli Carica un modello. Seleziona Scegli file, quindi scegli il file modello dal tuo computer locale. Seleziona Successivo.
3. Nel campo Stack name (Nome stack), immetti un nome per la pipeline. Vengono visualizzati i parametri specificati dal modello di esempio. Immetti i seguenti parametri:

   a. In ApplicationName, inserisci il nome della tua CodeDeploy applicazione. È possibile sostituire il nome predefinito `DemoApplication`.

   b. In BetaFleet, inserisci il nome del tuo gruppo di CodeDeploy distribuzione. È possibile sostituire il nome predefinito `DemoFleet`.

   c. In SourceObjectKey, entra`SampleApp_Linux.zip`. Carica questo file nel bucket dopo che il modello ha creato il bucket e la pipeline.

4. Seleziona Successivo. Accetta i valori predefiniti nella pagina successiva e scegli Next (Successivo).
5. In Capacità, seleziona Riconosco che AWS CloudFormation potrebbe creare risorse IAM, quindi scegli Create stack.
6. Alla fine della creazione dello stack, visualizza l'elenco di eventi per verificare che non vi siano errori.

   Risoluzione dei problemi

   L'utente IAM che sta creando la pipeline in AWS CloudFormation potrebbe richiedere autorizzazioni aggiuntive per creare risorse per la pipeline. Le seguenti autorizzazioni sono richieste nella policy AWS CloudFormation per consentire la creazione delle risorse Amazon CloudWatch Events richieste per la pipeline Amazon S3:

   ```
   {
        "Effect": "Allow",
        "Action": [
           "events:PutRule",
           "events:PutEvents",
           "events:PutTargets",
           "events:DeleteRule",
           "events:RemoveTargets",
           "events:DescribeRule"
        ],
        "Resource": "resource_ARN"
   }
   ```

7. Nella AWS CloudFormation scheda Risorse del tuo stack, visualizza le risorse che sono state create per il tuo stack.

   > **Note**
   >
   > Per visualizzare la pipeline che è stata creata, trova la colonna ID logico nella scheda Risorse relativa al tuo stack in. AWS CloudFormation Annota il nome nella colonna Physical ID della pipeline. In CodePipeline, puoi visualizzare la pipeline con lo stesso ID fisico (nome della pipeline) nella regione in cui hai creato lo stack.

Scegliere il bucket S3 con un'etichetta `sourcebucket` nel nome, ad esempio `s3-cfn-codepipeline-sourcebucket-y04EXAMPLE.`. Non scegliere il bucket dell'artefatto della pipeline.

Il bucket di origine è vuoto perché la risorsa è stata appena creata da AWS CloudFormation. Apri la console Amazon S3 e individua il tuo `sourcebucket` bucket. Scegliere Upload (Carica) e seguire le istruzioni per caricare il file .zip `SampleApp_Linux.zip`.

> **Note**
>
> Se Amazon S3 è il fornitore di origine per la tua pipeline, devi caricare nel tuo bucket tutti i file sorgente impacchettati come un unico file.zip. In caso contrario, l'azione di origine ha esito negativo.

8. Accedi AWS Management Console e apri la console all'indirizzo https://console.aws.amazon.com/codepipeline/. CodePipeline

   In Pipelines (Pipeline), seleziona la pipeline e scegli View (Visualizza). Il diagramma mostra le fasi di distribuzione e di origine della pipeline.
9. Completare i passaggi nella procedura seguente per creare le risorse AWS CloudTrail .

Crea le tue AWS CloudTrail risorse in AWS CloudFormation

1. Apri la AWS CloudFormation console e scegli Create Stack.
2. In Choose a template (Scegli un modello), scegliere Upload a template to Amazon S3 (Carica un modello su Amazon S3). Scegli Sfoglia, quindi seleziona il file modello per le AWS CloudTrail risorse dal tuo computer locale. Seleziona Successivo.
3. In Stack name (Nome stack) immetti un nome per lo stack di risorse. Vengono visualizzati i parametri specificati dal modello di esempio. Immetti i seguenti parametri:

   - In SourceObjectKey, accettate l'impostazione predefinita per il file zip dell'applicazione di esempio.
4. Seleziona Successivo. Accetta i valori predefiniti nella pagina successiva e scegli Next (Successivo).
5. In Capacità, seleziona Riconosco che AWS CloudFormation potrebbe creare risorse IAM, quindi scegli Crea.

6. Alla fine della creazione dello stack, visualizza l'elenco di eventi per verificare che non vi siano errori.

   Le seguenti autorizzazioni sono richieste nella policy per consentire AWS CloudFormation la creazione CloudTrail delle risorse richieste per la pipeline Amazon S3:

   ```
   {
        "Effect": "Allow",
        "Action": [
           "cloudtrail:CreateTrail",
           "cloudtrail:DeleteTrail",
           "cloudtrail:StartLogging",
           "cloudtrail:StopLogging",
           "cloudtrail:PutEventSelectors"
        ],
        "Resource": "resource_ARN"
   }
   ```

7. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

   In Pipelines (Pipeline), seleziona la pipeline e scegli View (Visualizza). Il diagramma mostra le fasi di distribuzione e di origine della pipeline.

8. Nel bucket di origine, confermare e inviare una modifica. Le risorse con rilevamento delle modifiche raccolgono la modifica e la pipeline si avvia.

Tutorial: crea una pipeline che utilizza le variabili delle azioni di AWS CloudFormation distribuzione

In questo tutorial, utilizzi la AWS CodePipeline console per creare una pipeline con un'azione di distribuzione. Quando viene eseguita la pipeline, il modello crea uno stack e un file `outputs`. Gli output generati dal modello di stack sono le variabili generate dall' AWS CloudFormation azione in. CodePipeline

Nell'operazione in cui viene creato lo stack dal modello, designi uno spazio dei nomi variabile. Le variabili prodotte dal file `outputs` possono quindi essere utilizzate da operazioni successive. In questo esempio, crei un set di modifiche basato sulla `StackName` variabile prodotta dall'azione AWS CloudFormation . Dopo un'approvazione manuale, puoi eseguire il set di modifiche e quindi creare un'operazione che elimina lo stack in base alla variabile `StackName`.

Argomenti

Prerequisiti: creare un ruolo AWS CloudFormation di servizio e un repository CodeCommit

Devi avere già quanto segue:

- Un CodeCommit repository. Puoi usare il AWS CodeCommit repository in cui hai creato. Tutorial: crea una pipeline semplice (CodeCommitrepository)
- Questo esempio crea uno stack Amazon DocumentDB da un modello. È necessario utilizzare AWS Identity and Access Management (IAM) per creare un ruolo di AWS CloudFormation servizio con le seguenti autorizzazioni per Amazon DocumentDB.

```
"rds:DescribeDBClusters",
"rds:CreateDBCluster",
"rds:DeleteDBCluster",
"rds:CreateDBInstance"
```

Fase 1: scarica, modifica e carica il modello di esempio AWS CloudFormation

Scarica il file AWS CloudFormation modello di esempio e caricalo nel tuo CodeCommit repository.

1. Passa alla pagina del modello di esempio per la tua regione. Ad esempio, la pagina per us-west-2 è all'indirizzo https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/sample-templates-services-us-west-2.html. In Amazon DocumentDB, scarica il modello per un cluster Amazon DocumentDB. Il nome del file è `documentdb_full_stack.yaml`.

2. Decomprimi il file `documentdb_full_stack.yaml` e aprilo in un editor di testo. Apportare le seguenti modifiche:

   a. Per questo esempio, aggiungi il seguente parametro `Purpose:` alla sezione `Parameters` del modello.

   ```
   Purpose:
     Type: String
     Default: testing
     AllowedValues:
       - testing
       - production
     Description: The purpose of this instance.
   ```

   b. Per questo esempio, aggiungi il seguente output `StackName` alla sezione `Outputs:` del modello.

   ```
   StackName:
     Value: !Ref AWS::StackName
   ```

3. Carica il file modello nel tuo AWS CodeCommit repository. Devi caricare il file del modello decompresso e modificato nella directory root del repository.

   Per utilizzare la CodeCommit console per caricare i file:

   a. Apri la CodeCommit console e scegli il tuo repository dall'elenco Repository.
   b. Seleziona Add file (Aggiungi file), quindi scegli Upload file (Carica file).
   c. Seleziona Choose file (Scegli file) e vai al file. Conferma la modifica inserendo il tuo nome utente e indirizzo e-mail. Scegliere Commit changes (Applica modifiche).

   Il tuo file dovrebbe essere simile al seguente a livello di root nel tuo repository:

   ```
   documentdb_full_stack.yaml
   ```

Fase 2: creazione della pipeline

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con un' CodeCommit azione in cui l'elemento sorgente è il file modello.

- Una fase di distribuzione con un' AWS CloudFormation azione di distribuzione.

A ogni operazione delle fasi di origine e distribuzione create dalla procedura guidata viene assegnato uno spazio dei nomi variabile `SourceVariables` e `DeployVariables`, rispettivamente. Poiché alle operazioni è assegnato uno spazio dei nomi, le variabili configurate in questo esempio sono disponibili per le operazioni downstream. Per ulteriori informazioni, consulta Variables.

Per creare una pipeline con la procedura guidata

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyCFNDeployPipeline`**.
4. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Service role (Ruolo del servizio), procedere in uno dei seguenti modi:
   - Scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.
   - Scegli Existing service role (Ruolo di servizio esistente). In Role name (Nome ruolo), scegli il ruolo del servizio dall'elenco.
6. In Artifact store (Archivio di artefatti):
   a. Scegli Posizione predefinita per utilizzare l'archivio di artifact predefinito, ad esempio il bucket di artifact Amazon S3 designato come predefinito, per la tua pipeline nella regione selezionata per la tua pipeline.
   b. Scegli Ubicazione personalizzata se disponi già di un archivio di artefatti, ad esempio un bucket di artefatti Amazon S3, nella stessa regione della pipeline.

> **Note**
>
> Non si tratta del bucket di origine per il codice sorgente, ma dell'archivio artefatti per la pipeline. È richiesto un archivio artefatti separato, ad esempio un bucket S3, per ogni

pipeline. Quando crei o modifichi una pipeline, devi avere un bucket di artefatti nella regione della pipeline e un bucket di artefatti per regione in cui stai eseguendo un'azione. AWS
Per ulteriori informazioni, consulta Artefatti di input e output e CodePipeline riferimento alla struttura della tubazione.

Seleziona Successivo.

7. In Fase 2: Aggiungi fase di origine:

   a. In Source provider (Provider origine), scegliere AWS CodeCommit.
   b. In Nome repository, scegli il nome del repository in cui hai creato. CodeCommit Fase 1: Creare un CodeCommit repository
   c. In Branch name (Nome ramo), scegliere il nome del ramo che contiene l'aggiornamento di codice più recente.

   Dopo aver selezionato il nome e il ramo del repository, viene visualizzata la regola Amazon CloudWatch Events da creare per questa pipeline.

   Seleziona Successivo.

8. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora).

   Seleziona Successivo.

9. In Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione):

   a. In Nome azione, scegli Distribuisci. In Deploy provider (Provider di distribuzione), scegliere CloudFormation.
   b. In Modalità azione, scegli Crea o aggiorna uno stack.
   c. In Nome stack, immetti un nome per lo stack. Questo è il nome dello stack che viene creato dal modello.
   d. In Nome file di output, immetti un nome per il file di output, ad esempio **`outputs`**. Questo è il nome del file che viene creato dall'operazione dopo la creazione dello stack.
   e. Espandere Advanced (Avanzate). In Sostituzioni di parametro, immetti le sostituzioni del modello come coppie chiave-valore. Ad esempio, questo modello richiede le sostituzioni seguenti.

```
{
"DBClusterName": "MyDBCluster",
"DBInstanceName": "MyDBInstance",
"MasterUser": "UserName",
"MasterPassword": "Password",
"DBInstanceClass": "db.r4.large",
"Purpose": "testing"}
```

Se non immetti le sostituzioni, il modello crea uno stack con valori predefiniti.

f. Seleziona Successivo.

g. Scegliere Create pipeline (Crea pipeline). Consenti l'esecuzione della pipeline. La pipeline in due fasi è completa e pronta per l'aggiunta di altre fasi.

Passaggio 3: aggiungere un'azione di AWS CloudFormation distribuzione per creare il set di modifiche

Crea un'azione successiva nella tua pipeline che AWS CloudFormation consenta di creare il set di modifiche prima dell'azione di approvazione manuale.

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

   In Pipelines (Pipeline), seleziona la pipeline e scegli View (Visualizza). Il diagramma mostra le fasi di distribuzione e di origine della pipeline.

2. Scegli questa opzione per modificare la pipeline o continuare a visualizzare la pipeline in modalità Modifica.

3. Scegli di modificare la fase di distribuzione.

4. Aggiungi un'azione di distribuzione che creerà un set di modifiche per lo stack creato nell'azione precedente. Aggiungi questa azione dopo l'azione esistente nella fase.

   a. In Nome azione, immetti Change_Set. In Action provider, scegli AWS CloudFormation .

   b. In Input artifact, scegli. SourceArtifact

   c. In Action mode (Modalità operazione) selezionare Create or replace a change set (Crea o sostituisci un set di modifiche).

d. In Nome stack, immetti la sintassi della variabile come indicato di seguito. Si tratta del nome dello stack per cui viene creato il set di modifiche in cui lo spazio dei nomi predefinito `DeployVariables` viene assegnato all'operazione.

```
#{DeployVariables.StackName}
```

e. In Nome del set di modifiche, immetti il nome del set di modifiche.

```
my-changeset
```

f. In Sostituzioni di parametro, modifica il parametro `Purpose` da `testing` a `production`.

```
{
"DBClusterName": "MyDBCluster",
"DBInstanceName": "MyDBInstance",
"MasterUser": "UserName",
"MasterPassword": "Password",
"DBInstanceClass": "db.r4.large",
"Purpose": "production"}
```

g. Seleziona Fatto per salvare l'operazione.

Fase 4: aggiunta di un'operazione di approvazione manuale

Crea un'operazione di approvazione manuale nella pipeline.

1. Scegli questa opzione per modificare la pipeline o continuare a visualizzare la pipeline in modalità Modifica.
2. Scegliete di modificare la fase Deploy.
3. Aggiungi un'operazione di approvazione manuale dopo l'operazione di distribuzione che crea il set di modifiche. Questa azione consente di verificare l'impostazione della modifica della risorsa creata AWS CloudFormation prima che la pipeline esegua il set di modifiche.

Passaggio 5: aggiungere un'azione di CloudFormation distribuzione per eseguire il set di modifiche

Crea un'azione successiva nella tua pipeline che AWS CloudFormation consenta di eseguire il set di modifiche dopo l'azione di approvazione manuale.

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

   In Pipelines (Pipeline), seleziona la pipeline e scegli View (Visualizza). Il diagramma mostra le fasi di distribuzione e di origine della pipeline.
2. Scegli questa opzione per modificare la pipeline o continuare a visualizzare la pipeline in modalità Modifica.
3. Scegli di modificare la fase di distribuzione.
4. Aggiungi un'azione di distribuzione che eseguirà il set di modifiche approvato nella precedente azione manuale:

   a. In Nome azione, immetti Execute_Change_Set. In Action provider, scegli AWS CloudFormation.

   b. In Input artifact, scegli. SourceArtifact

   c. In Action mode (Modalità operazione), selezionare Execute a change set (Esegui un set di modifiche).

   d. In Nome stack, immetti la sintassi della variabile come indicato di seguito. Questo è il nome dello stack per cui viene creato il set di modifiche.

   ```
   #{DeployVariables.StackName}
   ```

   e. In Nome del set di modifiche, immetto il nome del set di modifiche creato nell'operazione precedente.

   ```
   my-changeset
   ```

   f. Seleziona Fatto per salvare l'operazione.

   g. Continua l'esecuzione della pipeline.

Passaggio 6: aggiungere un'azione di CloudFormation distribuzione per eliminare lo stack

Crea un'azione finale nella tua pipeline che permetta di AWS CloudFormation ottenere il nome dello stack dalla variabile nel file di output ed eliminare lo stack.

1. Apri la console all'indirizzo https://console.aws.amazon.com/codepipeline/ CodePipeline .

   In Pipelines (Pipeline), seleziona la pipeline e scegli View (Visualizza). Il diagramma mostra le fasi di distribuzione e di origine della pipeline.
2. Scegli questa opzione per modificare la pipeline.
3. Scegli di modificare la fase di distribuzione.
4. Aggiungi un'operazione di distribuzione che elimina lo stack:

   a. In Nome azione, scegli DeleteStack. In Deploy provider (Provider di distribuzione), scegliere CloudFormation.

   b. In Modalità azione, scegli Elimina uno stack.

   c. In Nome stack, immetti la sintassi della variabile come indicato di seguito. Questo è il nome dello stack che viene eliminato dall'operazione.

   d. Seleziona Fatto per salvare l'operazione.

   e. Seleziona Salva per salvare la pipeline.

   La pipeline viene eseguita quando viene salvata.

Tutorial: distribuzione standard di Amazon ECS con CodePipeline

Questo tutorial ti aiuta a creare una pipeline di distribuzione end-to-end continua (CD) completa con Amazon ECS with. CodePipeline

> **Note**
>
> Questo tutorial riguarda l'azione di distribuzione standard di Amazon ECS per CodePipeline. Per un tutorial che utilizza Amazon ECS per l'implementazione CodeDeploy blu/green in

CodePipeline, consulta. Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy

Prerequisiti

Per utilizzare questo tutorial per creare la pipeline di distribuzione continua, è necessario disporre di alcune risorse. Ecco di cosa hai bisogno prima di iniziare:

> **Note**
>
> Tutte queste risorse devono essere create all'interno della stessa regione. AWS

- Un repository per il controllo del codice sorgente (utilizzato in questo tutorial CodeCommit) con il Dockerfile e l'origine dell'applicazione. Per ulteriori informazioni, consulta Creare un CodeCommit repository nella Guida per l'utente.AWS CodeCommit
- Un repository di immagini Docker (questo tutorial utilizza Amazon ECR) che contiene un'immagine che hai creato dal tuo Dockerfile e dall'origine dell'applicazione. Per ulteriori informazioni, consulta Creating a Repository and Pushing an Image nella Amazon Elastic Container Registry User Guide.
- Una definizione di attività Amazon ECS che fa riferimento all'immagine Docker ospitata nel tuo repository di immagini. Per ulteriori informazioni, consulta Creating a Task Definition nella Amazon Elastic Container Service Developer Guide.

  > **Important**
  >
  > L'azione di distribuzione standard di Amazon ECS CodePipeline crea la propria revisione della definizione dell'attività in base alla revisione utilizzata dal servizio Amazon ECS. Se crei nuove revisioni per la definizione dell'attività senza aggiornare il servizio Amazon ECS, l'azione di distribuzione ignorerà tali revisioni.

  Di seguito è riportato un esempio di definizione di attività utilizzata per questo tutorial. Il valore utilizzato `name` e che `family` verrà utilizzato nel passaggio successivo per il file delle specifiche della build.

```
{
  "ipcMode": null,
```

```
  "executionRoleArn": "role_ARN",
  "containerDefinitions": [
    {
      "dnsSearchDomains": null,
      "environmentFiles": null,
      "logConfiguration": {
        "logDriver": "awslogs",
        "secretOptions": null,
        "options": {
          "awslogs-group": "/ecs/hello-world",
          "awslogs-region": "us-west-2",
          "awslogs-stream-prefix": "ecs"
        }
      },
      "entryPoint": null,
      "portMappings": [
        {
          "hostPort": 80,
          "protocol": "tcp",
          "containerPort": 80
        }
      ],
      "command": null,
      "linuxParameters": null,
      "cpu": 0,
      "environment": [],
      "resourceRequirements": null,
      "ulimits": null,
      "dnsServers": null,
      "mountPoints": [],
      "workingDirectory": null,
      "secrets": null,
      "dockerSecurityOptions": null,
      "memory": null,
      "memoryReservation": 128,
      "volumesFrom": [],
      "stopTimeout": null,
      "image": "image_name",
      "startTimeout": null,
      "firelensConfiguration": null,
      "dependsOn": null,
      "disableNetworking": null,
      "interactive": null,
      "healthCheck": null,
```

```
      "essential": true,
      "links": null,
      "hostname": null,
      "extraHosts": null,
      "pseudoTerminal": null,
      "user": null,
      "readonlyRootFilesystem": null,
      "dockerLabels": null,
      "systemControls": null,
      "privileged": null,
      "name": "hello-world"
    }
  ],
  "placementConstraints": [],
  "memory": "2048",
  "taskRoleArn": null,
  "compatibilities": [
    "EC2",
    "FARGATE"
  ],
  "taskDefinitionArn": "ARN",
  "family": "hello-world",
  "requiresAttributes": [],
  "pidMode": null,
  "requiresCompatibilities": [
    "FARGATE"
  ],
  "networkMode": "awsvpc",
  "cpu": "1024",
  "revision": 1,
  "status": "ACTIVE",
  "inferenceAccelerators": null,
  "proxyConfiguration": null,
  "volumes": []
}
```

- Un cluster Amazon ECS che esegue un servizio che utilizza la definizione di attività menzionata in precedenza. Per ulteriori informazioni, consulta Creazione di un cluster e creazione di un servizio nella Amazon Elastic Container Service Developer Guide.

Dopo aver soddisfatto questi prerequisiti, puoi procedere con il tutorial e creare la pipeline di distribuzione continua.

Fase 1: aggiunta di un file di specifica di compilazione all'archivio di codice sorgente

Questo tutorial serve CodeBuild per creare la tua immagine Docker e inviarla ad Amazon ECR. Aggiungi un `buildspec.yml` file al tuo repository di codice sorgente per spiegare CodeBuild come eseguire questa operazione. La specifica di compilazione di esempio in basso comprende quanto segue:

- Fase precedente alla compilazione:
  - Accedi ad Amazon ECR.
  - Imposta l'URI dell'archivio sull'immagine ECR e aggiungi un tag di immagine con i primi sette caratteri dell'ID commit di Git del codice sorgente.
- Fase di compilazione:
  - Crea l'immagine Docker e contrassegna con dei tag l'immagine sia come `latest` che con l'ID commit di Git.
- Fase posteriore alla compilazione:
  - Inserisci l'immagine nell'archivio ECR con entrambi i tag.
  - Scrivi un file chiamato `imagedefinitions.json` nella build root che contenga il nome del contenitore del servizio Amazon ECS, l'immagine e il tag. La fase di distribuzione della pipeline di distribuzione utilizza queste informazioni per creare una nuova revisione della definizione di attività del servizio, quindi aggiorna il servizio per utilizzare la nuova definizione di attività. Il file `imagedefinitions.json` è richiesto per l'esecutore del processo ECS.

Incolla questo testo di esempio per creare il `buildspec.yml` file e sostituisci i valori per la definizione dell'immagine e dell'attività. Questo testo utilizza l'ID di account di esempio 111122223333.

```
version: 0.2

phases:
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - aws --version
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --
username AWS --password-stdin 111122223333.dkr.ecr.us-west-2.amazonaws.com
      - REPOSITORY_URI=012345678910.dkr.ecr.us-west-2.amazonaws.com/hello-world
```

```
      - COMMIT_HASH=$(echo $CODEBUILD_RESOLVED_SOURCE_VERSION | cut -c 1-7)
      - IMAGE_TAG=${COMMIT_HASH:=latest}
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $REPOSITORY_URI:latest .
      - docker tag $REPOSITORY_URI:latest $REPOSITORY_URI:$IMAGE_TAG
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker images...
      - docker push $REPOSITORY_URI:latest
      - docker push $REPOSITORY_URI:$IMAGE_TAG
      - echo Writing image definitions file...
      - printf '[{"name":"hello-world","imageUri":"%s"}]' $REPOSITORY_URI:$IMAGE_TAG >
 imagedefinitions.json
artifacts:
    files: imagedefinitions.json
```

La specifica di build è stata scritta per la definizione di attività di esempio fornita in Prerequisiti, utilizzata dal servizio Amazon ECS per questo tutorial. Il valore `REPOSITORY_URI` corrisponde all'archivio `image` (senza tag di immagini); il valore *`hello-world`* alla fine del file corrisponde al nome del container nella definizione di attività del servizio.

Aggiunta di un file **`buildspec.yml`** nell'archivio di codice sorgente

1. Apri un editor di testo e copia e incolla la specifica di compilazione riportata in alto in un nuovo file.
2. Sostituisci il `REPOSITORY_URI` valore (*`012345678910.dkr.ecr.us-west-2.amazonaws.com/hello-world`*) con l'URI del tuo repository Amazon ECR (senza tag di immagine) per la tua immagine Docker. Sostituisci *`hello-world`* con il nome del container della definizione di attività del servizio riferito all'immagine Docker.
3. Conferma e inserisci il file `buildspec.yml` nell'archivio del codice sorgente.

   a. Aggiungi il file.

   ```
   git add .
   ```

   b. Conferma la modifica.

```
git commit -m "Adding build specification."
```

c. Invia la conferma.

```
git push
```

Fase 2: creazione della pipeline di distribuzione continua

Utilizza la CodePipeline procedura guidata per creare le fasi della pipeline e connettere il repository di origine al servizio ECS.

Creazione della pipeline

1. Apri la console all'indirizzo https://console.aws.amazon.com/codepipeline/ CodePipeline .
2. Nella pagina Welcome (Benvenuto), seleziona Create pipeline (Crea pipeline).

   Se è la prima volta che lo usi CodePipeline, viene visualizzata una pagina introduttiva anziché Benvenuto. Scegliere Get Started Now (Inizia subito).
3. Nella pagina Step 1: Name, per Pipeline name, digita il nome della pipeline. Per questo tutorial, il nome della pipeline è hello-world.
4. In Tipo di pipeline, scegliete V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta la pagina Tipi di pipeline. Seleziona Next (Successivo).
5. Nella pagina Passaggio 2: Aggiungi fase di origine, per Provider di origine, scegli. AWS CodeCommit

   a. Per Nome del repository, scegli il nome del CodeCommit repository da utilizzare come posizione di origine per la pipeline.

   b. In Branch name (Nome ramo), seleziona il ramo da utilizzare e seleziona Next (Fase successiva).
6. Nella pagina Passaggio 3: Aggiungi fase di compilazione, per Build provider scegli AWS CodeBuild, quindi scegli Crea progetto.

   a. In Project name (Nome progetto), scegli un nome univoco per il progetto di compilazione. Per questo tutorial, il nome della progetto è hello-world.

   b. Per Environment image (Immagine ambiente), scegliere Managed image (Immagine gestita).

c. Per Operating system (Sistema operativo), scegliere Amazon Linux 2.

d. In Runtime(s) (Runtime), seleziona Standard.

e. Per Immagine, scegli **`aws/codebuild/amazonlinux2-x86_64-standard:3.0`**.

f. Per la Image version (Versione immagine) e Environment type (Tipo di ambiente), utilizzare i valori predefiniti.

g. Seleziona Enable this flag if you want to build Docker images or want your builds to get elevated privileges (Abilita questo flag se desideri creare immagini Docker o se desideri che le build ottengano privilegi elevati).

h. Deseleziona i CloudWatch registri. Potrebbe essere necessario espandere Advanced.

i. Scegli Continua a CodePipeline.

j. Seleziona Successivo.

> **Note**
>
> La procedura guidata crea un ruolo di CodeBuild servizio per il progetto di compilazione, chiamato codebuild- ***`build-project-name`***-service-role. Prendi nota di questo nome di ruolo, man mano che aggiungerai le autorizzazioni Amazon ECR in un secondo momento.

7. Nella pagina Fase 4: Aggiungi fase di distribuzione, per provider di distribuzione, scegli Amazon ECS.

   a. Per il nome del cluster, scegli il cluster Amazon ECS in cui è in esecuzione il tuo servizio. Per questo tutorial, il cluster è predefinito.

   b. In Service name (Nome del servizio), scegli il servizio da aggiornare, quindi seleziona Next (Fase successiva). Per questo tutorial, il nome del servizio è hello-world.

8. Nella pagina Step 5: Review (Fase 5: verifica), esamina la configurazione della pipeline, quindi seleziona Create pipeline (Crea pipeline) per creare la pipeline.

> **Note**
>
> Ora che la pipeline è stata creata, tenta l'esecuzione tramite le varie fasi della pipeline. Tuttavia, il CodeBuild ruolo predefinito creato dalla procedura guidata non dispone delle autorizzazioni per eseguire tutti i comandi contenuti nel `buildspec.yml` file,

quindi la fase di compilazione fallisce. La sezione successiva aggiunge le autorizzazioni necessarie per la fase di compilazione.

Fase 3: aggiungere le autorizzazioni Amazon ECR al ruolo CodeBuild

La CodePipeline procedura guidata ha creato un ruolo IAM per il progetto di compilazione, chiamato CodeBuild codebuild - -service-role. *build-project-name* Per questo tutorial, il nome è -role. codebuild-hello-world-service Poiché il `buildspec.yml` file effettua chiamate alle operazioni dell'API Amazon ECR, il ruolo deve disporre di una policy che consenta le autorizzazioni per effettuare queste chiamate Amazon ECR. La procedura seguente consente di collegare al ruolo le autorizzazioni appropriate.

Per aggiungere le autorizzazioni Amazon ECR al ruolo CodeBuild

1. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel pannello di navigazione a sinistra, seleziona Ruoli.
3. Nella casella di ricerca, digita codebuild- e scegli il ruolo creato dalla procedura guidata. CodePipeline Per questo tutorial, il nome del ruolo è codebuild-hello-world-service -role.
4. Nella pagina Summary (Riepilogo), seleziona Attach policy (Collega policy).
5. Seleziona la casella a sinistra della politica di AmazonEC2 e scegli ContainerRegistryPowerUser Allega politica.

Fase 4: test della pipeline

La tua pipeline dovrebbe avere tutto il necessario per eseguire una end-to-end distribuzione continua nativa. AWS Ora, testane la funzionalità inserendo una modifica del codice all'archivio del codice sorgente.

Test della pipeline

1. Effettua una modifica del codice nell'archivio del codice sorgente configurato, conferma e inserisci la modifica.
2. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
3. Seleziona la pipeline nell'elenco.

4. Osserva l'avanzamento della pipeline attraverso le varie fasi. La pipeline dovrebbe essere completata e il servizio Amazon ECS esegue l'immagine Docker creata dalla modifica del codice.

Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy

In questo tutorial, configuri una pipeline AWS CodePipeline che distribuisce le applicazioni container utilizzando una distribuzione blu/verde che supporta le immagini Docker. In una distribuzione blu/verde, è possibile avviare la nuova versione dell'applicazione insieme alla versione precedente e testare la nuova versione prima di reindirizzare il traffico. Puoi anche monitorare il processo di distribuzione ed eseguire rapidamente il rollback in caso di problemi.

> **Note**
>
> Questo tutorial è destinato all'azione di distribuzione di Amazon ECS to CodeDeploy blu/green. CodePipeline Per un tutorial che utilizza l'azione di distribuzione standard di Amazon ECS in CodePipeline, consultaTutorial: distribuzione standard di Amazon ECS con CodePipeline.

La pipeline completata rileva le modifiche all'immagine, che viene archiviata in un repository di immagini come Amazon ECR e utilizzata CodeDeploy per indirizzare e distribuire il traffico verso un cluster Amazon ECS e un sistema di bilanciamento del carico. CodeDeploy utilizza un listener per reindirizzare il traffico verso la porta del contenitore aggiornato specificata nel file. AppSpec Per informazioni su come il load balancer, il listener di produzione, i gruppi target e l'applicazione Amazon ECS vengono utilizzati in una distribuzione blu/verde, consulta Tutorial: Deploy an Amazon ECS Service.

La pipeline è inoltre configurata per utilizzare una posizione di origine CodeCommit, ad esempio dove è archiviata la definizione delle attività di Amazon ECS. In questo tutorial, configurerai ognuna di queste AWS risorse e poi creerai la tua pipeline con fasi che contengono azioni per ogni risorsa.

La tua pipeline di distribuzione continua creerà e distribuirà automaticamente immagini di container ogni volta che il codice sorgente viene modificato o viene caricata una nuova immagine di base su Amazon ECR.

Questo flusso utilizza i seguenti artefatti:

- Un file di immagine Docker che specifica il nome del contenitore e l'URI del repository del tuo repository di immagini Amazon ECR.
- Una definizione di attività Amazon ECS che elenca il nome dell'immagine Docker, il nome del contenitore, il nome del servizio Amazon ECS e la configurazione del load balancer.
- Un CodeDeploy AppSpec file che specifica il nome del file di definizione delle attività di Amazon ECS, il nome del contenitore dell'applicazione aggiornata e la porta del contenitore in cui CodeDeploy reindirizza il traffico di produzione. Puoi anche specificare la configurazione di rete opzionale e le funzioni Lambda che puoi eseguire durante gli hook di eventi del ciclo di vita di distribuzione.

> **Note**
>
> Quando esegui una modifica al tuo repository di immagini Amazon ECR, l'azione di origine della pipeline crea un `imageDetail.json` file per quel commit. Per ulteriori informazioni sul file `imageDetail.json`, consulta File ImageDetail.json per le azioni di distribuzione blu/verde di Amazon ECS.

Quando crei o modifichi la pipeline e aggiorni o specifichi artefatti di origine per la fase di distribuzione, assicurati di puntare agli artefatti di origine con il nome e la versione più recenti da utilizzare. Dopo aver configurato la pipeline, quando modifichi la definizione dell'immagine o dell'attività, potrebbe essere necessario aggiornare i file di artefatti di origine nei repository e quindi modificare la fase di distribuzione nella pipeline.

Argomenti

Prerequisiti

Devi aver già creato le seguenti risorse:

- Un repository. CodeCommit È possibile utilizzare il AWS CodeCommit repository in cui è stato creato. Tutorial: crea una pipeline semplice (CodeCommitrepository)
- Avvia un'istanza Amazon EC2 Linux e installa Docker per creare un'immagine come mostrato in questo tutorial. Se disponi già di un'immagine che intendi utilizzare, puoi ignorare questo prerequisito.

Fase 1: creare un'immagine e inviarla a un repository Amazon ECR

In questa sezione, usi Docker per creare un'immagine e poi lo usi AWS CLI per creare un repository Amazon ECR e inviare l'immagine al repository.

> **Note**
>
> Se disponi già di un'immagine che intendi utilizzare, puoi ignorare questa fase.

Per creare un'immagine

1. Accedi all'istanza Linux in cui è installato Docker.

   Seleziona un'immagine per `nginx`. Questo comando fornisce l'immagine: `nginx:latest`

   ```
   docker pull nginx
   ```

2. Esegui docker images. Dovresti poter visualizzare l'immagine nell'elenco.

   ```
   docker images
   ```

Per creare un repository Amazon ECR e inviare la tua immagine

1. Crea un repository Amazon ECR per archiviare l'immagine . Prendi nota del valore di `repositoryUri` nell'output.

   ```
   aws ecr create-repository --repository-name nginx
   ```

Output:

```
{
    "repository": {
        "registryId": "aws_account_id",
        "repositoryName": "nginx",
        "repositoryArn": "arn:aws:ecr:us-east-1:aws_account_id:repository/nginx",
        "createdAt": 1505337806.0,
        "repositoryUri": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/nginx"
    }
}
```

2. Applica un tag all'immagine con il valore `repositoryUri` ricavato nella fase precedente.

```
docker tag nginx:latest aws_account_id.dkr.ecr.us-east-1.amazonaws.com/nginx:latest
```

3. Esegui il aws ecr get-login-password comando, come mostrato in questo esempio per la `us-west-2` regione e l'ID dell'account 111122223333.

```
aws ecr get-login-password --region us-west-2 | docker login --username AWS --
password-stdin 111122223333.dkr.ecr.us-west-2.amazonaws.com/nginx
```

4. Invia l'immagine ad Amazon ECR utilizzando il `repositoryUri` passaggio precedente.

```
docker push 111122223333.dkr.ecr.us-east-1.amazonaws.com/nginx:latest
```

Fase 2: Creare la definizione delle attività e i file AppSpec sorgente e inviarli a un repository CodeCommit

In questa sezione, crei un file JSON di definizione delle attività e lo registri con Amazon ECS. Quindi crei un AppSpec file CodeDeploy e usi il tuo client Git per inviare i file al tuo CodeCommit repository.

Per creare una definizione di attività per l'immagine

1. Crea un file denominato `taskdef.json` con i seguenti contenuti. Per `image`, immetti il nome dell'immagine, ad esempio nginx. Tale valore viene aggiornato quando la pipeline viene eseguita.

> **Note**
>
> Assicurati che il ruolo di esecuzione specificato nella definizione dell'attività contenga `AmazonECSTaskExecutionRolePolicy`. Per ulteriori informazioni, consulta Amazon ECS Task Execution IAM Role nella Amazon ECS Developer Guide.

```
{
    "executionRoleArn": "arn:aws:iam::account_ID:role/ecsTaskExecutionRole",
    "containerDefinitions": [
        {
            "name": "sample-website",
            "image": "nginx",
            "essential": true,
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512",
    "family": "ecs-demo"
}
```

2. Registra la definizione dell'attività con il file `taskdef.json`.

```
aws ecs register-task-definition --cli-input-json file://taskdef.json
```

3. Una volta registrata la definizione dell'attività, modifica il file per rimuovere il nome dell'immagine e includi il testo segnaposto `<IMAGE1_NAME>` nel campo dell'immagine.

```
{
```

```
    "executionRoleArn": "arn:aws:iam::account_ID:role/ecsTaskExecutionRole",
    "containerDefinitions": [
        {
            "name": "sample-website",
            "image": "<IMAGE1_NAME>",
            "essential": true,
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512",
    "family": "ecs-demo"
}
```

Per creare un file AppSpec

- Il AppSpec file viene utilizzato per le CodeDeploy distribuzioni. Il file, che comprende campi opzionali, utilizza questo formato:

```
version: 0.0
Resources:
  - TargetService:
      Type: AWS::ECS::Service
      Properties:
        TaskDefinition: "task-definition-ARN"
        LoadBalancerInfo:
          ContainerName: "container-name"
          ContainerPort: container-port-number
# Optional properties
        PlatformVersion: "LATEST"
        NetworkConfiguration:
            AwsvpcConfiguration:
```

```
              Subnets: ["subnet-name-1", "subnet-name-2"]
              SecurityGroups: ["security-group"]
              AssignPublicIp: "ENABLED"
Hooks:
- BeforeInstall: "BeforeInstallHookFunctionName"
- AfterInstall: "AfterInstallHookFunctionName"
- AfterAllowTestTraffic: "AfterAllowTestTrafficHookFunctionName"
- BeforeAllowTraffic: "BeforeAllowTrafficHookFunctionName"
- AfterAllowTraffic: "AfterAllowTrafficHookFunctionName"
```

Per ulteriori informazioni sul AppSpec file, inclusi esempi, vedere CodeDeploy AppSpec File Reference.

Crea un file denominato `appspec.yaml` con i seguenti contenuti. Per `TaskDefinition`, non modificare il testo segnaposto `<TASK_DEFINITION>`. Tale valore viene aggiornato quando la pipeline viene eseguita.

```
version: 0.0
Resources:
  - TargetService:
      Type: AWS::ECS::Service
      Properties:
        TaskDefinition: <TASK_DEFINITION>
        LoadBalancerInfo:
          ContainerName: "sample-website"
          ContainerPort: 80
```

Per inviare file al tuo CodeCommit repository

1. Invia o carica i file nel tuo CodeCommit repository. Questi file sono l'elemento sorgente creato dalla procedura guidata Create pipeline per l'azione di distribuzione in. CodePipeline I file dovrebbero avere questo aspetto nella directory locale:

```
/tmp
  |my-demo-repo
    |-- appspec.yaml
    |-- taskdef.json
```

2. Scegli un metodo per caricare i file:

   a. Per usare la riga di comando git da un repository clonato sul computer locale:

i. Cambia le directory nel repository locale:

```
(For Linux, macOS, or Unix) cd /tmp/my-demo-repo
(For Windows) cd c:\temp\my-demo-repo
```

ii. Esegui il comando seguente per posizionare tutti i file contemporaneamente:

```
git add -A
```

iii. Esegui il comando seguente per eseguire il commit dei file con un messaggio di commit:

```
git commit -m "Added task definition files"
```

iv. Esegui il comando seguente per inviare i file dal repository locale al tuo repository: CodeCommit

```
git push
```

b. Per utilizzare la CodeCommit console per caricare i file:

i. Apri la CodeCommit console e scegli il tuo repository dall'elenco Repository.

ii. Seleziona Add file (Aggiungi file), quindi scegli Upload file (Carica file).

iii. Seleziona Choose file (Scegli file), quindi seleziona il file. Conferma la modifica inserendo il tuo nome utente e indirizzo e-mail. Scegliere Commit changes (Applica modifiche).

iv. Ripeti questa fase per ogni file da caricare.

Fase 3: creazione dell'Application Load Balancer e dei gruppi di destinazione

In questa sezione, crei un Application Load Balancer di Amazon EC2. Utilizzerai i nomi delle sottoreti e i valori dei gruppi target che creerai con il tuo sistema di bilanciamento del carico in un secondo momento, quando crei il tuo servizio Amazon ECS. Puoi creare un Application Load Balancer o un Network Load Balancer. Il sistema di bilanciamento del carico deve utilizzare un VPC con due sottoreti pubbliche in zone di disponibilità diverse. In queste fasi, confermerai il VPC predefinito, creerai un sistema di bilanciamento del carico, quindi due gruppi di destinazione per il sistema di

bilanciamento del carico. Per ulteriori informazioni, consulta Gruppi di destinazione per i Network Load Balancer.

Per verificare il VPC predefinito e le sottoreti pubbliche

1. Accedi AWS Management Console e apri la console Amazon VPC all'indirizzo https://console.aws.amazon.com/vpc/.
2. Verifica il VPC predefinito da utilizzare. Nel pannello di navigazione, scegli Your VPCs (I tuoi VPC). Controlla quale VPC mostra Yes (Sì) nella colonna Default VPC (VPC predefinito). Questo è il VPC predefinito. Contiene le sottoreti predefinite da selezionare.
3. Scegli Subnets (Sottoreti). Scegli due sottoreti che mostrano Yes (Sì) nella colonna Default subnet (Sottorete predefinita).

   > **Note**
   >
   > Annota gli ID delle sottoreti. Saranno necessari più avanti in questo tutorial.

4. Scegli le sottoreti, quindi scegli la scheda Description (Descrizione). Verifica che le sottoreti da utilizzare siano in diverse zone di disponibilità.
5. Scegli le sottoreti, quindi scegli la scheda Route Table (Tabella di routing). Per verificare che ciascuna sottorete che desideri utilizzare sia una sottorete pubblica, controlla che una riga di gateway sia inclusa nella tabella di routing.

Per creare un Application Load Balancer di Amazon EC2

1. Accedi AWS Management Console e apri la console Amazon EC2 all'indirizzo https://console.aws.amazon.com/ec2/.
2. Selezionare Sistemi di bilanciamento del carico nel riquadro di navigazione.
3. Seleziona Crea sistema di bilanciamento del carico.
4. Scegli Application Load Balancer, quindi scegli Create (Crea).
5. In Name (Nome), immetti il nome del sistema di bilanciamento del carico.
6. In Scheme (Schema), scegli internet-facing.
7. In IP address type (Tipo di indirizzo IP), scegli ipv4.
8. Configura due porte listener per il sistema di bilanciamento del carico:

   a. In Load Balancer Protocol (Protocollo del sistema di bilanciamento del carico), scegli HTTP. In Load Balancer Port (Porta del sistema di bilanciamento del carico), immetti **80**.
   b. Scegli Add listener (Aggiungi listener).
   c. In Load Balancer Protocol (Protocollo del sistema di bilanciamento del carico) per il secondo listener, scegli HTTP. In Load Balancer Port (Porta del sistema di bilanciamento del carico), immetti **8080**.
9. In Availability Zones (Zone di disponibilità), in VPC, scegli il VPC predefinito. Quindi, scegli le due sottoreti predefinite da utilizzare.
10. Seleziona Next: Configure Security Settings (Fase successiva: configurazione delle impostazioni di sicurezza).
11. Seleziona Next: Configure Security Groups (Fase successiva: configurazione dei gruppi di sicurezza).
12. Scegli Select an existing security group (Seleziona un gruppo di sicurezza esistente) e prendi nota dell'ID del gruppo di sicurezza.
13. Seleziona Successivo: Configurazione del routing.
14. In Target group (Gruppo di destinazione), scegli New target group (Nuovo gruppo di destinazione) e configura il primo gruppo di destinazione:
    a. In Name (Nome), immetti il nome di un gruppo di destinazione (ad esempio, **target-group-1**).
    b. In Target type (Tipo di destinazione), scegli IP.
    c. In Protocol (Protocollo), scegli HTTP. In Port (Porta), immetti **80**.
    d. Seleziona Next: Register Targets (Fase successiva: registrazione delle destinazioni).
15. Scegli Next: Review (Fase successiva: revisione), quindi seleziona Create (Crea).

Per creare un secondo gruppo di destinazione per il sistema di bilanciamento del carico

1. Dopo aver effettuato il provisioning del sistema di bilanciamento del carico, apri la console Amazon EC2. Seleziona Gruppi di destinazioni nel riquadro di navigazione.
2. Scegliere Crea gruppo target.
3. In Name (Nome), immetti il nome di un gruppo di destinazione (ad esempio, **target-group-2**).
4. In Target type (Tipo di destinazione), scegli IP.
5. In Protocol (Protocollo), scegli HTTP. In Port (Porta), immetti **8080**.

6. In VPC, scegli il VPC predefinito.
7. Scegli Crea.

> **Note**
>
> È necessario disporre di due gruppi di destinazione creati per il sistema di bilanciamento del carico affinché la distribuzione si avvii. È necessario solo annotare l'ARN del primo gruppo di destinazione. Questo ARN è utilizzato nel file JSON `create-service` nella prossima fase.

Per aggiornare il sistema di bilanciamento del carico per includere il secondo gruppo di destinazione

1. Aprire la console Amazon EC2. Selezionare Sistemi di bilanciamento del carico nel riquadro di navigazione.
2. Seleziona il sistema di bilanciamento del carico, quindi la scheda Listeners (Listener). Scegli il listener con porta 8080 e quindi scegli Edit (Modifica).
3. Seleziona l'icona della matita accanto a Forward to (Inoltra a). Scegli il secondo gruppo di destinazione e quindi seleziona il segno di spunta. Seleziona Update (Aggiorna) per salvare gli aggiornamenti.

Fase 4: crea il cluster e il servizio Amazon ECS

In questa sezione, crei un cluster e un servizio Amazon ECS in cui CodeDeploy indirizza il traffico durante la distribuzione (verso un cluster Amazon ECS anziché istanze EC2). Per creare il tuo servizio Amazon ECS, devi utilizzare i nomi di sottorete, il gruppo di sicurezza e il valore del gruppo target che hai creato con il tuo sistema di bilanciamento del carico per creare il tuo servizio.

> **Note**
>
> Quando utilizzi questi passaggi per creare un cluster Amazon ECS, utilizzi il modello di cluster Networking only, che fornisce i contenitori AWS Fargate. AWS Fargate è una tecnologia che gestisce al posto tuo l'infrastruttura delle istanze di container. Non è necessario scegliere o creare manualmente istanze Amazon EC2 per il cluster Amazon ECS.

Come creare un cluster Amazon ECS

1. Apri la console classica Amazon ECS all'indirizzo https://console.aws.amazon.com/ecs/.
2. Nel pannello di navigazione scegliere Clusters (Cluster).
3. Scegli Crea cluster.
4. Scegli il modello di cluster Solo rete che utilizza AWS Fargate, quindi scegli Passaggio successivo.
5. Nella pagina Configure cluster (Configura cluster), immetti un nome di cluster. È possibile aggiungere un tag facoltativo per la risorsa. Scegli Crea.

Per creare un servizio Amazon ECS

Usa il AWS CLI per creare il tuo servizio in Amazon ECS.

1. Crea un file JSON e denominalo `create-service.json`. Incolla quanto segue nel file JSON.

   Per il `taskDefinition` campo, quando registri una definizione di attività in Amazon ECS, le dai una famiglia. L'operazione è analoga a un nome per più versioni della definizione di attività, specificato con un numero di revisione. In questo esempio, utilizza "`ecs-demo:1`" per la famiglia e il numero di revisione nel file. Utilizzare i nomi di sottorete, il gruppo di sicurezza e il valore del gruppo di destinazione creati con il sistema di bilanciamento del carico in Fase 3: creazione dell'Application Load Balancer e dei gruppi di destinazione .

   > **Note**
   >
   > È necessario includere l'ARN del gruppo di destinazione in questo file. Apri la console Amazon EC2 e dal pannello di navigazione, in LOAD BALANCING, scegli Target Groups. Scegli il primo gruppo di destinazione. Copia l'ARN dalla scheda Description (Descrizione).

```
{
    "taskDefinition": "family:revision-number",
    "cluster": "my-cluster",
    "loadBalancers": [
        {
            "targetGroupArn": "target-group-arn",
            "containerName": "sample-website",
```

```
            "containerPort": 80
        }
    ],
    "desiredCount": 1,
    "launchType": "FARGATE",
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "CODE_DEPLOY"
    },
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                "subnet-1",
                "subnet-2"
            ],
            "securityGroups": [
                "security-group"
            ],
            "assignPublicIp": "ENABLED"
        }
    }
}
```

2. Esegui il comando create-service e specifica il file JSON:

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

Questo esempio crea un servizio denominato `my-service`.

> **Note**
>
> In questo esempio il comando crea un servizio denominato my-service. Se si dispone già di un servizio con questo nome, il comando restituisce un errore.

```
aws ecs create-service --service-name my-service --cli-input-json file://create-service.json
```

L'output restituisce i campi di descrizione per il servizio.

3. Esegui il comando describe-services per verificare l'avvenuta creazione del servizio.

```
aws ecs describe-services --cluster cluster-name --services service-name
```

Fase 5: Crea CodeDeploy l'applicazione e il gruppo di distribuzione (piattaforma di calcolo ECS)

Quando crei un' CodeDeploy applicazione e un gruppo di distribuzione per la piattaforma di calcolo Amazon ECS, l'applicazione viene utilizzata durante una distribuzione per fare riferimento al gruppo di distribuzione, ai gruppi target, agli ascoltatori e al comportamento di reindirizzamento del traffico corretti.

Per creare CodeDeploy un'applicazione

1. Apri la CodeDeploy console e scegli Crea applicazione.
2. In Application name (Nome applicazione), immetti il nome da utilizzare.
3. In Compute platform (Piattaforma di calcolo), scegli Amazon ECS.
4. Scegli Crea applicazione.

Per creare un gruppo CodeDeploy di distribuzione

1. Nella pagina dell'applicazione, nella scheda Deployment groups (Gruppi di distribuzione), scegli Create deployment group (Crea gruppo di distribuzione).
2. In Deployment group name (Nome gruppo di distribuzione), inserire un nome che descriva il gruppo di distribuzione.
3. In Ruolo di servizio, scegli un ruolo di servizio che garantisca CodeDeploy l'accesso ad Amazon ECS. Per creare un nuovo ruolo del servizio, attenersi alla seguente procedura:

   a. Apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/).
   b. Dal pannello di controllo della console, scegli Roles (Ruoli).
   c. Scegli Crea ruolo.
   d. In Seleziona il tipo di entità affidabile, seleziona Servizio AWS. In Scegli un caso d'uso, seleziona CodeDeploy. In Seleziona il tuo caso d'uso, seleziona CodeDeploy - ECS. Scegli

Successivo: autorizzazioni. La policy gestita `AWSCodeDeployRoleForECS` è già collegata al ruolo.

e. Scegli Next: Tags (Successivo: Tag) e Next: Review (Successivo: Verifica).

f. Immetti un nome per il ruolo, ad esempio **`CodeDeployECSRole`**, quindi seleziona Create role (Crea ruolo).

4. In Configurazione dell'ambiente, scegli il nome del cluster Amazon ECS e il nome del servizio.
5. In Load balancer, scegli il nome del load balancer che fornisce il traffico al tuo servizio Amazon ECS.
6. Da Production listener port (Porta del listener di produzione), scegli la porta e il protocollo per il listener che serve il traffico di produzione al servizio Amazon ECS. Da Test listener port (Porta listener test), scegli la porta e il protocollo per il listener test.
7. Da Target group 1 name (Nome gruppo di destinazione 1) e Target group 2 name (Nome gruppo di destinazione 2), scegli i gruppi di destinazione utilizzati per instradare il traffico durante la distribuzione. Assicurati che questi siano i gruppi di destinazione creati per il sistema di bilanciamento del carico.
8. Scegli Reindirizzare immediatamente il traffico per determinare quanto tempo dopo una corretta implementazione reindirizzare il traffico verso la tua attività Amazon ECS aggiornata.
9. Scegliere Create deployment group (Crea gruppo di distribuzione).

Fase 6: creazione della pipeline

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Un' CodeCommit azione in cui gli artefatti di origine sono la definizione dell'attività e il file. AppSpec
- Una fase di origine con un'azione sorgente Amazon ECR in cui l'artefatto di origine è il file di immagine.
- Una fase di distribuzione con un'azione di distribuzione di Amazon ECS in cui la distribuzione viene eseguita con un' CodeDeploy applicazione e un gruppo di distribuzione.

Per creare una pipeline a due fasi con la procedura guidata

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegliere Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyImagePipeline`**.
4. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Ruolo di servizio, scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.
6. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).
7. In Step 2: Add source stage (Fase 2: aggiunta della fase di origine), in Source provider (Provider origine), scegliere AWS CodeCommit. In Nome repository, scegli il nome del CodeCommit repository in cui hai creato. Fase 1: Creare un CodeCommit repository In Branch name (Nome ramo), scegliere il nome del ramo che contiene l'aggiornamento di codice più recente.

   Seleziona Successivo.
8. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Successivo.
9. In Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione):

   a. In Deploy provider (Distribuisci provider), scegli Amazon ECS (Blue/Green) (Blu/verde)). In Application name (Nome applicazione) immetti o scegli il nome dell'applicazione dall'elenco, ad esempio `codedeployapp`. In Deployment group (Gruppo di distribuzione), immetti o scegli il gruppo di distribuzione dall'elenco, ad esempio `codedeploydeplgroup`.

   > **Note**
   >
   > Il nome "Deploy (Distribuzione)" è quello assegnato per impostazione predefinita alla fase creata in Fase 4: distribuzione, così come "Source (Origine)" è il nome assegnato alla prima fase della pipeline.

   b. Nella definizione delle attività di Amazon ECS, scegli SourceArtifact. Nel campo, immetti **`taskdef.json`**.

c. In AWS CodeDeploy AppSpec File, scegli SourceArtifact. Nel campo, immetti **`appspec.yaml`**.

> **Note**
>
> In questa fase, non immettere nessuna informazione in Dynamically update task definition image (Aggiorna dinamicamente l'immagine di definizione dell'attività).

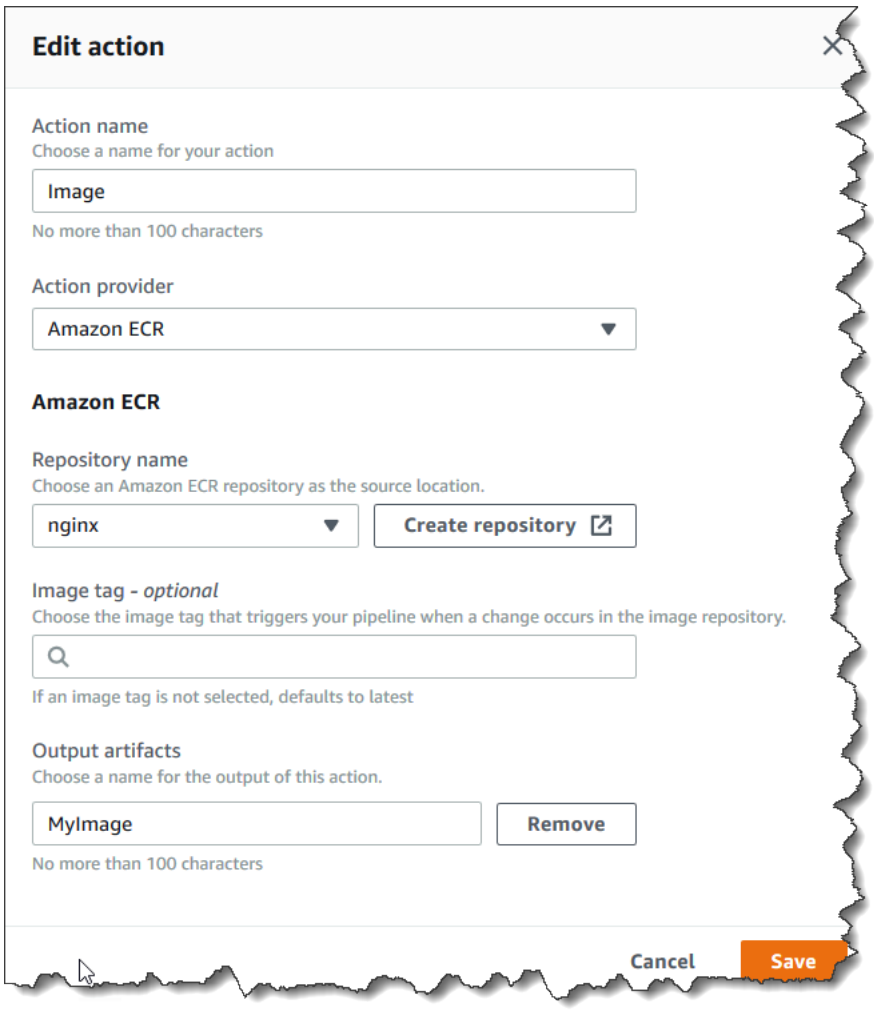
d. Seleziona Successivo.

10. In Step 5: Review (Fase 5: revisione), esaminare le informazioni e quindi scegliere Create pipeline (Crea pipeline).

Per aggiungere un'azione sorgente Amazon ECR alla tua pipeline

Visualizza la tua pipeline e aggiungi un'azione sorgente Amazon ECR alla tua pipeline.

1. Seleziona la pipeline. In alto a sinistra, scegliere Edit (Modifica).
2. Nella fase di origine, scegli Edit stage (Modifica fase).
3. Aggiungi un'azione parallela selezionando + Aggiungi azione accanto all'azione CodeCommit sorgente.
4. In Action name (Nome operazione), immetti un nome (ad esempio, **`Image`**).
5. In Action provider (Provider operazione), scegli Amazon ECR.



6. In Nome repository, scegli il nome del tuo repository Amazon ECR.
7. In Image tag (Tag immagine), specifica il nome e la versione dell'immagine, se diverso da LATEST.
8. In Output artifacts (Artefatti di output), scegli l'artefatto di output predefinito (ad esempio `MyImage`), che include il nome dell'immagine e le informazioni di URI del repository che desideri sia utilizzato nella prossima fase.
9. Scegli Save (Salva) nella schermata dell'operazione. Scegli Done (Fatto) nella schermata della fase. Scegli Save (Salva) nella pipeline. Un messaggio mostra la regola Amazon CloudWatch Events da creare per l'azione sorgente di Amazon ECR.

Per collegare gli artefatti di origine all'operazione di distribuzione

1. Scegli Modifica nella fase di distribuzione e scegli l'icona per modificare l'azione Amazon ECS (blu/verde).

2. Scorri fino alla fine del riquadro. In Input artifacts (Artefatti di input), scegli Add (Aggiungi). Aggiungi l'elemento sorgente dal tuo nuovo repository Amazon ECR (ad esempio,). `MyImage`
3. In Task Definition, scegli SourceArtifact, quindi inserisci la verifica. **`taskdef.json`**
4. In AWS CodeDeploy AppSpec File, scegliete SourceArtifact, quindi **`appspec.yaml`** viene inserita la verifica.
5. In Aggiorna dinamicamente l'immagine di definizione dell'attività, in Input Artifact with Image URI, MyImagescegli, quindi inserisci il testo segnaposto utilizzato nel file:. `taskdef.json` **`IMAGE1_NAME`** Selezionare Salva.
6. Nel AWS CodePipeline riquadro, scegliete Salva modifica alla pipeline, quindi scegliete Salva modifica. Visualizza la pipeline aggiornata.

   Dopo la creazione di questa pipeline di esempio, la configurazione dell'operazione per le voci della console viene visualizzata nella struttura della pipeline come segue:

```
"configuration": {
  "AppSpecTemplateArtifact": "SourceArtifact",
  "AppSpecTemplatePath": "appspec.yaml",
  "TaskDefinitionTemplateArtifact": "SourceArtifact",
  "TaskDefinitionTemplatePath": "taskdef.json",
  "ApplicationName": "codedeployapp",
  "DeploymentGroupName": "codedeploydeplgroup",
  "Image1ArtifactName": "MyImage",
  "Image1ContainerName": "IMAGE1_NAME"
},
```

7. Per inviare le modifiche e avviare una compilazione tramite pipeline, scegliere Release change (Rilascia modifica) e quindi scegliere Release (Rilascia).
8. Scegli l'azione di distribuzione in cui visualizzarla CodeDeploy e vedere l'avanzamento dello spostamento del traffico.

> **Note**
>
> È possibile che venga visualizzato un passaggio di distribuzione che mostra un tempo di attesa opzionale. Per impostazione predefinita, CodeDeploy attende un'ora dopo una corretta distribuzione prima di terminare il set di attività originale. È possibile utilizzare questo tempo per ripristinare o terminare l'attività. La distribuzione viene completata al termine del set di attività.

Fase 7: modifica della pipeline e verifica della distribuzione

Apporta una modifica alla tua immagine e poi trasferisci la modifica al tuo repository Amazon ECR. In questo modo viene attivata la pipeline per l'esecuzione. Verifica che l'origine dell'immagine venga distribuita.

Tutorial: creazione di una pipeline che distribuisce una competenza Amazon Alexa

In questo tutorial, è possibile configurare una pipeline che distribuisce in modo continuo le competenze Alexa utilizzando Alexa Skills Kit come provider di operazioni di distribuzione nella fase di distribuzione. La pipeline completata rileva le modifiche alla competenza quando si modificano i file di origine nel repository di origine. La pipeline usa quindi Alexa Skills Kit per distribuire alla fase di sviluppo di competenze Alexa.

> **Note**
>
> Questa funzionalità non è disponibile nella regione Asia Pacifico (Hong Kong) o Europa (Milano). Per utilizzare altre azioni di distribuzione disponibili in quella regione, consultaIntegrazioni di operazioni di distribuzione.

Per creare un'abilità personalizzata come funzione Lambda, consulta Ospitare un'abilità personalizzata come funzione Lambda AWS. Puoi anche creare una pipeline che utilizza i file sorgente Lambda e CodeBuild un progetto per implementare le modifiche a Lambda per le tue competenze. Ad esempio, per creare una nuova competenza e la funzione Lambda correlata, puoi creare un progetto AWS CodeStar . Consulta Creazione di un progetto di competenze Alexa in AWS CodeStar. Per questa opzione, la pipeline include una terza fase di creazione con un' CodeBuildazione e un'azione nella fase di distribuzione per. AWS CloudFormation

Prerequisiti

Devi avere già quanto segue:

- Un CodeCommit repository. È possibile utilizzare il AWS CodeCommit repository in cui è stato creato. Tutorial: crea una pipeline semplice (CodeCommitrepository)
- Un account sviluppatore Amazon. Questo è l'account che possiede le competenze Alexa. È possibile creare un account gratuitamente nell'Alexa Skills Kit.

- Una competenza Alexa. È possibile creare una competenza di esempio utilizzando il tutorial Ottieni il codice di esempio per competenze personalizzate.
- Installa la CLI ASK e configurala utilizzando le tue `ask init` AWS credenziali. Consulta Installazione e inizializzazione di ASK CLI.

Fase 1: creazione di un profilo di sicurezza LWA Alexa Developer Services

In questa sezione crei un profilo di sicurezza da usare con Login with Amazon (LWA). Se hai già un profilo, questa fase può essere ignorata.

- Segui la procedura descritta generate-lwa-tokensper creare un profilo di sicurezza.
- Dopo aver creato il profilo, annotare il Client ID (ID client) e il Client Secret (Segreto client).
- Assicurati di immettere gli Allowed Return URLs (URL restituiti consentiti) come indicato nelle istruzioni. Gli URL consentono al comando dell'interfaccia a riga di comando ASK di reindirizzare le richieste di token di aggiornamento.

Passaggio 2: crea i file sorgente delle abilità Alexa e inviali al tuo repository CodeCommit

In questa sezione, è possibile creare e inviare i file di origine di competenza Alexa nel repository che la pipeline impiega per la fase di origine. Per la competenza creata nella console per sviluppatori di Amazon, devi produrre e inviare quanto segue:

- Un file `skill.json`.
- Una cartella `interactionModel/custom`.

> **Note**
>
> Questa struttura di directory è conforme ai requisiti del formato dei pacchetti di skill di Alexa Skills Kit, come descritto in Formato del pacchetto di competenze. Se la struttura di directory non utilizza il formato dei pacchetti di skill corretto, le modifiche non vengono distribuite correttamente alla console Alexa Skills Kit.

Per creare file di origine per la competenza

1. Recupera l'ID competenza dalla console per sviluppatori Alexa Skills Kit. Usa questo comando:

   ```
   ask api list-skills
   ```

   Individua la competenza in base al nome e copia il relativo ID nel campo `skillId`.

2. Genera un file `skill.json` che contiene i dettagli della competenza. Usa questo comando:

   ```
   ask api get-skill -s skill-ID > skill.json
   ```

3. (Facoltativo) Crea una cartella `interactionModel/custom`.

   Utilizza questo comando per generare il file del modello di interazione all'interno della cartella. Come impostazione, questo tutorial utilizza en-US come lingua locale nel nome del file.

   ```
   ask api get-model --skill-id skill-ID --locale locale >
       ./interactionModel/custom/locale.json
   ```

Per inviare file al tuo repository CodeCommit

1. Invia o carica i file nel tuo CodeCommit repository. Questi file sono l'artefatto di origine creato dalla procedura guidata Create Pipeline (Crea pipeline) per l'operazione di distribuzione in AWS CodePipeline. I file dovrebbero avere questo aspetto nella directory locale:

   ```
   skill.json
   /interactionModel
     /custom
       |en-US.json
   ```

2. Scegli un metodo per caricare i file:

   a. Per usare la riga di comando Git da un repository clonato sul computer locale:

      i. Esegui il comando seguente per posizionare tutti i file contemporaneamente:

         ```
         git add -A
         ```

      ii. Esegui il comando seguente per eseguire il commit dei file con un messaggio di commit:

```
git commit -m "Added Alexa skill files"
```

iii. Esegui il comando seguente per inviare i file dal repository locale al tuo CodeCommit repository:

```
git push
```

b. Per utilizzare la CodeCommit console per caricare i file:

i. Apri la CodeCommit console e scegli il tuo repository dall'elenco Repository.

ii. Seleziona Add file (Aggiungi file), quindi scegli Upload file (Carica file).

iii. Seleziona Choose file (Scegli file), quindi seleziona il file. Conferma la modifica inserendo il tuo nome utente e indirizzo e-mail. Scegliere Commit changes (Applica modifiche).

iv. Ripeti questa fase per ogni file da caricare.

Fase 3: utilizzo dei comandi ASK CLI per creare un token di aggiornamento

CodePipeline utilizza un token di aggiornamento basato sull'ID cliente e sul segreto nel tuo account sviluppatore Amazon per autorizzare le azioni che esegue per tuo conto. In questa sezione, è possibile utilizzare l'interfaccia a riga di comando ASK per creare il token. Usa queste credenziali quando utilizzi la procedura guidata Create Pipeline (Crea pipeline).

Per creare un token di aggiornamento con le credenziali dell'account sviluppatore Amazon

1. Utilizza il seguente comando:

```
ask util generate-lwa-tokens
```

2. Quando richiesto, immetti l'ID e il segreto client come illustrato in questo esempio:

```
? Please type in the client ID:
amzn1.application-client.example112233445566
? Please type in the client secret:
example112233445566
```

3. Viene visualizzata la pagina di accesso del browser. Accedi con le credenziali dell'account sviluppatore Amazon.

4. Torna alla schermata della riga di comando. Il token di accesso e di aggiornamento vengono generati nell'output. Copia il token di aggiornamento restituito nell'output.

Fase 4: creazione della pipeline

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con un' CodeCommit azione in cui gli artefatti di origine sono i file delle abilità di Alexa che supportano la tua abilità.
- Una fase di distribuzione con un'operazione di distribuzione Alexa Skills Kit.

Per creare una pipeline con la procedura guidata

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Scegli la AWS regione in cui desideri creare il progetto e le relative risorse. Il runtime delle competenze Alexa è disponibile solo nelle seguenti regioni:
   - Asia Pacifico (Tokyo)
   - Europa (Irlanda)
   - Stati Uniti orientali (Virginia settentrionale)
   - US West (Oregon)
3. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegliere Create pipeline (Crea pipeline).
4. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyAlexaPipeline`**.
5. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
6. In Ruolo di servizio, scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.
7. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).
8. In Step 2: Add source stage (Fase 2: aggiunta della fase di origine), in Source provider (Provider origine), scegliere AWS CodeCommit. In Nome repository, scegli il nome del CodeCommit

repository in cui hai creato. Fase 1: Creare un CodeCommit repository In Branch name (Nome ramo), scegliere il nome del ramo che contiene l'aggiornamento di codice più recente.

Dopo aver selezionato il nome del repository e il ramo, un messaggio mostra la regola Amazon CloudWatch Events da creare per questa pipeline.

Seleziona Successivo.

9. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora).

   Seleziona Successivo.

10. In Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione):

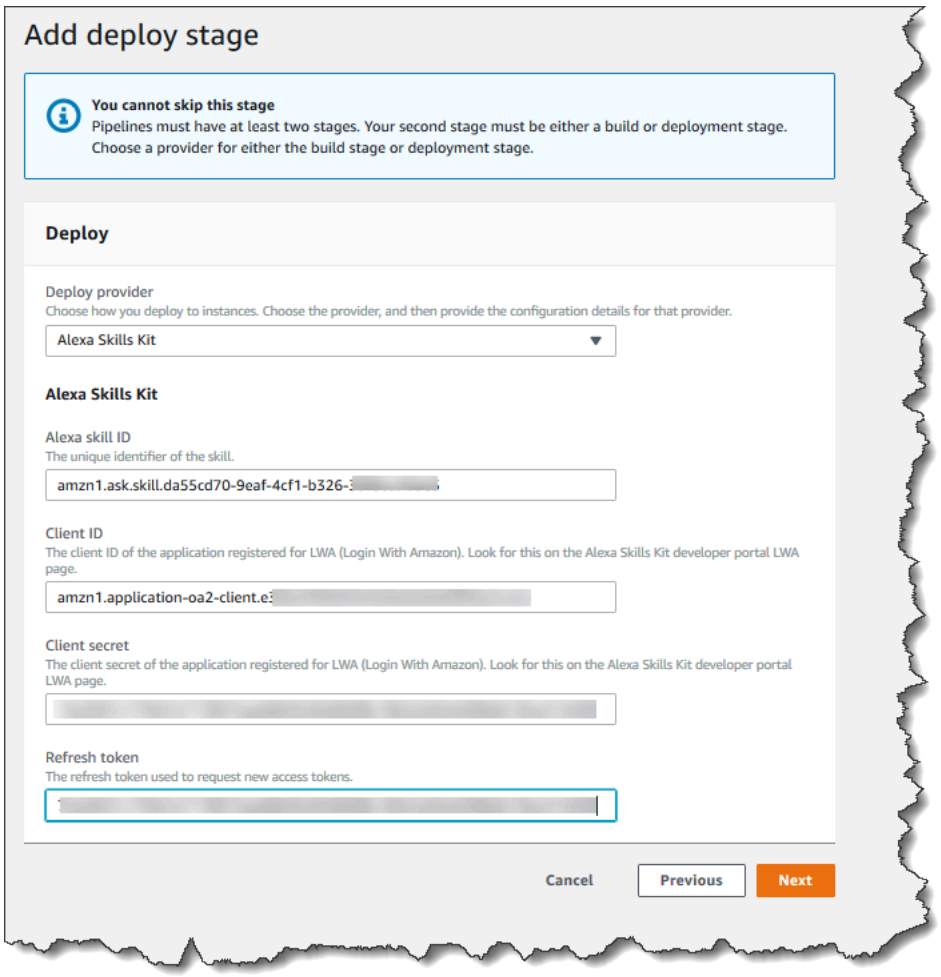
    a. In Deploy provider (Provider di distribuzione), scegli Alexa Skills Kit.

    b. In Alexa skill ID (ID competenza Alexa), immetti l'ID competenza assegnato alla tua competenza nella console per sviluppatori Alexa Skills Kit.

    c. In Client ID (ID client), immetti l'ID dell'applicazione registrata.

    d. In Client secret (Segreto client), immetti il segreto scelto al momento della registrazione.

    e. In Refresh token (Token di aggiornamento), immetti il token generato alla fase 3.



f. Seleziona Successivo.

11. In Step 5: Review (Fase 5: revisione), esaminare le informazioni e quindi scegliere Create pipeline (Crea pipeline).

Fase 5: modifica di un file di origine e verifica della distribuzione

Modifica la competenza e applica la modifica al repository. In questo modo viene attivata la pipeline per l'esecuzione. Verifica che la competenza sia aggiornata nella console per sviluppatori Alexa Skills Kit.

Tutorial: crea una pipeline che utilizzi Amazon S3 come provider di distribuzione

In questo tutorial, configurerai una pipeline che distribuisce continuamente file utilizzando Amazon S3 come provider di azioni di distribuzione nella tua fase di distribuzione. La pipeline completata rileva le modifiche quando si modificano i file di origine nel repository di origine. La pipeline utilizza quindi Amazon S3 per distribuire i file nel bucket. Ogni volta che modifichi o aggiungi i file del tuo sito Web nella posizione di origine, l'implementazione crea il sito Web con i file più recenti.

> **Note**
>
> Anche se elimini file dal repository di origine, l'azione di distribuzione di S3 non elimina gli oggetti S3 corrispondenti ai file eliminati.

Questo tutorial offre due opzioni:

- Crea una pipeline che consente di distribuire un sito Web statico al bucket pubblico S3. Questo esempio crea una pipeline con un'azione di AWS CodeCommit origine e un'azione di distribuzione di Amazon S3. Per informazioni, consulta Opzione 1: distribuire file statici di siti Web su Amazon S3.
- Crea una pipeline che compili il TypeScript codice di esempio JavaScript e distribuisca l'artefatto di CodeBuild output nel bucket S3 per l'archiviazione. Questo esempio crea una pipeline con un'azione di origine di Amazon S3, CodeBuild un'azione di compilazione e un'azione di distribuzione di Amazon S3. Per informazioni, consulta Opzione 2: distribuzione di file di archivio integrati su Amazon S3 da un bucket di origine S3.

> **Important**
>
> Molte delle azioni che aggiungi alla tua pipeline in questa procedura coinvolgono AWS risorse che devi creare prima di creare la pipeline. AWS le risorse per le tue azioni di origine devono sempre essere create nella stessa AWS regione in cui crei la pipeline. Ad esempio, se crei la pipeline nella regione Stati Uniti orientali (Ohio), il tuo CodeCommit repository deve trovarsi nella regione Stati Uniti orientali (Ohio).

Puoi aggiungere azioni interregionali quando crei la pipeline. AWS le risorse per le azioni interregionali devono trovarsi nella stessa AWS regione in cui intendi eseguire l'azione. Per ulteriori informazioni, consulta Aggiungere un'azione interregionale in CodePipeline.

Opzione 1: distribuire file statici di siti Web su Amazon S3

In questo esempio, scarichi il file modello di sito web statico di esempio, carichi i file nel tuo AWS CodeCommit repository, crei il tuo bucket e lo configuri per l'hosting. Successivamente, usi la AWS CodePipeline console per creare la pipeline e specificare una configurazione di distribuzione di Amazon S3.

Prerequisiti

Devi avere già quanto segue:

- Un CodeCommit repository. È possibile utilizzare il AWS CodeCommit repository in cui è stato creato. Tutorial: crea una pipeline semplice (CodeCommitrepository)
- I file di origine per il sito Web statico. Utilizza questo collegamento per scaricare un sito Web statico di esempio. Il download di sample-website.zip produce i file seguenti:
  - Un file `index.html`
  - Un file `main.css`
  - Un file `graphic.jpg`
- Un bucket S3 configurato per l'hosting di siti Web. Consulta Hosting di un sito Web statico su Amazon S3. Accertati di creare il bucket nella stessa regione della pipeline.

> **Note**
>
> Per ospitare un sito Web, il bucket deve disporre di accesso in lettura pubblico, che consente a tutti di avere l'accesso in lettura. Fatta eccezione per l'hosting di siti Web, è consigliabile mantenere le impostazioni predefinite di accesso che bloccano l'accesso pubblico ai bucket S3.

Passaggio 1: invia i file sorgente al tuo repository CodeCommit

In questa sezione, effettua il push dei file di origine al repository che la pipeline impiega per la fase di origine.

Per inviare file al tuo repository CodeCommit

1. Estrai i file di esempio scaricati. Non caricare il file ZIP nel repository.
2. Invia o carica i file nel tuo CodeCommit repository. Questi file sono l'elemento sorgente creato dalla procedura guidata Create Pipeline per l'azione di distribuzione in. CodePipeline I file dovrebbero avere questo aspetto nella directory locale:

```
index.html
main.css
graphic.jpg
```

3. Puoi usare Git o la CodeCommit console per caricare i tuoi file:
   a. Per usare la riga di comando Git da un repository clonato sul computer locale:
      i. Esegui il comando seguente per posizionare tutti i file contemporaneamente:

```
git add -A
```

      ii. Esegui il comando seguente per eseguire il commit dei file con un messaggio di commit:

```
git commit -m "Added static website files"
```

      iii. Esegui il comando seguente per inviare i file dal repository locale al tuo CodeCommit repository:

```
git push
```

   b. Per utilizzare la CodeCommit console per caricare i file:
      i. Apri la CodeCommit console e scegli il tuo repository dall'elenco Repository.
      ii. Seleziona Add file (Aggiungi file), quindi scegli Upload file (Carica file).
      iii. Seleziona Choose file (Scegli file) e vai al file. Conferma la modifica inserendo il tuo nome utente e indirizzo e-mail. Scegliere Commit changes (Applica modifiche).
      iv. Ripeti questa fase per ogni file da caricare.

Fase 2: creazione della pipeline

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con un' CodeCommit azione in cui gli artefatti di origine sono i file del tuo sito web.
- Una fase di distribuzione con un'azione di distribuzione di Amazon S3.

Per creare una pipeline con la procedura guidata

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyS3DeployPipeline`**.
4. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Ruolo di servizio, scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.
6. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).
7. In Step 2: Add source stage (Fase 2: aggiunta della fase di origine), in Source provider (Provider origine), scegliere AWS CodeCommit. In Nome repository, scegli il nome del CodeCommit repository in cui hai creato. Fase 1: Creare un CodeCommit repository In Branch name (Nome ramo), scegliere il nome del ramo che contiene l'aggiornamento di codice più recente. Se l'utente non ha creato personalmente un ramo diverso, è disponibile solo `main`.

   Dopo aver selezionato il nome e il ramo del repository, viene visualizzata la regola Amazon CloudWatch Events da creare per questa pipeline.

   Seleziona Successivo.
8. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora).

Seleziona Successivo.

9. In Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione):

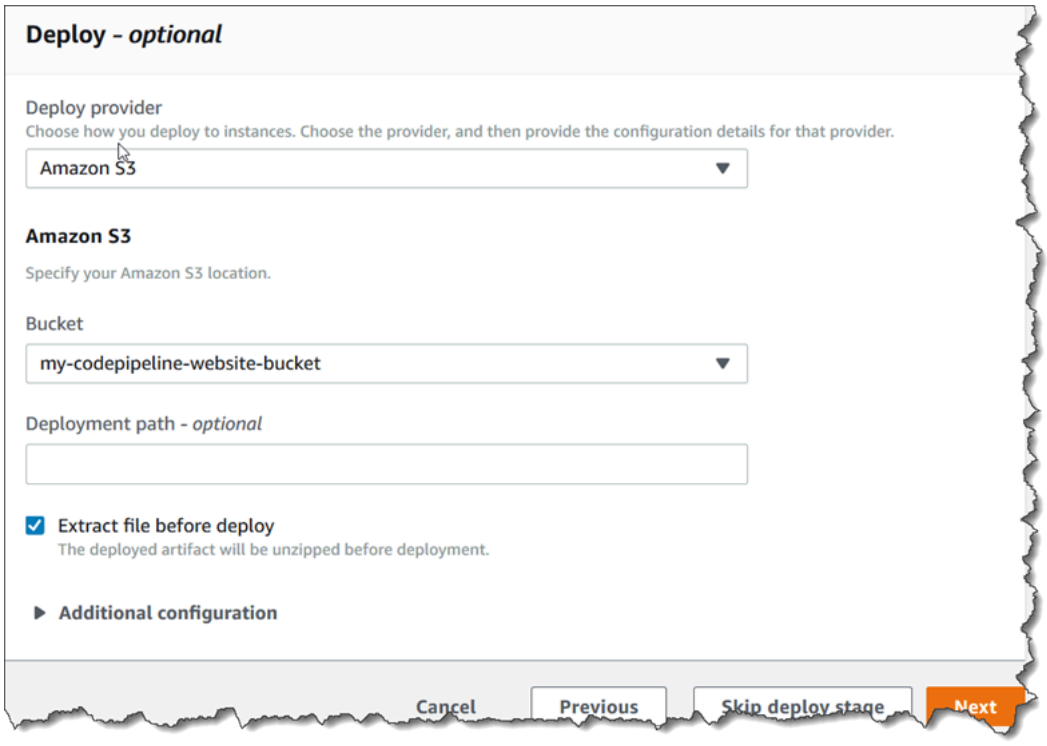
   a. In Deploy provider (Provider di distribuzione), scegli Amazon S3.

   b. In Bucket, inserisci il nome del bucket pubblico.

   c. Seleziona Extract file before deploy (Estrai file prima di distribuire).

   > **Note**
   >
   > La distribuzione non riesce se non selezioni Estrai il file prima di distribuire Questo perché l' AWS CodeCommit azione nella pipeline comprime gli artefatti di origine e il file è un file ZIP.

   Quando Extract file before deploy (Estrai file prima di distribuire) è selezionato, viene visualizzato Deployment path (Percorso di distribuzione). Inserisci il nome del percorso che vuoi utilizzare. Questo crea una struttura di cartelle in Amazon S3 in cui vengono estratti i file. Per questo tutorial, lascia questo campo vuoto.



   d. (Facoltativo) In Canned ACL (ACL predefinita), puoi applicare un set di concessioni predefinite, note come ACL predefinita, agli artefatti caricati.

e. (Facoltativo) In Cache control (Controllo cache), immetti i parametri della memorizzazione nella cache. Con questa impostazione puoi controllare il comportamento della memorizzazione nella cache per le richieste/risposte. Per i valori validi, vedi il campo di intestazione `Cache-Control` per le operazioni HTTP.

f. Seleziona Successivo.

10. In Step 5: Review (Fase 5: revisione), esaminare le informazioni e quindi scegliere Create pipeline (Crea pipeline).

11. Dopo che la pipeline è stata eseguita correttamente, apri la console Amazon S3 e verifica che i file vengano visualizzati nel bucket pubblico come mostrato:

```
index.html
main.css
graphic.jpg
```

12. Accedi all'endpoint per testare il sito Web. L'endpoint segue questo formato: `http://`*`bucket-name`*`.s3-website-`*`region`*`.amazonaws.com/`.

    Endpoint di esempio: `http://my-bucket.s3-website-us-west-2.amazonaws.com/`

    Viene visualizzata la pagina Web di esempio.

Fase 3: modifica di un file di origine e verifica della distribuzione

Modifica i file di origine e applica la modifica al repository. In questo modo viene attivata la pipeline per l'esecuzione. Verifica che il sito Web sia aggiornato.

Opzione 2: distribuzione di file di archivio integrati su Amazon S3 da un bucket di origine S3

In questa opzione, i comandi di compilazione in fase di compilazione compilano il TypeScript codice in JavaScript codice e distribuiscono l'output nel bucket di destinazione S3 in una cartella separata con data e ora. Innanzitutto, create il codice e un file buildspec.yml. TypeScript Dopo aver combinato i file di origine in un file ZIP, caricate il file ZIP di origine nel bucket di origine S3 e utilizzate una CodeBuild fase per distribuire un file ZIP dell'applicazione creato nel bucket di destinazione S3. Il codice compilato viene conservato come un archivio nel bucket di destinazione.

Prerequisiti

Devi avere già quanto segue:

- Un bucket di origine S3. Puoi usare il bucket creato in Tutorial: creazione di una semplice pipeline (bucket S3).
- Un bucket di destinazione S3. Consulta Hosting di un sito Web statico su Amazon S3. Assicurati di creare il bucket nella stessa Regione AWS pipeline che desideri creare.

> **Note**
>
> In questo esempio viene illustrata la distribuzione dei file in un bucket privato. Non abilitare il bucket di destinazione per l'hosting di siti Web o collegare policy che rendono il bucket pubblico.

Fase 1: creazione e caricamento dei file di origine nel bucket di origine S3

In questa sezione, è possibile creare e caricare i file di origine nel bucket che la pipeline impiega per la fase di origine. Questa sezione include istruzioni per creare i seguenti file di origine:

- Un `buildspec.yml` file, che viene utilizzato per CodeBuild creare progetti.
- Un file `index.ts`.

Per creare un file buildspec.yml

- Crea un file denominato `buildspec.yml` con i seguenti contenuti. Questi comandi di compilazione installano TypeScript e utilizzano il TypeScript compilatore per riscrivere il codice in `index.ts` codice. JavaScript

```
version: 0.2

phases:
  install:
    commands:
      - npm install -g typescript
  build:
    commands:
      - tsc index.ts
```

```
artifacts:
  files:
    - index.js
```

Per creare un file index.ts

- Crea un file denominato `index.ts` con i seguenti contenuti.

```
interface Greeting {
    message: string;
}

class HelloGreeting implements Greeting {
    message = "Hello!";
}

function greet(greeting: Greeting) {
    console.log(greeting.message);
}

let greeting = new HelloGreeting();

greet(greeting);
```

Per caricare i file nel bucket di origine S3

1. I file dovrebbero avere questo aspetto nella directory locale:

   ```
   buildspec.yml
   index.ts
   ```

   Comprimi i file e assegna al file il nome `source.zip`.
2. Nella console Amazon S3, per il tuo bucket di origine, scegli Upload. Scegli Add files (Aggiungi file), quindi individua il file ZIP creato.
3. Scegli Carica. Questi file sono l'elemento sorgente creato dalla procedura guidata Create Pipeline per l'azione di distribuzione in. CodePipeline Il file dovrebbe avere questo aspetto nel bucket:

```
source.zip
```

Fase 2: creazione della pipeline

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con un'azione Amazon S3 in cui gli artefatti di origine sono i file per l'applicazione scaricabile.
- Una fase di distribuzione con un'azione di distribuzione di Amazon S3.

Per creare una pipeline con la procedura guidata

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyS3DeployPipeline`**.
4. In Ruolo di servizio, scegli Nuovo ruolo di servizio CodePipeline per consentire la creazione di un ruolo di servizio in IAM.
5. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).
6. In Step 2: Add source stage (Fase 2: aggiunta della fase di origine), in Source provider (Provider di origine), scegli Amazon S3. In Bucket, scegli il nome del bucket di origine. In S3 object key (Chiave oggetto S3), inserisci il nome del file ZIP di origine. Assicurati di includere l'estensione del file.zip.

   Seleziona Successivo.
7. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione):

   a. In Build provider (Provider compilazione), scegli CodeBuild.

   b. Scegliere Create build project (Crea progetto di compilazione). Nella pagina Create project (Crea progetto):

   c. In Project name (Nome progetto) immettere un nome per questo progetto di compilazione.

d. In Environment (Ambiente), scegliere Managed image (Immagine gestita). In Operating system (Sistema operativo), seleziona Ubuntu.

e. In Runtime, seleziona Standard. Per Runtime version (Versione runtime), scegli aws/codebuild/standard:1.0.

f. In Image version (Versione immagine), scegli Always use the latest image for this runtime version (Usa sempre l'immagine più recente per questa versione di runtime).

g. Per Ruolo di servizio, scegli il tuo ruolo di CodeBuild servizio o creane uno.

h. Per Build specifications (Compila specifiche), scegli Use a buildspec file (Usa un file buildspec).

i. Scegli Continua a CodePipeline. Viene visualizzato un messaggio se il progetto è stato creato correttamente.

j. Seleziona Successivo.

8. In Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione):

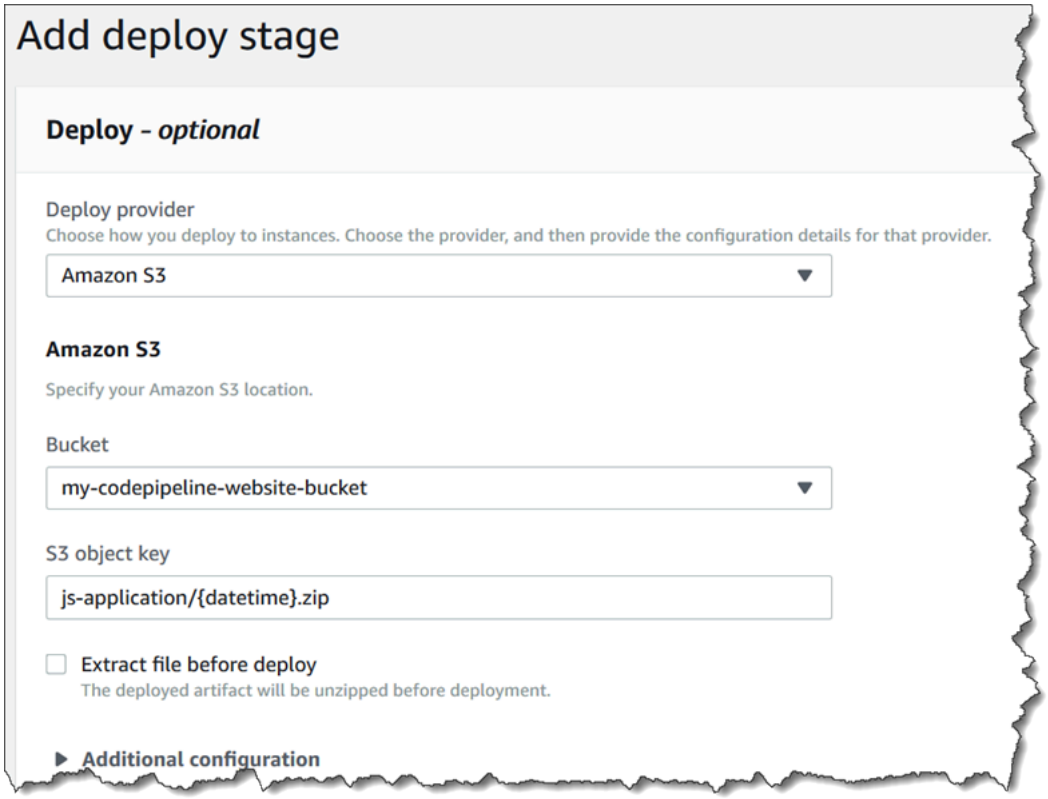
a. In Deploy provider (Provider di distribuzione), scegli Amazon S3.

b. In Bucket, inserisci il nome del bucket di destinazione S3.

c. Assicurati che Extract file before deploy (Estrai file prima di distribuire) sia deselezionato.

Quando Extract file before deploy (Estrai file prima di distribuire) è deselezionato, viene visualizzato S3 object key (Chiave oggetto S3). Inserisci il nome del percorso che vuoi utilizzare: `js-application/{datetime}.zip`.

In questo modo viene creata una `js-application` cartella in Amazon S3 in cui vengono estratti i file. In questa cartella, la variabile `{datetime}` crea un timestamp su ciascun file di output quando la pipeline viene eseguita.



d. (Facoltativo) In Canned ACL (ACL predefinita), puoi applicare un set di concessioni predefinite, note come ACL predefinita, agli artefatti caricati.

e. (Facoltativo) In Cache control (Controllo cache), immetti i parametri della memorizzazione nella cache. Con questa impostazione puoi controllare il comportamento della memorizzazione nella cache per le richieste/risposte. Per i valori validi, vedi il campo di intestazione `Cache-Control` per le operazioni HTTP.

f. Seleziona Successivo.

9. In Step 5: Review (Fase 5: revisione), esaminare le informazioni e quindi scegliere Create pipeline (Crea pipeline).

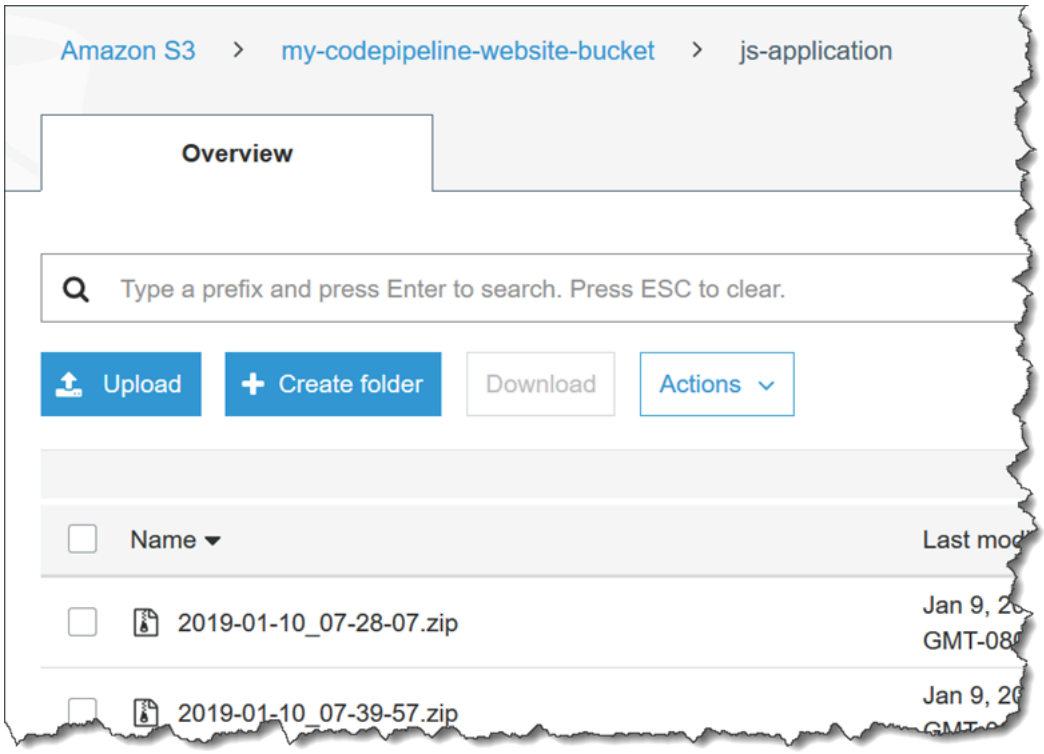
10. Una volta eseguita correttamente la pipeline, visualizza il bucket nella console Amazon S3. Verifica che il file ZIP distribuito sia visualizzato nel bucket di destinazione nella cartella `js-application`. Il JavaScript file contenuto nel file ZIP dovrebbe essere. `index.js` Il file `index.js` contiene il seguente output:

```
var HelloGreeting = /** @class */ (function () {
    function HelloGreeting() {
        this.message = "Hello!";
    }
    return HelloGreeting;
```

```
}());
function greet(greeting) {
    console.log(greeting.message);
}
var greeting = new HelloGreeting();
greet(greeting);
```

Fase 3: modifica di un file di origine e verifica della distribuzione

Modifica i file di origine e caricali nel bucket di origine. In questo modo viene attivata la pipeline per l'esecuzione. Visualizza il bucket di destinazione e verifica che i file di output distribuiti siano disponibili nella cartella `js-application` come illustrato:



Tutorial: crea una pipeline che pubblichi la tua applicazione serverless su AWS Serverless Application Repository

Puoi utilizzarla AWS CodePipeline per fornire continuamente la tua applicazione AWS SAM serverless a. AWS Serverless Application Repository

Questo tutorial mostra come creare e configurare una pipeline per creare un'applicazione serverless ospitata in GitHub e pubblicarla automaticamente. AWS Serverless Application Repository La pipeline

utilizza GitHub come provider di origine e CodeBuild come provider di build. Per pubblicare la tua applicazione serverless su AWS Serverless Application Repository, distribuisci un'applicazione (da AWS Serverless Application Repository) e associ la funzione Lambda creata da quell'applicazione come provider di azioni Invoke nella tua pipeline. Quindi è possibile fornire continuamente aggiornamenti dell'applicazione a AWS Serverless Application Repository, senza scrivere alcun codice.

> **Important**
>
> Molte delle azioni aggiunte alla pipeline in questa procedura coinvolgono AWS risorse che è necessario creare prima di creare la pipeline. AWS le risorse per le tue azioni di origine devono sempre essere create nella stessa AWS regione in cui crei la pipeline. Ad esempio, se crei la pipeline nella regione Stati Uniti orientali (Ohio), il tuo CodeCommit repository deve trovarsi nella regione Stati Uniti orientali (Ohio).
>
> Puoi aggiungere azioni interregionali quando crei la pipeline. AWS le risorse per le azioni interregionali devono trovarsi nella stessa AWS regione in cui intendi eseguire l'azione. Per ulteriori informazioni, consulta Aggiungere un'azione interregionale in CodePipeline.

Prima di iniziare

In questo tutorial si presuppone quanto segue.

- Familiarità con AWS Serverless Application Model (AWS SAM) e AWS Serverless Application Repository.
- Hai un'applicazione serverless ospitata in GitHub che hai pubblicato AWS Serverless Application Repository utilizzando la AWS SAM CLI. Per pubblicare un'applicazione di esempio su AWS Serverless Application Repository, consulta Quick Start: Publishing Applications nella AWS Serverless Application Repository Developer Guide. Per pubblicare la tua applicazione su AWS Serverless Application Repository, consulta Pubblicazione di applicazioni utilizzando la AWS SAM CLI nella Guida per gli AWS Serverless Application Model sviluppatori.

Fase 1: creazione di un file buildspec.yml

Create un `buildspec.yml` file con i seguenti contenuti e aggiungetelo al repository dell' GitHub applicazione serverless. Sostituisci *`template.yml`* con il AWS SAM modello dell'applicazione e il *`bucketname con il bucket S3`* in cui è archiviata l'applicazione in pacchetto.

```
version: 0.2
phases:
  install:
    runtime-versions:
        python: 3.8
  build:
    commands:
      - sam package --template-file template.yml --s3-bucket bucketname --output-
template-file packaged-template.yml
artifacts:
  files:
    - packaged-template.yml
```

Fase 2: creazione e configurazione della pipeline

Segui questi passaggi per creare la tua pipeline nel punto in cui desideri pubblicare l'applicazione serverless. Regione AWS

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. Se necessario, passa al Regione AWS punto in cui desideri pubblicare l'applicazione serverless.
3. Scegliere Create pipeline (Crea pipeline). Nella pagina Choose pipeline settings (Scegli impostazioni della pipeline), in Pipeline name (Nome pipeline), immetti il nome della pipeline.
4. Nel tipo Pipeline, scegliete V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Ruolo di servizio, scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.
6. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).
7. Nella pagina Aggiungi fase di origine, in Provider di origine, scegli GitHub.
8. In Connessione, scegli una connessione esistente o creane una nuova. Per creare o gestire una connessione per l'azione GitHub sorgente, consultaGitHub connessioni.
9. In Repository, scegli il tuo repository GitHub di origine.
10. In Branch, scegli la tua GitHub filiale.
11. Lascia le impostazioni predefinite rimanenti per l'azione di origine. Seleziona Successivo.

12. Nella pagina Add build stage (Aggiungi fase di compilazione), aggiungi una fase di compilazione:

    a. In Build provider (Provider compilazione), scegli AWS CodeBuild. Per Region (Regione), utilizza la regione della pipeline.

    b. Seleziona Crea progetto.

    c. In Project name (Nome progetto) immettere un nome per questo progetto di compilazione.

    d. In Environment image (Immagine ambiente), scegli Managed image (Immagine gestita). In Operating system (Sistema operativo), seleziona Ubuntu.

    e. Per Runtime e Runtime versione (Versione runtime), scegli il runtime e la versione richiesti per l'applicazione serverless.

    f. Per Service Role (Ruolo del servizio), scegli New service role (Nuovo ruolo del servizio).

    g. Per Build specifications (Compila specifiche), scegli Use a buildspec file (Usa un file buildspec).

    h. Scegli Continua a. CodePipeline In questo modo si apre la CodePipeline console e viene creato un CodeBuild progetto che utilizza la console presente `buildspec.yml` nel repository per la configurazione. Il progetto di compilazione utilizza un ruolo di servizio per gestire le Servizio AWS autorizzazioni. Questa operazione potrebbe richiedere un paio di minuti.

    i. Seleziona Next (Successivo).

13. Nella pagina Add deploy stage (Aggiungi fase di distribuzione), scegli Skip deploy stage (Ignora fase di distribuzione), quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Successivo.

14. Scegliere Create pipeline (Crea pipeline). Dovresti visualizzare uno schema che mostra le fasi di origine e compilazione.

15. Concedi al ruolo CodeBuild di servizio l'autorizzazione ad accedere al bucket S3 in cui è archiviata l'applicazione in pacchetto.

    a. Nella fase di creazione della nuova pipeline, scegli. CodeBuild

    b. Seleziona la scheda Build details (Dettagli compilazione).

    c. In Ambiente, scegli il ruolo CodeBuild di servizio per aprire la console IAM.

    d. Espandi la selezione per `CodeBuildBasePolicy` e scegli Edit policy (Modifica policy).

    e. Scegli JSON.

f. Aggiungi una nuova istruzione della policy con i seguenti contenuti. L'istruzione consente di CodeBuild inserire oggetti nel bucket S3 in cui è archiviata l'applicazione in pacchetto. Sostituisci *bucketname* con il nome del bucket S3.

```
{
    "Effect": "Allow",
    "Resource": [
        "arn:aws:s3:::bucketname/*"
    ],
    "Action": [
        "s3:PutObject"
    ]
}
```

g. Scegli Verifica policy.

h. Seleziona Salvataggio delle modifiche.

Fase 3: distribuzione dell'applicazione di pubblicazione

Segui questi passaggi per distribuire l'applicazione che contiene la funzione Lambda che esegue la pubblicazione su. AWS Serverless Application RepositoryQuesta applicazione è aws-serverless-codepipeline-serverlessrepo -publish.

> **Note**
>
> È necessario distribuire l'applicazione nella Regione AWS stessa pipeline.

1. Vai alla pagina delle applicazioni e scegli Deploy (Distribuisci).
2. Seleziona I acknowledge that this app creates custom IAM roles (Sono consapevole che questa app crea ruoli IAM personalizzati).
3. Seleziona Deploy (Implementa).
4. Scegli View AWS CloudFormation Stack per aprire la console. AWS CloudFormation
5. Espandere la sezione Resources (Risorse). Vedi ServerlessRepoPublish, qual è del tipo AWS::Lambda::Function. Prendi nota dell'ID fisico di questa risorsa per la prossima fase. Utilizzi questo ID fisico quando crei la nuova azione di pubblicazione in CodePipeline.

Fase 4: creazione dell'operazione di pubblicazione

Segui questi passaggi per creare l'operazione di pubblicazione nella pipeline.

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. Nella sezione di navigazione a sinistra, scegli la pipeline che desideri modificare.
3. Scegli Modifica.
4. Dopo l'ultima fase della pipeline corrente, scegli + Add stage (+ Aggiungi fase). In Stage name (Nome fase) immetti un nome, ad esempio **`Publish`**, quindi scegli Add stage (Aggiungi fase).
5. Nella nuova fase, scegliere + Add action group (Aggiungi gruppo di operazioni).
6. Immetti un nome per l'operazione. Da Action provider (Provider operazione), in Invoke (Richiama), scegli AWS Lambda.
7. Da Inserisci artefatti, scegli. BuildArtifact
8. Da Nome funzione, scegli l'ID fisico della funzione Lambda che hai annotato nel passaggio precedente.
9. Scegli Save (Salva) per l'operazione.
10. Scegli Done (Fatto) per la fase.
11. In alto a destra, scegli Save (Salva).
12. Per verificare la pipeline, apporta una modifica all'applicazione in. GitHub Ad esempio, modifica la descrizione dell'applicazione nella `Metadata` sezione del file AWS SAM modello. Applica la modifica e inviala alla tua GitHub filiale. In questo modo viene attivata la pipeline per l'esecuzione. Una volta che la pipeline è completa, controlla che l'applicazione sia stata aggiornata con la modifica in AWS Serverless Application Repository.

Tutorial: Utilizzo delle variabili con le azioni di richiamo Lambda

Un'azione di richiamo Lambda può utilizzare variabili di un'altra azione come parte del suo input e restituire nuove variabili insieme al relativo output. Per informazioni sulle variabili per le azioni in CodePipeline, vedi. Variables

Alla fine di questo tutorial, avrai:

- Un'azione di richiamo Lambda che:
  - Consuma la `CommitId` variabile da un'azione di origine CodeCommit
  - Restituisce tre nuove variabili: `dateTime`, `testRunId`, e `region`

- Un'azione di approvazione manuale che utilizza le nuove variabili dell'azione di invocazione Lambda per fornire un URL di test e un ID di esecuzione del test.
- Una pipeline aggiornata con le nuove operazioni

Argomenti

Prerequisiti

Prima di iniziare, devi disporre di quanto segue:

- Puoi creare o utilizzare la pipeline con il codice sorgente inserito. CodeCommit Tutorial: crea una pipeline semplice (CodeCommitrepository)
- Modifica la pipeline esistente in modo che l'azione di CodeCommit origine abbia un namespace. Assegna all'operazione le `SourceVariables` dello spazio dei nomi.

Fase 1: Creazione di una funzione Lambda

Utilizza i seguenti passaggi per creare una funzione Lambda e un ruolo di esecuzione Lambda. L'azione Lambda viene aggiunta alla pipeline dopo aver creato la funzione Lambda.

Per creare una funzione Lambda e un ruolo di esecuzione

1. Accedi AWS Management Console e apri la AWS Lambda console all'indirizzo https://console.aws.amazon.com/lambda/.
2. Scegli Crea funzione. Lasciare selezionato Author from scratch (Crea da zero).
3. In Function name (Nome funzione), immettere il nome della funzione, ad esempio **`myInvokeFunction`**. In Runtime, lasciare selezionata l'opzione predefinita.
4. Espandere Choose or create an execution role (Scegli o crea un ruolo di esecuzione). Scegliere Create a new role with basic Lambda permissions (Crea un nuovo ruolo con le autorizzazioni Lambda di base).
5. Scegli Crea funzione.

6. Per utilizzare una variabile da un'altra operazione, deve essere passata ai `UserParameters` nella configurazione dell'operazione di invocazione Lambda. Configuri l'operazione nella pipeline più avanti in questo tutorial, ma aggiungi il codice presumendo che la variabile venga passata.

```
const commitId =
event["CodePipeline.job"].data.actionConfiguration.configuration.UserParameters;
```

Per produrre nuove variabili, imposta una proprietà denominata `outputVariables` sull'input di `putJobSuccessResult`. Tieni presente che non è possibile produrre variabili come parte di un file `putJobFailureResult`.

```
const successInput = {
  jobId: jobId,
  outputVariables: {
    testRunId: Math.floor(Math.random() * 1000).toString(),
    dateTime: Date(Date.now()).toString(),
    region: lambdaRegion
  }
};
```

Nella nuova funzione, lasciare selezionato Edit code inline (Modifica codice inline) e incollare il seguente codice di esempio in `index.js`.

```
var AWS = require('aws-sdk');

exports.handler = function(event, context) {
    var codepipeline = new AWS.CodePipeline();

    // Retrieve the Job ID from the Lambda action
    var jobId = event["CodePipeline.job"].id;

    // Retrieve the value of UserParameters from the Lambda action configuration in
 CodePipeline,
    // in this case it is the Commit ID of the latest change of the pipeline.
    var params =
 event["CodePipeline.job"].data.actionConfiguration.configuration.UserParameters;

    // The region from where the lambda function is being executed.
    var lambdaRegion = process.env.AWS_REGION;

    // Notify CodePipeline of a successful job
```

```
    var putJobSuccess = function(message) {
        var params = {
            jobId: jobId,
            outputVariables: {
                testRunId: Math.floor(Math.random() * 1000).toString(),
                dateTime: Date(Date.now()).toString(),
                region: lambdaRegion
            }
        };
        codepipeline.putJobSuccessResult(params, function(err, data) {
            if(err) {
                context.fail(err);
            } else {
                context.succeed(message);
            }
        });
    };

    // Notify CodePipeline of a failed job
    var putJobFailure = function(message) {
        var params = {
            jobId: jobId,
            failureDetails: {
                message: JSON.stringify(message),
                type: 'JobFailed',
                externalExecutionId: context.invokeid
            }
        };
        codepipeline.putJobFailureResult(params, function(err, data) {
            context.fail(message);
        });
    };

    var sendResult = function() {
        try {
            console.log("Testing commit - " + params);

            // Your tests here

            // Succeed the job
            putJobSuccess("Tests passed.");
        } catch (ex) {
            // If any of the assertions failed then fail the job
            putJobFailure(ex);
```

```
        }
    };

    sendResult();
};
```

7. Selezionare Salva.
8. Copiare l'Amazon Resource Name (ARN) nella parte superiore dello schermo.
9. Come ultimo passaggio, apri la console AWS Identity and Access Management (IAM) all'indirizzo https://console.aws.amazon.com/iam/. Modifica il ruolo di esecuzione Lambda per aggiungere la seguente politica:. AWSCodePipelineCustomActionAccess Per le fasi della creazione di un ruolo di esecuzione Lambda o della modifica della policy di ruolo, consulta Fase 2: Creare la funzione Lambda .

Passaggio 2: aggiungi un'azione di richiamo Lambda e un'azione di approvazione manuale alla tua pipeline

In questo passaggio, aggiungi un'azione di richiamo Lambda alla tua pipeline. È possibile aggiungere l'operazione come parte di una fase denominata Test. Il tipo di operazione è un'operazione di richiamo. È quindi possibile aggiungere un'operazione di approvazione manuale dopo l'operazione di richiamo.

Per aggiungere un'azione Lambda e un'azione di approvazione manuale alla pipeline

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account. Scegliere la pipeline in cui si desidera aggiungere l'operazione.
2. Aggiungi l'azione di test Lambda alla tua pipeline.

   a. Per modificare la pipeline, scegliere Edit (Modifica). Aggiungere una fase dopo l'operazione di origine nella pipeline esistente. Immettere un nome per la fase, ad esempio **`Test`**.

   b. Nella nuova fase, scegliere l'icona per aggiungere un'operazione. In Action name (Nome operazione), immettere il nome dell'operazione di richiamo, ad esempio **`Test_Commit`**.

   c. In Action provider, scegli. AWS Lambda

   d. In Input artifacts (Artefatti di input), scegliere il nome dell'artefatto di output dell'operazione di origine, ad esempio `SourceArtifact`.

e. In Nome funzione, scegli il nome della funzione Lambda che hai creato.

f. In Parametri utente, inserisci la sintassi della variabile per l'ID di CodeCommit commit. Questa operazione crea la variabile di output che si risolve nel commit da esaminare e approvare ogni volta che viene eseguita la pipeline.

```
#{SourceVariables.CommitId}
```

g. In Variable namespace (Spazio dei nomi delle variabili) aggiungere il nome dello spazio dei nomi, ad esempio **`TestVariables`**.

h. Seleziona Fatto.

3. Aggiungere l'operazione di approvazione manuale alla pipeline.

a. Con la pipeline ancora in modalità di modifica, aggiungere una fase dopo l'operazione di richiamo. Immettere un nome per la fase, ad esempio **`Approval`**.

b. Nella nuova fase, scegliere l'icona per aggiungere un'operazione. In Action name (Nome operazione), immettere il nome dell'operazione di approvazione, ad esempio **`Change_Approval`**.

c. In Action provider (Provider operazione), scegliere Manual approval (Approvazione manuale).

d. In URL for review (URL per revisione), costruire l'URL aggiungendo la sintassi della variabile per le variabili `region` e `CommitId`. Assicurarsi di utilizzare gli spazi dei nomi assegnati alle operazioni che forniscono le variabili di output.

Per questo esempio, l'URL con la sintassi variabile per un' CodeCommit azione ha lo spazio dei nomi predefinito. `SourceVariables` La variabile di output della regione Lambda ha lo spazio dei nomi `TestVariables`. L'URL ha il seguente aspetto:

```
https://#{TestVariables.region}.console.aws.amazon.com/codesuite/codecommit/
repositories/MyDemoRepo/commit/#{SourceVariables.CommitId}
```

In Comments (Commenti), costruire il testo del messaggio di approvazione aggiungendo la sintassi della variabile per la variabile `testRunId`. Per questo esempio, l'URL con la sintassi della variabile di output `testRunId` Lambda ha lo spazio dei nomi `TestVariables`. Inserire il seguente messaggio.

```
Make sure to review the code before approving this action. Test Run ID:
 #{TestVariables.testRunId}
```

4. Scegliere Done (Fatto) per chiudere la schermata di modifica per l'operazione, quindi scegliere Done (Fatto) per chiudere la schermata di modifica per la fase. Per salvare la pipeline, scegliere Done (Fatto). La pipeline completata ora contiene una struttura con fasi di origine, test, approvazione e distribuzione.

   Scegliere Release change (Rilascia modifica) per eseguire l'ultima modifica nella struttura della pipeline.

5. Quando la pipeline raggiunge la fase di approvazione manuale, scegliere Review (Revisione). Le variabili risolte vengono visualizzate come URL per l'ID commit. L'approvatore può scegliere l'URL per visualizzare il commit.
6. Dopo che la pipeline è stata eseguita correttamente, è anche possibile visualizzare i valori delle variabili nella pagina della cronologia dell'esecuzione delle operazioni.

Tutorial: utilizzare un'azione di AWS Step Functions richiamo in una pipeline

È possibile utilizzare AWS Step Functions per creare e configurare macchine a stati. Questo tutorial illustra come aggiungere un'operazione di richiamo a una pipeline che attiva le esecuzioni della macchina a stati dalla pipeline.

In questo tutorial, vengono effettuate le seguenti operazioni:

- Crea una macchina a stati standard in AWS Step Functions.
- Immettere direttamente l'input JSON della macchina a stati. Puoi anche caricare il file di input della macchina a stati in un bucket Amazon Simple Storage Service (Amazon S3).
- Aggiornare la pipeline aggiungendo l'operazione della macchina a stati.

Argomenti

Prerequisito: creare o scegliere una pipeline semplice

In questo tutorial viene aggiunta un'operazione di richiamo su una pipeline esistente. È possibile utilizzare la pipeline creata in Tutorial: creazione di una semplice pipeline (bucket S3) o Tutorial: crea una pipeline semplice (CodeCommitrepository).

Si utilizza una pipeline esistente con un'operazione di origine e almeno una struttura a due fasi, ma non si utilizzano gli artefatti di origine per questo esempio.

> **Note**
>
> Potrebbe essere necessario aggiornare il ruolo del servizio utilizzato dalla pipeline con autorizzazioni aggiuntive necessarie per eseguire questa operazione. A tale scopo, apri la console AWS Identity and Access Management (IAM), trova il ruolo, quindi aggiungi le autorizzazioni alla policy del ruolo. Per ulteriori informazioni, consulta Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline.

Fase 1: creazione della macchina a stati di esempio

Nella console Step Functions, create una macchina a stati utilizzando il modello `HelloWorld` di esempio. Per istruzioni, consulta Creare una macchina a stati nella Guida per gli AWS Step Functions sviluppatori.

Passaggio 2: aggiungi un'azione di invocazione Step Functions alla tua pipeline

Aggiungi un'azione di richiamo Step Functions alla tua pipeline come segue:

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegliere il nome della pipeline da modificare. Questa operazione apre una visualizzazione dettagliata della pipeline. compreso lo stato di ciascuna delle operazioni in ciascuna fase della pipeline.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).

4. Nella seconda fase della pipeline semplice, scegliere Edit stage (Modifica fase). Scegli Elimina. Questo elimina la seconda fase ora che non ne hai più bisogno.
5. In fondo al diagramma, scegliere + Add stage (+ Aggiungi fase)
6. In Stage name (Nome fase), inserire un nome per la fase, ad esempio **`Invoke`**, e poi scegliere Add stage (Aggiungi fase).
7. Scegliere + Add action group (+ Aggiungi gruppo di operazioni).
8. Alla voce Action name (Nome operazione), inserire un nome, ad esempio **`Invoke`**.
9. Nel provider Action, scegli AWS Step Functions. Consenti a Region (Regione) di preimpostarsi sulla regione della pipeline.
10. In Input artifacts (Artefatti di input), scegliere `SourceArtifact`.
11. In State machine ARN (ARN macchina a stati), scegliere l'Amazon Resource Name (ARN) per la macchina a stati creata in precedenza.
12. (Facoltativo) In Execution name prefix (Prefisso nome esecuzione), immettere un prefisso da aggiungere all'ID di esecuzione della macchina a stati.
13. In Input type (Tipo di input), scegliere Literal (Letterale).
14. In Input, immettere il codice JSON di input previsto dalla macchina a stati di esempio `HelloWorld`.

    > **Note**
    >
    > L'input per l'esecuzione della macchina a stati è diverso dal termine usato CodePipeline per descrivere gli artefatti di input per le azioni.

    In questo esempio, inserire il seguente codice JSON:

    ```
    {"IsHelloWorldExample": true}
    ```

15. Seleziona Fatto.
16. Nello fase che si sta modificando, scegliere Done (Fatto). Nel riquadro AWS CodePipeline , scegli Save (Salva) e quindi scegli Save (Salva) sul messaggio di avviso.
17. Per inviare le modifiche e avviare l'esecuzione della pipeline, scegli Release change (Rilascia modifica) e quindi scegli Release (Rilascia).
18. Sulla pipeline completata, scegli AWS Step Functions nell'azione di invoca. Nella AWS Step Functions console, visualizza l'ID di esecuzione della tua macchina a stati. L'ID mostra il nome

della macchina a stati `HelloWorld` e l'ID di esecuzione della macchina a stati con il prefisso `my-prefix`.

```
arn:aws:states:us-west-2:account-ID:execution:HelloWorld:my-
prefix-0d9a0900-3609-4ebc-925e-83d9618fcca1
```

Tutorial: crea una pipeline da utilizzare AWS AppConfig come provider di distribuzione

In questo tutorial, configurerai una pipeline che fornisce continuamente file di configurazione da utilizzare AWS AppConfig come provider di azioni di distribuzione nella fase di distribuzione.

Argomenti

Prerequisiti

Prima di iniziare, devi completare quanto segue:

- Questo esempio utilizza una sorgente S3 per la pipeline. Crea o usa un bucket Amazon S3 con il controllo delle versioni abilitato. Per creare un bucket S3, segui le istruzioni contenute in Fase 1: creazione di un bucket S3 per l'applicazione.

Fase 1: Crea le tue risorse AWS AppConfig

In questa sezione, crei le seguenti risorse:

- Un'applicazione in AWS AppConfig è un'unità logica di codice che fornisce funzionalità ai clienti.
- Un ambiente in AWS AppConfig è un gruppo di AppConfig destinazione di distribuzione logica, ad esempio applicazioni in un ambiente beta o di produzione.

- Un profilo di configurazione è una raccolta di impostazioni che influenzano il comportamento dell'applicazione. Il profilo di configurazione consente AWS AppConfig di accedere alla configurazione nella posizione memorizzata.
- (Facoltativo) Una strategia di distribuzione AWS AppConfig definisce il comportamento di una distribuzione di configurazione, ad esempio la percentuale di client che deve ricevere la nuova configurazione distribuita in un dato momento durante una distribuzione.

Per creare un'applicazione, un ambiente, un profilo di configurazione e una strategia di distribuzione

1. Accedi alla AWS Management Console.
2. Utilizza i passaggi descritti nei seguenti argomenti per creare le tue risorse in AWS AppConfig.
   - Crea un'applicazione.
   - Crea un ambiente.
   - Crea un profilo AWS CodePipeline di configurazione.
   - (Facoltativo) Scegli una strategia di implementazione predefinita o creane una personalizzata.

Passaggio 2: carica i file nel tuo bucket di origine S3

In questa sezione, crea uno o più file di configurazione. Quindi comprimi e invia i file sorgente nel bucket utilizzato dalla pipeline per la fase di origine.

Per creare file di configurazione

1. Crea un `configuration.json` file per ogni configurazione in ogni regione. Includi i seguenti contenuti:

   ```
   Hello World!
   ```

2. Utilizza i seguenti passaggi per comprimere e caricare i file di configurazione.

Per comprimere e caricare i file sorgente

1. Crea un file.zip con i tuoi file e assegna un nome al file.zip. `configuration-files.zip` Ad esempio, il tuo file.zip può utilizzare la seguente struttura:

   ```
   .
   ```

```
### appconfig-configurations
    ### MyConfigurations
        ### us-east-1
        #   ### configuration.json
        ### us-west-2
            ### configuration.json
```

2. Nella console Amazon S3 del tuo bucket, scegli Carica e segui le istruzioni per caricare il tuo file.zip.

Fase 3: creazione della pipeline

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con un'azione Amazon S3 in cui gli artefatti di origine sono i file per la configurazione.
- Una fase di distribuzione con un' AppConfig azione di distribuzione.

Per creare una pipeline con la procedura guidata

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyAppConfigPipeline`**.
4. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Ruolo di servizio, scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.
6. Lasciare i valori predefiniti delle impostazioni in Advanced settings (Impostazioni avanzate), quindi scegliere Next (Successivo).
7. In Step 2: Add source stage (Fase 2: aggiunta della fase di origine), in Source provider (Provider di origine), scegli Amazon S3. In Bucket, scegli il nome del tuo bucket di origine S3.

   Nella chiave dell'oggetto S3, inserisci il nome del tuo file.zip:. `configuration-files.zip`

Seleziona Successivo.

8. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora).

   Seleziona Successivo.

9. In Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione):

   a. In Deploy provider (Provider di distribuzione), scegliere AWS AppConfig.

   b. In Applicazione, scegli il nome dell'applicazione in cui hai creato. AWS AppConfig Il campo mostra l'ID dell'applicazione.

   c. In Ambiente, scegli il nome dell'ambiente in cui hai creato AWS AppConfig. Il campo mostra l'ID del tuo ambiente.

   d. In Profilo di configurazione, scegli il nome del profilo di configurazione in cui hai creato AWS AppConfig. Il campo mostra l'ID del tuo profilo di configurazione.

   e. In Strategia di implementazione, scegli il nome della tua strategia di distribuzione. Può trattarsi di una strategia di distribuzione creata in precedenza AppConfig o di una strategia di distribuzione scelta tra quelle predefinite. AppConfig Il campo mostra l'ID della tua strategia di distribuzione.

   f. In Input artifact configuration path, inserisci il percorso del file. Assicurati che il percorso di configurazione degli artefatti di input corrisponda alla struttura di directory nel file.zip del bucket S3. Per questo esempio, inserisci il seguente percorso del file:. `appconfig-configurations/MyConfigurations/us-west-2/configuration.json`

   g. Seleziona Successivo.

10. In Step 5: Review (Fase 5: revisione), esaminare le informazioni e quindi scegliere Create pipeline (Crea pipeline).

Passo 4: Apporta una modifica a qualsiasi file sorgente e verifica la distribuzione

Apporta una modifica ai file sorgente e carica la modifica nel tuo bucket. In questo modo viene attivata la pipeline per l'esecuzione. Verifica che la configurazione sia disponibile visualizzando la versione.

Tutorial: usa il clone completo con una sorgente di GitHub pipeline

Puoi scegliere l'opzione di clonazione completa per la tua azione di GitHub origine in. CodePipeline Usa questa opzione per eseguire CodeBuild comandi per i metadati Git nell'azione di creazione della pipeline.

In questo tutorial, creerai una pipeline che si connette al tuo GitHub repository, utilizza l'opzione full clone per i dati di origine ed CodeBuild eseguirai una build che clona il tuo repository ed esegue comandi Git per il repository.

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni di Asia Pacifico (Hong Kong), Africa (Città del Capo), Medio Oriente (Bahrein), Europa (Zurigo) o (Stati Uniti occidentali). AWS GovCloud Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Argomenti

Prerequisiti

Prima di iniziare è necessario:

- Crea un GitHub repository con il tuo GitHub account.
- Tieni a portata di mano GitHub le tue credenziali. Quando utilizzi il AWS Management Console per configurare una connessione, ti viene chiesto di accedere con GitHub le tue credenziali.

Fase 1: Creare un file README

Dopo aver creato il GitHub repository, segui questi passaggi per aggiungere un file README.

1. Accedi al tuo GitHub repository e scegli il tuo repository.
2. Per creare un nuovo file, scegli Aggiungi file > Crea nuovo file. Assegna un nome al file`README.md`. file e aggiungi il testo seguente.

   ```
   This is a GitHub repository!
   ```

3. Scegliere Commit changes (Applica modifiche).

   Assicurati che il file `README.md` si trovi al livello root del repository.

Fase 2: Crea la tua pipeline e crea il progetto

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con una connessione al GitHub repository e all'azione.
- Una fase di compilazione con un'azione di AWS CodeBuild compilazione.

Per creare una pipeline con la procedura guidata

1. Accedi alla CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyGitHubPipeline`**.
4. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Service Role (Ruolo del servizio), scegliere New service role (Nuovo ruolo del servizio).

   > **Note**
   >
   > Se invece scegli di utilizzare il tuo ruolo di CodePipeline servizio esistente, assicurati di aver aggiunto l'autorizzazione `codeconnections:UseConnection` IAM alla tua policy

sul ruolo di servizio. Per istruzioni sul ruolo CodePipeline di servizio, consulta Aggiungere autorizzazioni al ruolo CodePipeline di servizio.

6. In Impostazioni avanzate non modificare le impostazioni predefinite. In Artifact store (Archivio artefatti), seleziona Default location (Posizione predefinita) per utilizzare l'archivio artefatti predefinito, ad esempio il bucket Amazon S3 dedicato agli artefatti designato come predefinito, per la pipeline nella regione selezionata.

> **Note**
>
> Non si tratta del bucket di origine per il codice sorgente, ma dell'archivio artefatti per la pipeline. È richiesto un archivio artefatti separato, ad esempio un bucket S3, per ogni pipeline.

Seleziona Next (Successivo).

7. Nella Fase 2: Aggiungi una fase di origine, aggiungi una fase di origine:

   a. In Provider di origine, scegli GitHub (versione 2).

   b. In Connessione, scegli una connessione esistente o creane una nuova. Per creare o gestire una connessione per l'azione GitHub sorgente, consultaGitHub connessioni.

   c. In Nome archivio, scegli il nome del tuo GitHub repository.

   d. In Nome del ramo, scegli il ramo del repository che desideri utilizzare.

   e. Assicurati che l'opzione Avvia la pipeline alla modifica del codice sorgente sia selezionata.

   f. In Formato artefatto di output, scegli Clone completo per abilitare l'opzione Git clone per il repository di origine. Solo le azioni fornite da CodeBuild possono utilizzare l'opzione Git clone. Passaggio 3: Aggiornare la politica del ruolo CodeBuild di servizio per utilizzare le connessioniIn questo tutorial utilizzerai questa opzione per aggiornare le autorizzazioni per il tuo ruolo di CodeBuild project service.

Seleziona Next (Successivo).

8. In Add build stage (Aggiungi fase di compilazione), aggiungi una fase di compilazione:

   a. In Build provider (Provider compilazione), scegli AWS CodeBuild. Consenti a Region (Regione) di preimpostarsi sulla regione della pipeline.

   b. Seleziona Crea progetto.

c. In Project name (Nome progetto) immettere un nome per questo progetto di compilazione.

d. In Environment image (Immagine ambiente), scegli Managed image (Immagine gestita). In Operating system (Sistema operativo), seleziona Ubuntu.

e. In Runtime, seleziona Standard. Per Immagine, scegli aws/codebuild/standard:5.0.

f. Per Service Role (Ruolo del servizio), scegli New service role (Nuovo ruolo del servizio).

> **Note**
>
> Annota il nome del tuo ruolo di CodeBuild servizio. Avrai bisogno del nome del ruolo per l'ultimo passaggio di questo tutorial.

g. In Buildspec, per Build specifications (Specifiche di compilazione), scegli Insert build commands (Inserisci comandi di compilazione). Scegli Passa all'editor e incolla quanto segue nei comandi di creazione.

> **Note**
>
> Nella `env` sezione delle specifiche di compilazione, assicurati che l'helper delle credenziali per i comandi git sia abilitato come mostrato in questo esempio.

```
version: 0.2

env:
  git-credential-helper: yes
phases:
  install:
    #If you use the Ubuntu standard image 2.0 or later, you must specify
 runtime-versions.
    #If you specify runtime-versions and use an image other than Ubuntu
 standard image 2.0, the build fails.
    runtime-versions:
      nodejs: 12
      # name: version
    #commands:
      # - command
      # - command
  pre_build:
    commands:
```

```
      - ls -lt
      - cat README.md
  build:
    commands:
      - git log | head -100
      - git status
      - ls
      - git archive --format=zip HEAD > application.zip
  #post_build:
    #commands:
      # - command
      # - command
artifacts:
  files:
    - application.zip
    # - location
  #name: $(date +%Y-%m-%d)
  #discard-paths: yes
  #base-directory: location
#cache:
  #paths:
    # - paths
```

h. Scegli Continua a. CodePipeline Questo ritorna alla CodePipeline console e crea un CodeBuild progetto che utilizza i comandi di compilazione per la configurazione. Il progetto di compilazione utilizza un ruolo di servizio per gestire le Servizio AWS autorizzazioni. Questa operazione potrebbe richiedere un paio di minuti.

i. Seleziona Next (Successivo).

9. Nella pagina Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione), scegli Skip deploy stage (Ignora fase di distribuzione), quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Next (Successivo).
10. Nella Step 5: Review (Fase 5: revisione), scegliere Create pipeline (Crea pipeline).

Passaggio 3: Aggiornare la politica del ruolo CodeBuild di servizio per utilizzare le connessioni

L'esecuzione iniziale della pipeline avrà esito negativo perché il ruolo CodeBuild di servizio deve essere aggiornato con le autorizzazioni per utilizzare le connessioni. Aggiungi l'autorizzazione `codeconnections:UseConnection` IAM alla tua policy sui ruoli di servizio. Per istruzioni su

come aggiornare la policy nella console IAM, consultaAggiungi le autorizzazioni per le connessioni a Bitbucket, Enterprise Server o.com CodeBuild GitClone GitHub GitHub GitLab.

Passaggio 4: Visualizza i comandi del repository nell'output della build

1. Quando il ruolo di servizio viene aggiornato correttamente, scegli Riprova nella fase non riuscita CodeBuild .
2. Dopo che la pipeline è stata eseguita correttamente, nella fase di compilazione corretta, scegli Visualizza dettagli.

   Nella pagina dei dettagli, scegli la scheda Registri. Visualizza l'output della CodeBuild build. I comandi restituiscono il valore della variabile inserita.

   I comandi generano il contenuto del `README.md` file, elencano i file nella directory, clonano il repository, visualizzano il registro e archiviano il repository come file ZIP.

Tutorial: usa il clone completo con una sorgente di CodeCommit pipeline

Puoi scegliere l'opzione di clonazione completa per la tua azione di CodeCommit origine in. CodePipeline Usa questa opzione per consentire l'accesso CodeBuild ai metadati Git nell'azione di creazione della pipeline.

In questo tutorial, crei una pipeline che accede al tuo CodeCommit repository, utilizza l'opzione full clone per i dati di origine ed esegue una CodeBuild build che clona il tuo repository ed esegue comandi Git per il repository.

> **Note**
>
> CodeBuild le azioni sono le uniche azioni a valle che supportano l'uso dei metadati Git disponibili con l'opzione Git clone. Inoltre, sebbene la pipeline possa contenere azioni tra più account, l' CodeCommitazione e l' CodeBuild azione devono trovarsi nello stesso account affinché l'opzione di clonazione completa abbia successo.

Argomenti

- Prerequisiti

Prerequisiti

Prima di iniziare, devi creare un CodeCommit repository nello stesso AWS account e nella stessa regione della pipeline.

Passaggio 1: creare un file README

Segui questi passaggi per aggiungere un file README al tuo repository di origine. Il file README fornisce un file sorgente di esempio da leggere per l'azione a CodeBuild valle.

Per aggiungere un file README

1. Accedi al tuo repository e scegli il tuo repository.
2. Per creare un nuovo file, scegli Aggiungi file > Crea file. Assegna un nome al file`README.md`. file e aggiungi il testo seguente.

   ```
   This is a CodeCommit repository!
   ```
3. Scegliere Commit changes (Applica modifiche).

   Assicurati che il file `README.md` si trovi al livello root del repository.

Fase 2: Crea la tua pipeline e crea il progetto

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con un'azione CodeCommit di origine.
- Una fase di compilazione con un'azione di AWS CodeBuild compilazione.

Per creare una pipeline con la procedura guidata

1. Accedi alla CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyCodeCommitPipeline`**.
4. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Service role (Ruolo del servizio), procedere in uno dei seguenti modi:
   - Scegli Existing service role (Ruolo di servizio esistente).
   - Scegliete il vostro ruolo di CodePipeline servizio esistente. Questo ruolo deve disporre dell'autorizzazione `codecommit:GetRepository` IAM per la tua politica relativa al ruolo di servizio. Vedi Aggiungere autorizzazioni al ruolo di CodePipeline servizio.
6. In Impostazioni avanzate non modificare le impostazioni predefinite. Seleziona Successivo.
7. Nella pagina Passaggio 2: Aggiungi fase di origine, procedi come segue:

   a. In Source provider (Provider origine), scegliere CodeCommit.

   b. In Nome archivio, scegli il nome del tuo repository.

   c. In Nome del ramo, scegli il nome del tuo ramo.

   d. Assicurati che l'opzione Avvia la pipeline alla modifica del codice sorgente sia selezionata.

   e. In Formato artefatto di output, scegli Clone completo per abilitare l'opzione Git clone per il repository di origine. Solo le azioni fornite da CodeBuild possono utilizzare l'opzione Git clone.

   Seleziona Successivo.
8. Nella fase Aggiungi compilazione, procedi come segue:

   a. In Build provider (Provider compilazione), scegli AWS CodeBuild. Consenti a Region (Regione) di preimpostarsi sulla regione della pipeline.

   b. Seleziona Crea progetto.

   c. In Project name (Nome progetto) immettere un nome per questo progetto di compilazione.

   d. In Environment image (Immagine ambiente), scegli Managed image (Immagine gestita). In Operating system (Sistema operativo), seleziona Ubuntu.

   e. In Runtime, seleziona Standard. Per Immagine, scegli aws/codebuild/standard:5.0.

f. Per Service Role (Ruolo del servizio), scegli New service role (Nuovo ruolo del servizio).

> **Note**
>
> Annota il nome del tuo ruolo CodeBuild di servizio. Avrai bisogno del nome del ruolo per l'ultimo passaggio di questo tutorial.

g. In Buildspec, per Build specifications (Specifiche di compilazione), scegli Insert build commands (Inserisci comandi di compilazione). Scegli Passa all'editor, quindi in Comandi di creazione incolla il codice seguente.

```
version: 0.2

env:
  git-credential-helper: yes
phases:
  install:
    #If you use the Ubuntu standard image 2.0 or later, you must specify
 runtime-versions.
    #If you specify runtime-versions and use an image other than Ubuntu
 standard image 2.0, the build fails.
    runtime-versions:
      nodejs: 12
      # name: version
    #commands:
      # - command
      # - command
  pre_build:
    commands:
      - ls -lt
      - cat README.md
  build:
    commands:
      - git log | head -100
      - git status
      - ls
      - git describe --all
  #post_build:
    #commands:
      # - command
      # - command
#artifacts:
```

```
  #files:
    # - location
  #name: $(date +%Y-%m-%d)
  #discard-paths: yes
  #base-directory: location
#cache:
  #paths:
    # - paths
```

h. Scegli Continua a CodePipeline. Questo ti riporta alla CodePipeline console e crea un CodeBuild progetto che utilizza i comandi di build per la configurazione. Il progetto di compilazione utilizza un ruolo di servizio per gestire le Servizio AWS autorizzazioni. Questa operazione potrebbe richiedere un paio di minuti.

i. Seleziona Next (Successivo).

9. Nella pagina Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione), scegli Skip deploy stage (Ignora fase di distribuzione), quindi accetta il messaggio di avviso scegliendo Skip (Ignora). Seleziona Next (Successivo).
10. Nella Step 5: Review (Fase 5: revisione), scegliere Create pipeline (Crea pipeline).

Fase 3: Aggiornare la policy del ruolo CodeBuild di servizio per clonare il repository

L'esecuzione iniziale della pipeline avrà esito negativo perché è necessario aggiornare il ruolo di CodeBuild servizio con le autorizzazioni per l'estrazione dal repository.

Aggiungi l'autorizzazione `codecommit:GitPull` IAM alla tua policy sui ruoli di servizio. Per istruzioni su come aggiornare la policy nella console IAM, consultaAggiungi le CodeBuild GitClone autorizzazioni per le azioni di origine CodeCommit.

Passaggio 4: Visualizza i comandi del repository nell'output della build

Per visualizzare l'output della build

1. Quando il ruolo di servizio viene aggiornato correttamente, scegli Riprova nella CodeBuild fase non riuscita.
2. Dopo che la pipeline è stata eseguita correttamente, nella fase di compilazione corretta, scegli Visualizza dettagli.

Nella pagina dei dettagli, scegli la scheda Registri. Visualizza l'output della CodeBuild build. I comandi restituiscono il valore della variabile inserita.

I comandi generano il contenuto del `README.md` file, elencano i file nella directory, clonano il repository, visualizzano il registro ed eseguono. `git describe --all`

Tutorial: creare una pipeline con azioni AWS CloudFormation StackSets di distribuzione

In questo tutorial, utilizzi la AWS CodePipeline console per creare una pipeline con azioni di distribuzione per la creazione di uno stack set e la creazione di istanze di stack. Quando la pipeline viene eseguita, il modello crea uno stack set e inoltre crea e aggiorna le istanze in cui viene distribuito lo stack set.

Esistono due modi per gestire le autorizzazioni per un set di stack: ruoli IAM autogestiti e ruoli IAM gestiti. AWS Questo tutorial fornisce esempi di autorizzazioni autogestite.

Per utilizzare nel modo più efficace gli Stacksets in CodePipeline, è necessario avere una chiara comprensione dei concetti alla base AWS CloudFormation StackSets e del loro funzionamento. Consulta StackSets i concetti nella Guida per l'AWS CloudFormation utente.

Argomenti

Prerequisiti

Per le operazioni di stack set, si utilizzano due account diversi: un account di amministrazione e un account di destinazione. I set di stack vengono creati nell'account amministratore. Crei singoli stack che appartengono a uno stack impostato nell'account di destinazione.

Per creare un ruolo di amministratore con il tuo account amministratore

- Segui le istruzioni riportate in Configurare le autorizzazioni di base per le operazioni relative allo stack set. Il tuo ruolo deve avere un nome. **`AWSCloudFormationStackSetAdministrationRole`**

Per creare un ruolo di servizio nell'account di destinazione

- Crea un ruolo di servizio nell'account di destinazione che consideri affidabile l'account amministratore. Segui le istruzioni in Configurare le autorizzazioni di base per le operazioni di stack set. Il tuo ruolo deve avere un nome. **`AWSCloudFormationStackSetExecutionRole`**

Fase 1: Caricare il AWS CloudFormation modello di esempio e il file dei parametri

Crea un bucket sorgente per il modello e i file dei parametri del set di stack. Scarica il file AWS CloudFormation modello di esempio, configura un file di parametri, quindi comprimi i file prima di caricarli nel bucket di origine S3.

> **Note**
>
> Assicurati di comprimere i file sorgente prima di caricarli nel bucket di origine S3, anche se l'unico file sorgente è il modello.

Per creare un bucket sorgente S3

1. Accedi AWS Management Console e apri la console Amazon S3 all'indirizzo https://console.aws.amazon.com/s3/.
2. Seleziona Crea bucket.
3. In Nome bucket, inserisci un nome del bucket.

   In Regione, scegli la regione in cui desideri creare la tua pipeline. Seleziona Crea bucket.
4. Dopo aver creato il bucket, viene visualizzato un banner di successo. Scegliere Go to bucket details (Vai ai dettagli del bucket).

5. Nella scheda Properties (Proprietà) scegliere Versioning (Funzione Versioni multiple). Scegliere Enable versioning (Abilita funzione Versioni multiple), quindi scegliere Save (Salva).

Per creare il file AWS CloudFormation modello

1. Scarica il seguente file modello di esempio per generare la CloudTrail configurazione per i set di stack:https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/EnableAWSCloudtrail.yml.
2. Salva il file con nome `template.yml`.

Per creare il file parameters.txt

1. Crea un file con i parametri per la tua distribuzione. I parametri sono valori che desideri aggiornare nello stack in fase di esecuzione. Il seguente file di esempio aggiorna i parametri del modello per il set di stack per abilitare la convalida della registrazione e gli eventi globali.

```
[
  {
    "ParameterKey": "EnableLogFileValidation",
    "ParameterValue": "true"
  },
  {
    "ParameterKey": "IncludeGlobalEvents",
    "ParameterValue": "true"
  }
]
```

2. Salva il file con nome `parameters.txt`.

Per creare il file accounts.txt

1. Create un file con gli account in cui desiderate creare le istanze, come illustrato nel seguente file di esempio.

```
[
    "111111222222","333333444444"
]
```

2. Salva il file con nome `accounts.txt`.

Per creare e caricare file sorgente

1. Combina i file in un unico file ZIP. I tuoi file dovrebbero avere questo aspetto nel tuo file ZIP.

```
template.yml
parameters.txt
accounts.txt
```

2. Carica il file ZIP nel tuo bucket S3. Questo file è l'elemento di origine creato dalla procedura guidata Create Pipeline per l'azione di distribuzione in. CodePipeline

Fase 2: creazione della pipeline

In questa sezione, andrai a creare una pipeline con le operazioni seguenti:

- Una fase di origine con un'azione sorgente S3 in cui l'artefatto di origine è il file modello e tutti i file sorgente di supporto.
- Una fase di distribuzione con un'azione di distribuzione AWS CloudFormation dello stack set che crea lo stack set.
- Una fase di distribuzione con un'azione di distribuzione di istanze in AWS CloudFormation stack che crea gli stack e le istanze all'interno degli account di destinazione.

Per creare una pipeline con un'azione CloudFormationStackSet

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Welcome (Benvenuto), pagina Getting started (Nozioni di base) o pagina Pipelines (Pipeline), scegli Create pipeline (Crea pipeline).
3. In Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere **`MyStackSetsPipeline`**.
4. Nel tipo di pipeline, scegli V1 ai fini di questo tutorial. Puoi anche scegliere V2; tuttavia, tieni presente che i tipi di tubazione differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. In Ruolo di servizio, scegli Nuovo ruolo di servizio per consentire la creazione CodePipeline di un ruolo di servizio in IAM.
6. Nel negozio Artifact, lascia le impostazioni predefinite.

> **Note**
>
> Non si tratta del bucket di origine per il codice sorgente, ma dell'archivio artefatti per la pipeline. È richiesto un archivio artefatti separato, ad esempio un bucket S3, per ogni pipeline. Quando crei o modifichi una pipeline, devi avere un bucket di artefatti nella regione della pipeline e un bucket di artefatti per regione in cui stai eseguendo un'azione. AWS
> Per ulteriori informazioni, consulta Artefatti di input e output e CodePipeline riferimento alla struttura della tubazione.

Seleziona Successivo.

7. Nella pagina Fase 2: aggiunta fase di origine in Source provider (Provider origine) scegliere Amazon S3.
8. In Bucket, inserisci il bucket di origine S3 che hai creato per questo tutorial, ad esempio. `BucketName` Nella chiave oggetto S3, inserisci il percorso e il nome del file per il tuo file ZIP, ad esempio. `MyFiles.zip`
9. Seleziona Successivo.
10. In Step 3: Add build stage (Fase 3: aggiunta della fase di compilazione), scegli Skip build stage (Ignora fase di compilazione) e quindi accetta il messaggio di avviso scegliendo Skip (Ignora).

    Seleziona Successivo.
11. In Step 4: Add deploy stage (Fase 4: aggiunta della fase di distribuzione):

    a. In Deploy provider, scegli AWS CloudFormation Stack Set.

    b. In Stack set name, inserisci un nome per lo stack set. Questo è il nome dello stack set creato dal modello.

    > **Note**
    >
    > Prendi nota del nome del set di stack. Lo utilizzerai quando aggiungerai la seconda azione di StackSets distribuzione alla tua pipeline.

    c. In Percorso del modello, inserisci il nome dell'artefatto e il percorso del file in cui hai caricato il file modello. Ad esempio, immettete quanto segue utilizzando il nome dell'artefatto di origine predefinito. `SourceArtifact`

```
SourceArtifact::template.yml
```

d. In Destinazioni di distribuzione, inserisci il nome dell'artefatto e il percorso del file in cui hai caricato il file degli account. Ad esempio, inserisci quanto segue utilizzando il nome dell'artefatto di origine predefinito. `SourceArtifact`

```
SourceArtifact::accounts.txt
```

e. In Deployment target Regioni AWS, inserisci una regione per la distribuzione dell'istanza dello stack iniziale, ad esempio. `us-east-1`

f. Espandi le opzioni di distribuzione. In Parametri, inserisci il nome dell'artefatto e il percorso del file in cui hai caricato il file dei parametri. Ad esempio, inserite quanto segue utilizzando il nome predefinito dell'artefatto di origine. `SourceArtifact`

```
SourceArtifact::parameters.txt
```

Per inserire i parametri come input letterale anziché come percorso di file, immettete quanto segue:

```
ParameterKey=EnableLogFileValidation,ParameterValue=true
ParameterKey=IncludeGlobalEvents,ParameterValue=true
```

g. In Capabilities, scegliete CAPABILITY_IAM e CAPABILITY_NAMED_IAM.

h. Nel modello di autorizzazione, scegliete SELF_MANAGED.

i. In Percentuale di tolleranza ai guasti, immettete. `20`

j. In Percentuale simultanea massima, immettere`25`.

k. Seleziona Successivo.

l. Scegliere Create pipeline (Crea pipeline). Viene visualizzata la pipeline.

m. Consenti l'esecuzione della pipeline.

Fase 3: Visualizza la distribuzione iniziale

Visualizza le risorse e lo stato della distribuzione iniziale. Dopo aver verificato che la distribuzione abbia creato correttamente il set di stack, puoi aggiungere la seconda azione alla fase di distribuzione.

Per visualizzare le risorse

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. In Pipelines (Pipeline), seleziona la pipeline e scegli View (Visualizza). Il diagramma mostra le fasi di distribuzione e di origine della pipeline.
3. Scegli l' AWS CloudFormation azione relativa all'CloudFormationStackSetazione nella tua pipeline. Il modello, le risorse e gli eventi per il tuo set di stack vengono visualizzati nella AWS CloudFormation console.
4. Nel pannello di navigazione a sinistra, scegli StackSets. Nell'elenco, scegli il nuovo set di stack.
5. Scegli la scheda Stack instances. Verifica che un'istanza stack per ogni account fornito sia stata creata nella regione us-east-1. Verifica che lo stato di ogni istanza dello stack sia. `CURRENT`

Fase 4: Aggiungere un' CloudFormationStackInstancesazione

Crea un'azione successiva nella tua pipeline che AWS CloudFormation StackSets consenta di creare le restanti istanze dello stack.

Per creare un'azione successiva nella tua pipeline

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.

   In Pipelines (Pipeline), seleziona la pipeline e scegli View (Visualizza). Il diagramma mostra le fasi di distribuzione e di origine della pipeline.
2. Scegli questa opzione per modificare la pipeline. La pipeline viene visualizzata in modalità Modifica.
3. Nella fase di distribuzione, scegliete Modifica.
4. Nell'azione AWS CloudFormation Stack Set deploy, scegli Aggiungi gruppo di azioni.
5. Nella pagina Modifica azione, aggiungi i dettagli dell'azione:

   a. In Nome azione, inserisci un nome per l'azione.

   b. Nel provider Action, scegli AWS CloudFormation Stack Instances.

   c. In Input artefacts, scegli. SourceArtifact

   d. In Stack set name, inserite il nome del set di stack. Questo è il nome dello stack set che hai fornito nella prima azione.

e. In Deployment targets, inserisci il nome dell'artefatto e il percorso del file in cui hai caricato il file degli account. Ad esempio, inserisci quanto segue utilizzando il nome dell'artefatto di origine predefinito. `SourceArtifact`

```
SourceArtifact::accounts.txt
```

f. In Deployment target Regioni AWS, inserisci le regioni per la distribuzione delle istanze dello stack rimanenti, ad esempio e come `us-east-2` segue: `eu-central-1`

```
us-east2, eu-central-1
```

g. In Percentuale di tolleranza agli errori, inserisci. `20`

h. In Percentuale simultanea massima, immettere`25`.

i. Selezionare Salva.

j. .Rilasciare manualmente una modifica. La pipeline aggiornata viene visualizzata con due azioni nella fase di distribuzione.

Fase 5: Visualizza le risorse dello stack set per la distribuzione

È possibile visualizzare le risorse e lo stato della distribuzione dello stack set.

Per visualizzare le risorse

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. In Pipelines, scegli la tua pipeline, quindi scegli Visualizza. Il diagramma mostra le fasi di distribuzione e di origine della pipeline.
3. Scegli l' AWS CloudFormation azione relativa all'**`AWS CloudFormation Stack Instances`**azione nella tua pipeline. Il modello, le risorse e gli eventi per il tuo set di stack vengono visualizzati nella AWS CloudFormation console.
4. Nel pannello di navigazione a sinistra, scegli StackSets. Nell'elenco, scegli il tuo set di stack.
5. Scegli la scheda Stack instances. Verifica che tutte le istanze stack rimanenti per ogni account che hai fornito siano state create o aggiornate nelle regioni previste. Verifica che lo stato di ogni istanza dello stack sia. `CURRENT`

Passaggio 6: Effettua un aggiornamento al tuo set di stack

Aggiorna il tuo set di stack e distribuisci l'aggiornamento alle istanze. In questo esempio, apporti anche una modifica agli obiettivi di distribuzione che desideri designare per l'aggiornamento. Le istanze che non fanno parte dell'aggiornamento passano a uno stato obsoleto.

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. In Pipeline, scegli la tua pipeline, quindi scegli Modifica. Nella fase di distribuzione, scegli Modifica.
3. Scegli di modificare l'azione AWS CloudFormation Stack Set nella tua pipeline. In Descrizione, sovrascrivi la descrizione esistente con una nuova descrizione per lo stack set.
4. Scegli di modificare l'azione AWS CloudFormation Stack Instances nella tua pipeline. In Deployment target Regioni AWS, elimina il `us-east-2` valore che è stato inserito al momento della creazione dell'azione.
5. Salvare le modifiche. Scegli Release change per eseguire la pipeline.
6. Apri la tua azione in AWS CloudFormation. Scegli la scheda StackSet Informazioni. Nella StackSet descrizione, verifica che sia mostrata la nuova descrizione.
7. Scegli la scheda Stack instances. In Status, verifica che lo stato delle istanze dello stack in us-east-2 sia. `OUTDATED`

CodePipeline migliori pratiche e casi d'uso

Le sezioni seguenti descrivono le migliori pratiche per CodePipeline.

Argomenti

Casi d'uso per CodePipeline

Puoi creare pipeline che si integrano con altre Servizi AWS. Questi possono essere Servizi AWS, come Amazon S3, o prodotti di terze parti, come. GitHub Questa sezione fornisce esempi CodePipeline da utilizzare per automatizzare i rilasci di codice utilizzando diverse integrazioni di prodotti. Per un elenco completo delle integrazioni CodePipeline organizzate per tipo di azione, consulta. CodePipeline riferimento alla struttura della tubazione

Argomenti

CodePipeline Utilizzabile con Amazon S3 e AWS CodeCommitAWS CodeDeploy

Quando crei una pipeline, CodePipeline si integra con AWS prodotti e servizi che agiscono come fornitori di azioni in ogni fase della pipeline. Quando si scelgono le fasi nella procedura guidata,

occorre selezionare una fase di origine e almeno una fase di compilazione o di distribuzione. La procedura guidata crea automaticamente le fasi con nomi predefiniti che non possono essere modificati. Questi sono i nomi delle fasi create quando si imposta una pipeline suddivisa in tre fasi completa nella procedura guidata:

- Una fase operazione di origine con un nome predefinito "Origine".
- Una fase operazione di compilazione con un nome predefinito "Compilazione".
- Una fase operazione di distribuzione con un nome predefinito "Gestione temporanea".

Puoi utilizzare i tutorial in questa guida per creare pipeline e specificare fasi:

- I passaggi descritti ti Tutorial: creazione di una semplice pipeline (bucket S3) aiutano a utilizzare la procedura guidata per creare una pipeline con due fasi predefinite: «Source» e «Staging», in cui il tuo repository Amazon S3 è il provider di origine. Questo tutorial crea una pipeline da utilizzare AWS CodeDeploy per distribuire un'applicazione di esempio da un bucket Amazon S3 a istanze Amazon EC2 che eseguono Amazon Linux.
- I passaggi descritti ti Tutorial: crea una pipeline semplice (CodeCommitrepository) aiutano a utilizzare la procedura guidata per creare una pipeline con una fase «Source» che utilizza il tuo repository come provider di origine. AWS CodeCommit Questo tutorial crea una pipeline che utilizza AWS CodeDeploy per distribuire un'applicazione di esempio da un AWS CodeCommit repository a un'istanza Amazon EC2 che esegue Amazon Linux.

Utilizzalo CodePipeline con provider di azioni di terze parti (e Jenkins) GitHub

Puoi creare pipeline che si integrano con prodotti di terze parti come Jenkins GitHub e Jenkins. Le fasi in Tutorial: creazione di una pipeline a quattro fasi mostrano come creare una pipeline che consente di:

- Ottiene il codice sorgente da un repository, GitHub
- Utilizzare Jenkins per compilare e testare il codice sorgente
- Viene utilizzato AWS CodeDeploy per distribuire il codice sorgente creato e testato su istanze Amazon EC2 che eseguono Amazon Linux o Microsoft Windows Server.

Usalo CodePipeline con AWS CodeStar per creare una pipeline in un progetto di codice

AWS CodeStar è un servizio basato su cloud che fornisce un'interfaccia utente unificata per la gestione di progetti di sviluppo software su. AWS AWS CodeStar collabora con CodePipeline per combinare AWS le risorse in una toolchain di sviluppo del progetto. Puoi utilizzare la AWS CodeStar dashboard per creare automaticamente la pipeline, i repository, il codice sorgente, i file delle specifiche di compilazione, il metodo di distribuzione e le istanze di hosting o le istanze serverless necessarie per un progetto di codice completo.

Per creare il tuo AWS CodeStar progetto, scegli il linguaggio di codifica e il tipo di applicazione che desideri distribuire. Puoi creare i seguenti tipi di progetto: un'applicazione Web, un servizio Web o uno skill Alexa.

In qualsiasi momento, puoi integrare il tuo IDE preferito nella tua AWS CodeStar dashboard. Inoltre, puoi aggiungere e rimuovere membri del team e gestire le autorizzazioni per i membri del team nel progetto. Per un tutorial che mostra come creare una pipeline di esempio per un'applicazione serverless, vedi Tutorial: Creazione e gestione di un progetto serverless in. AWS CodeStar AWS CodeStar

CodePipeline Da utilizzare per compilare, creare e testare il codice con CodeBuild

CodeBuild è un servizio di compilazione gestito nel cloud che consente di creare e testare il codice senza un server o un sistema. Usalo CodePipeline with CodeBuild per automatizzare le revisioni in esecuzione attraverso la pipeline per la distribuzione continua di build software ogni volta che viene apportata una modifica al codice sorgente. Per ulteriori informazioni, consulta Use CodePipeline with CodeBuild per testare il codice ed eseguire build.

CodePipeline Utilizzalo con Amazon ECS per la distribuzione continua di applicazioni basate su container nel cloud

Amazon ECS è un servizio di gestione dei container che consente di distribuire applicazioni basate su contenitori su istanze Amazon ECS nel cloud. Utilizzalo CodePipeline con Amazon ECS per automatizzare l'esecuzione delle revisioni attraverso la pipeline per la distribuzione continua di applicazioni basate su container ogni volta che viene apportata una modifica all'archivio di immagini di origine. Per ulteriori informazioni, consulta Tutorial: Continuous Deployment with. CodePipeline

Utilizzalo CodePipeline con Elastic Beanstalk per la distribuzione continua di applicazioni Web sul cloud

Elastic Beanstalk è un servizio di elaborazione che consente di distribuire applicazioni e servizi Web su server Web. Utilizzalo CodePipeline con Elastic Beanstalk per la distribuzione continua di applicazioni Web nel tuo ambiente applicativo. Puoi anche usarlo AWS CodeStar per creare una pipeline con un'azione di distribuzione di Elastic Beanstalk.

Usalo CodePipeline con AWS Lambda per la distribuzione continua di applicazioni basate su Lambda e serverless

È possibile utilizzare AWS Lambda with CodePipeline per richiamare una AWS Lambda funzione, come descritto in Distribuzione di applicazioni serverless. È inoltre possibile utilizzare AWS Lambda e creare una pipeline AWS CodeStar per la distribuzione di applicazioni serverless.

Utilizzalo CodePipeline con AWS CloudFormation modelli per la distribuzione continua al cloud

È possibile utilizzare AWS CloudFormation con CodePipeline per la distribuzione e l'automazione continue. Per ulteriori informazioni, consulta Continuous Delivery with CodePipeline. AWS CloudFormation viene utilizzato anche per creare i modelli per le pipeline create in AWS CodeStar.

Assegnazione di tag alle risorse

Un tag è un'etichetta di attributo personalizzata che l'utente o AWS assegna a una AWS risorsa. Ogni AWS tag è composto da due parti:

- Una chiave di tag (ad esempio, `CostCenter`, `Environment`, `Project` o `Secret`). Le chiavi dei tag prevedono una distinzione tra lettere maiuscole e minuscole.
- Un campo facoltativo noto come valore del tag (ad esempio, `111122223333`, `Production` o un nome di team). Non specificare il valore del tag equivale a utilizzare una stringa vuota. Analogamente alle chiavi dei tag, i valori dei tag prevedono una distinzione tra lettere maiuscole e minuscole.

Tutti questi sono noti come coppie chiave-valore.

I tag ti aiutano a identificare e organizzare AWS le tue risorse. Molti Servizi AWS supportano l'etichettatura, quindi puoi assegnare lo stesso tag a risorse di servizi diversi per indicare che le risorse sono correlate. Ad esempio, puoi assegnare a una pipeline lo stesso tag che assegni a un bucket di origine Amazon S3.

Per suggerimenti sull'utilizzo dei tag, consulta il post AWS Tagging Strategies nel blog AWS Answers.

Puoi taggare i seguenti tipi di risorse in: CodePipeline

- Contrassegna una pipeline in CodePipeline
- Etichetta un'azione personalizzata in CodePipeline

Puoi utilizzare le AWS CLI CodePipeline API o gli AWS SDK per:

- Aggiungere tag a una pipeline, a un'operazione personalizzata o a un webhook al momento della creazione.
- Aggiungere, gestire e rimuovere i tag per una pipeline, un'operazione azione personalizzata o un webhook.

È inoltre possibile utilizzare la console per aggiungere, gestire e rimuovere i tag per una pipeline.

Oltre a identificare, organizzare e tracciare la tua risorsa con i tag, puoi utilizzare i tag nelle policy IAM per controllare chi può visualizzare e interagire con la tua risorsa. Per esempi di policy di accesso basate su tag, consulta Utilizzo dei tag per controllare l'accesso alle CodePipeline risorse.

Utilizzo CodePipeline con Amazon Virtual Private Cloud

AWS CodePipeline ora supporta gli endpoint Amazon Virtual Private Cloud (Amazon VPC) con tecnologia. AWS PrivateLink Ciò significa che puoi connetterti direttamente CodePipeline tramite un endpoint privato nel tuo VPC, mantenendo tutto il traffico all'interno del tuo VPC e della rete. AWS

Amazon VPC è un software Servizio AWS che puoi utilizzare per avviare AWS risorse in una rete virtuale definita dall'utente. Con un VPC, hai il controllo delle impostazioni di rete, ad esempio:

- Intervallo di indirizzo IP
- Sottoreti
- Tabelle di instradamento
- Gateway di rete

Gli endpoint VPC di interfaccia sono alimentati da AWS PrivateLink, una AWS tecnologia che facilita la comunicazione privata tra l' Servizi AWS utilizzo di un'interfaccia di rete elastica con indirizzi IP privati. Per connettere il tuo VPC CodePipeline, definisci un'interfaccia VPC endpoint per. CodePipeline Questo tipo di endpoint ti consente di connettere il tuo Servizi AWS VPC a. L'endpoint fornisce una connettività affidabile e scalabile CodePipeline senza richiedere un gateway Internet, un'istanza NAT (Network Address Translation) o una connessione VPN. Per ulteriori informazioni sulla configurazione di un VPC, consulta la Guida per l'utente di VPC.

Disponibilità

CodePipeline attualmente supporta gli endpoint VPC nei seguenti casi: Regioni AWS

- Stati Uniti orientali (Ohio)
- Stati Uniti orientali (Virginia settentrionale)
- Stati Uniti occidentali (California settentrionale)
- US West (Oregon)
- Canada (Centrale)
- Europa (Francoforte)
- Europa (Irlanda)
- Europa (Londra)
- Europa (Milano) *

- Europa (Parigi)
- Europa (Stoccolma)
- Asia Pacifico (Hong Kong) *
- Asia Pacifico (Mumbai)
- Asia Pacifico (Tokyo)
- Asia Pacific (Seul)
- Asia Pacifico (Singapore)
- Asia Pacifico (Sydney)
- Sud America (San Paolo)
- AWS GovCloud (Stati Uniti occidentali)

* È necessario abilitare questa regione prima di poterla utilizzare.

Creazione di un endpoint VPC per CodePipeline

Puoi utilizzare la console Amazon VPC per creare com.amazonaws. endpoint VPC ***region*** .codepipeline. Nella console, *region è l'identificatore di regione* per una regione Regione AWS supportata da CodePipeline, ad esempio per la regione Stati Uniti orientali (`us-east-2`Ohio). Per ulteriori informazioni, consulta Creazione di un endpoint di interfaccia nella Guida per l'utente di Amazon VPC.

L'endpoint è precompilato con la regione specificata al momento dell'accesso ad AWS. Se accedi a un'altra regione, allora l'endpoint VPC si aggiorna con la nuova regione.

> **Note**
>
> Altri Servizi AWS che forniscono supporto VPC e si integrano con CodePipeline, ad esempio CodeCommit, potrebbero non supportare l'utilizzo degli endpoint Amazon VPC per tale integrazione. Ad esempio, il traffico tra CodePipeline e CodeCommit non può essere limitato all'intervallo della sottorete VPC.

Risoluzione dei problemi relativi alla configurazione del VPC

Durante la risoluzione dei problemi relativi al VPC, utilizza le informazioni che compaiono nei messaggi di errore sulla connettività Internet per identificare, diagnosticare e risolvere tali problemi.

1. Assicurati che il gateway Internet sia collegato al tuo VPC.
2. Assicurarsi che la tabella di routing della sottorete pubblica punti all'Internet Gateway.
3. Assicurarsi che le liste di controllo degli accessi di rete consentano il traffico sul flusso.
4. Assicurarsi che i gruppi di sicurezza consentano il traffico sul flusso.
5. Assicurati che la tabella di routing per le sottoreti private punti al gateway privato virtuale.
6. Assicurati che il ruolo di servizio utilizzato da CodePipeline disponga delle autorizzazioni appropriate. Ad esempio, se CodePipeline non disponi delle autorizzazioni Amazon EC2 necessarie per lavorare con un Amazon VPC, potresti ricevere un errore che dice «Errore EC2 inaspettato:». UnauthorizedOperation

Lavorare con le condutture in CodePipeline

Per definire un processo di rilascio automatico in AWS CodePipeline, crei una pipeline, che è un costrutto di flusso di lavoro che descrive come le modifiche al software passano attraverso un processo di rilascio. Una pipeline è costituita da fasi e operazioni configurabili.

> **Note**
>
> Quando aggiungete le fasi Build, Deploy, Test o Invoke, oltre alle opzioni predefinite fornite CodePipeline, potete scegliere azioni personalizzate che avete già creato da utilizzare con le vostre pipeline. Le operazioni personalizzate si possono usare per attività quali l'esecuzione di un processo di creazione sviluppato internamente o l'esecuzione di una serie di test. Sono inclusi degli identificatori di versione per distinguere le diverse versioni di un'operazione personalizzata negli elenchi del provider. Per ulteriori informazioni, consulta Creare e aggiungere un'azione personalizzata in CodePipeline.

Prima di poter creare una pipeline, è necessario innanzitutto completare i passaggi descritti in Guida introduttiva con CodePipeline.

Per ulteriori informazioni sulle pipeline, consulta e CodePipeline concetti CodePipeline tutorial, se desideri utilizzare la per AWS CLI creare una pipeline,. CodePipeline riferimento alla struttura della tubazione Per visualizzare un elenco di pipeline, consulta Visualizza le pipeline e i dettagli in CodePipeline.

Argomenti

- Creare il ruolo CodePipeline di servizio
- Contrassegna una pipeline in CodePipeline
- Creazione di una regola di notifica

Avvia una pipeline in CodePipeline

Ogni esecuzione della pipeline può essere avviata in base a un trigger diverso. Ogni esecuzione della pipeline può avere un tipo di trigger diverso, a seconda di come viene avviata la pipeline. Il tipo di trigger per ogni esecuzione viene mostrato nella cronologia delle esecuzioni di una pipeline. I tipi di trigger possono dipendere dal provider di azioni di origine nel modo seguente:

> **Note**
>
> Non è possibile specificare più di un trigger per azione sorgente.

- Creazione di una pipeline: quando viene creata una pipeline, l'esecuzione della pipeline viene avviata automaticamente. Questo è il tipo di **`CreatePipeline`** trigger nella cronologia di esecuzione.
- Modifiche sugli oggetti revisionati: questa categoria rappresenta il tipo di `PutActionRevision` trigger nella cronologia di esecuzione.
- Rilevamento delle modifiche su branch e commit per un push di codice: questa categoria rappresenta il tipo di `CloudWatchEvent` trigger nella cronologia di esecuzione. Quando viene rilevata una modifica a un commit e a un branch di origine nel repository di origine, viene avviata la pipeline. Questo tipo di trigger utilizza il rilevamento automatico delle modifiche. I provider di source action che utilizzano questo tipo di trigger sono S3 e CodeCommit. Questo tipo viene utilizzato anche per una pianificazione che avvia la pipeline. Per informazioni, consulta Avvia una pipeline in base a una pianificazione.
- Sondaggio delle modifiche all'origine: questa categoria rappresenta il tipo di `PollForSourceChanges` trigger nella cronologia di esecuzione. Quando viene rilevata una modifica a un commit e a un branch di origine nel repository di origine tramite polling, la pipeline si avvia. Questo tipo di trigger non è consigliato e deve essere migrato per utilizzare il rilevamento automatico delle modifiche. I provider di source action che utilizzano questo tipo di trigger sono S3 e. CodeCommit

- Eventi Webhook per fonti di terze parti: Questa categoria rappresenta il tipo di `Webhook` trigger nella cronologia di esecuzione. Quando viene rilevata una modifica da un evento webhook, viene avviata la pipeline. Questo tipo di trigger utilizza il rilevamento automatico delle modifiche. I source action provider che utilizzano questo tipo di trigger sono connessioni configurate per code push (Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab autogestite).
- Eventi WebHookV2 per fonti di terze parti: questa categoria rappresenta il tipo di trigger nella **`WebhookV2`** cronologia di esecuzione. Questo tipo è destinato alle esecuzioni che vengono attivate in base ai trigger definiti nella definizione della pipeline. Quando viene rilevata una versione con un tag Git specificato, viene avviata la pipeline. Puoi utilizzare i tag Git per contrassegnare un commit con un nome o un altro identificatore che permetta agli altri utenti del repository di comprenderne l'importanza. Puoi inoltre utilizzare i tag Git per identificare un determinato commit nella cronologia di un repository. Questo tipo di trigger disabilita il rilevamento automatico delle modifiche. I provider di azioni di origine che utilizzano questo tipo di trigger sono connessioni configurate per i tag Git (Bitbucket Cloud GitHub, GitHub Enterprise Server e GitLab .com).
- Avvio manuale di una pipeline: questa categoria rappresenta il tipo di **`StartPipelineExecution`** trigger nella cronologia di esecuzione. È possibile utilizzare la console o AWS CLI avviare una pipeline manualmente. Per informazioni, consulta Avvio manuale di una pipeline.
- RollbackStage: Questa categoria rappresenta il tipo di `RollbackStage` trigger nella cronologia di esecuzione. È possibile utilizzare la console o AWS CLI ripristinare uno stage manualmente o automaticamente. Per informazioni, consulta Configurazione del rollback dello stage.

Quando aggiungi un'azione di origine alla tua pipeline che utilizza tipi di trigger per il rilevamento automatico delle modifiche, tali azioni utilizzano risorse aggiuntive. La creazione di ciascuna azione di origine è dettagliata in sezioni separate grazie a queste risorse aggiuntive per il rilevamento delle modifiche. Per informazioni dettagliate su ciascun fornitore di origine e sui metodi di rilevamento delle modifiche necessari per il rilevamento automatico delle modifiche, vedere Azioni all'origine e metodi di rilevamento delle modifiche.

Argomenti

- Azioni all'origine e metodi di rilevamento delle modifiche
- Avvio manuale di una pipeline
- Avvia una pipeline in base a una pianificazione
- Avvia una pipeline con una modifica della revisione del codice sorgente

Azioni all'origine e metodi di rilevamento delle modifiche

Quando aggiungi un'azione di origine alla tua pipeline, le azioni funzionano con risorse aggiuntive descritte nella tabella.

> **Note**
>
> Le azioni di origine CodeCommit e S3 richiedono una risorsa configurata per il rilevamento delle modifiche (una EventBridge regola) o utilizzano l'opzione per eseguire il polling del repository per le modifiche all'origine. Per le pipeline con un'azione di origine Bitbucket o GitHub Enterprise Server GitHub, non è necessario configurare un webhook o impostare il polling come impostazione predefinita. L'azione connessioni gestisce automaticamente il rilevamento delle modifiche.

Origine	Utilizza risorse aggiuntive?	Fasi
Amazon S3	Questa azione sorgente utilizza risorse aggiuntive. Quando utilizzi la CLI o CloudFormation per creare questa azione, crei e gestisci anche queste risorse.	Vedi Creare una pipeline in CodePipeline e Azioni di origine di Amazon S3 e con EventBridge AWS CloudTrail
Bitbucket Cloud	Questa azione di origine utilizza una risorsa di connessione.	Per informazioni, consultare Connessioni Bitbucket Cloud.
AWS CodeCommit	Amazon EventBridge (consigliato). Questa è l'impostazione predefinita per le pipeline con un' CodeCommi t origine creata o modificata nella console.	Vedi Creare una pipeline in CodePipeline e CodeCommit azioni di origine e EventBridge
Amazon ECR	Amazon EventBridge. Questo viene creato dalla procedura guidata per le pipeline con una fonte Amazon ECR creata o modificata nella console.	Consulta Creare una pipeline in CodePipeline e Azioni e risorse relative ai sorgenti di Amazon ECR EventBridge .

Origine	Utilizza risorse aggiuntive?	Fasi
GitHub o Enterpris e Cloud GitHub	Questa azione sorgente utilizza una risorsa di connessione.	Per informazioni, consultare GitHub connessioni.
GitHub Server aziendale	Questa azione di origine utilizza una risorsa di connessione e una risorsa host.	Per informazioni, consultare GitHub Connessioni Enterprise Server.
GitLab.com	Questa azione di origine utilizza una risorsa di connessione.	Per informazioni, consultare GitLabconnessioni.com.
GitLab autogestito	Questa azione di origine utilizza una risorsa di connessione e una risorsa host.	Per informazioni, consultare Connessioni per la GitLab gestione automatica.

Se si dispone di una pipeline che utilizza il polling, è possibile aggiornarla per utilizzare il metodo di rilevamento consigliato. Per ulteriori informazioni, consulta Aggiornamento delle pipeline di polling nel metodo di rilevamento delle modifiche consigliato.

Se desideri disattivare il rilevamento delle modifiche per un'azione di origine che utilizza connessioni, consultaCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Avvio manuale di una pipeline

Per impostazione predefinita, una pipeline viene avviata automaticamente al momento della creazione e ogni volta che viene apportata una modifica in un repository di origine. Tuttavia, è possibile eseguire nuovamente la revisione più recente nella pipeline una seconda volta. Puoi utilizzare la CodePipeline console o il start-pipeline-execution comando AWS CLI and per rieseguire manualmente la revisione più recente attraverso la pipeline.

Argomenti

- Avvio manuale di una pipeline (console)
- Avvio manuale di una pipeline (CLI)

Avvio manuale di una pipeline (console)

Per avviare manualmente una pipeline ed eseguire la revisione più recente in una pipeline

1. Accedi AWS Management Console e apri la console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home. CodePipeline
2. In Name (Nome), scegliere il nome della pipeline da avviare.
3. Nella pagina dei dettagli della pipeline, scegli Release change. Se la pipeline è configurata per passare parametri (variabili della pipeline), scegliendo Release change si apre la finestra Release change. Nelle variabili Pipeline, nel campo o nei campi per le variabili a livello di pipeline, inserisci il valore o i valori che desideri passare per l'esecuzione di questa pipeline. Per ulteriori informazioni, consulta Variables.

   In questo modo viene avviata la revisione più recente disponibile in ogni percorso di origine specificato in un'operazione di origine nella pipeline.

Avvio manuale di una pipeline (CLI)

Per avviare manualmente una pipeline ed eseguire la versione più recente di un artefatto in una pipeline

1. Apri un terminale (Linux, macOS o Unix) o il prompt dei comandi (Windows) e usa AWS CLI per eseguire il start-pipeline-execution comando, specificando il nome della pipeline che desideri avviare. Ad esempio, per iniziare a eseguire l'ultima modifica tramite una pipeline denominata: *MyFirstPipeline*

   ```
   aws codepipeline start-pipeline-execution --name MyFirstPipeline
   ```

   Per avviare una pipeline in cui le variabili sono configurate a livello di pipeline, utilizzate il start-pipeline-execution comando con l'--variablesargomento opzionale per avviare la pipeline e aggiungere le variabili che verranno utilizzate nell'esecuzione. Ad esempio, per aggiungere una variabile `var1` con un valore di1, utilizzate il comando seguente:

   ```
   aws codepipeline start-pipeline-execution --name MyFirstPipeline --variables
    name=var1,value=1
   ```
2. Per verificare l'esito positivo, visualizzare l'oggetto restituito. Questo comando restituisce un ID esecuzione, simile al seguente:

```
{
    "pipelineExecutionId": "c53dbd42-This-Is-An-Example"
}
```

> **Note**
>
> Dopo aver avviato la pipeline, è possibile monitorarne l'avanzamento nella CodePipeline console o eseguendo il get-pipeline-state comando. Per ulteriori informazioni, consulta Visualizza le pipeline (console) e Visualizzazione dei dettagli e della cronologia della pipeline (CLI).

Avvia una pipeline in base a una pianificazione

È possibile impostare una regola EventBridge per avviare una pipeline in base a una pianificazione.

Crea una EventBridge regola che pianifichi l'avvio della pipeline (console)

Per creare una EventBridge regola con una pianificazione come origine dell'evento

1. Apri la EventBridge console Amazon all'indirizzo https://console.aws.amazon.com/events/.
2. Nel pannello di navigazione, scegli Regole.
3. Scegli Crea regola, quindi in Dettagli della regola, scegli Pianifica.
4. Configurare la pianificazione utilizzando una frequenza o espressione fissa. Per informazioni, consulta la sezione relativa alla pianificazione dell'espressione per regole.
5. In Obiettivi, scegli CodePipeline.
6. Immettere l'ARN della pipeline per l'esecuzione della pipeline per questa pianificazione.

> **Note**
>
> Puoi trovare l'ARN della pipeline in Impostazioni nella console. Per informazioni, consulta Visualizza l'ARN della pipeline e l'ARN del ruolo di servizio (console).

7. Scegli una delle seguenti opzioni per creare o specificare un ruolo del servizio IAM che dia EventBridge le autorizzazioni per richiamare la destinazione associata alla tua EventBridge regola (in questo caso, la destinazione è). CodePipeline

- Scegli Crea un nuovo ruolo per questa risorsa specifica per creare un ruolo di servizio che conceda le EventBridge autorizzazioni per avviare le esecuzioni della pipeline.
- Scegli Usa il ruolo esistente per inserire un ruolo di servizio che concede le EventBridge autorizzazioni per avviare le esecuzioni della pipeline.

8. Scegli Configura dettagli.
9. Nella pagina Configure rule details (Configura dettagli della regola), immetti un nome e una descrizione per la regola e quindi scegli State (Stato) per abilitare la regola.
10. Se la regola ti soddisfa, scegli Create rule (Crea regola).

Crea una EventBridge regola che pianifichi l'avvio della pipeline (CLI)

Per utilizzare la AWS CLI per creare una regola, chiamate il put-rule comando, specificando:

- Un nome che identifica in modo univoco la regola che stai creando. Questo nome deve essere univoco in tutte le pipeline create CodePipeline associate al tuo AWS account.
- L'espressione di pianificazione per la regola.

Per creare una EventBridge regola con una pianificazione come origine dell'evento

1. Chiama il comando put-rule e includi i parametri `--name` e `--schedule-expression`.

   Esempi:

   Il seguente comando di esempio utilizza la creazione --schedule-expression di una regola denominata `MyRule2` che filtra EventBridge in base a una pianificazione.

   ```
   aws events put-rule --schedule-expression 'cron(15 10 ? * 6L 2002-2005)' --name
    MyRule2
   ```

2. Concedi le autorizzazioni EventBridge da utilizzare per CodePipeline richiamare la regola. Per ulteriori informazioni, consulta Utilizzo delle politiche basate sulle risorse per Amazon. EventBridge

   a. Usa l'esempio seguente per creare la politica di fiducia che consente di EventBridge assumere il ruolo di servizio. Denominalo `trustpolicyforEB.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "events.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

b. Utilizza il seguente comando per creare il ruolo `Role-for-MyRule` e collegare la policy di attendibilità.

```
aws iam create-role --role-name Role-for-MyRule --assume-role-policy-document
 file://trustpolicyforEB.json
```

c. Crea il JSON della policy delle autorizzazioni come mostrato in questo esempio per la pipeline denominata `MyFirstPipeline`. Denomina la policy delle autorizzazioni `permissionspolicyforEB.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:StartPipelineExecution"
            ],
            "Resource": [
                "arn:aws:codepipeline:us-west-2:80398EXAMPLE:MyFirstPipeline"
            ]
        }
    ]
}
```

d. Utilizza il comando seguente per collegare la nuova policy delle autorizzazioni `CodePipeline-Permissions-Policy-for-EB` al ruolo `Role-for-MyRule` che hai creato.

```
aws iam put-role-policy --role-name Role-for-MyRule --policy-name CodePipeline-
Permissions-Policy-For-EB --policy-document file://permissionspolicyforCWE.json
```

Avvia una pipeline con una modifica della revisione del codice sorgente

È possibile utilizzare le sostituzioni per avviare una pipeline con un ID di revisione dell'origine specifico fornito per l'esecuzione della pipeline. Ad esempio, se desideri avviare una pipeline che elabori un ID di commit specifico dalla tua CodeCommit origine, puoi aggiungere l'ID di commit come override all'avvio della pipeline.

Esistono quattro tipi di revisione del codice sorgente per: `revisionType`

- `COMMIT_ID`
- `IMAGE_DIGEST`
- `S3_OBJECT_VERSION_ID`
- `S3_OBJECT_OBJECT_KEY`

> **Note**
>
> Per i `IMAGE_DIGEST` tipi `COMMIT_ID` e i tipi di revisioni del codice sorgente, l'ID di revisione del codice sorgente si applica a tutto il contenuto del repository, in tutte le filiali.

> **Note**
>
> Per i `S3_OBJECT_KEY` tipi `S3_OBJECT_VERSION_ID` e le revisioni dei sorgenti, entrambi i tipi possono essere utilizzati indipendentemente oppure possono essere usati insieme per sovrascrivere l'origine con un VersionID specifico. ObjectKey

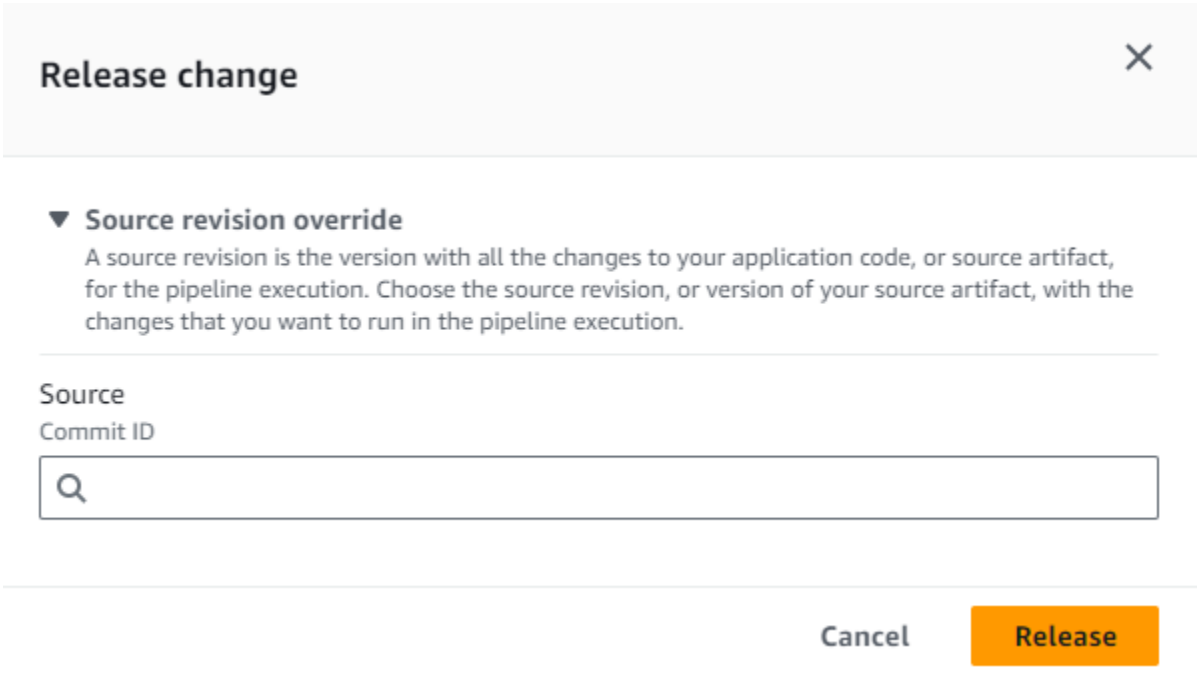
Argomenti

- Avvia una pipeline con una modifica della revisione del codice sorgente (console)
- Avvia una pipeline con un override della revisione del codice sorgente (CLI)

Avvia una pipeline con una modifica della revisione del codice sorgente (console)

Per avviare manualmente una pipeline ed eseguire la revisione più recente in una pipeline

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. In Name (Nome), scegliere il nome della pipeline da avviare.
3. Nella pagina dei dettagli della pipeline, scegli Release change. Scegliendo Release change si apre la finestra Release change. Per Sostituire la revisione della versione del codice sorgente, scegliete la freccia per espandere il campo. In Source, inserisci l'ID di revisione del codice sorgente. Ad esempio, se la tua pipeline ha un' CodeCommit origine, scegli l'ID di commit dal campo che desideri utilizzare.



Avvia una pipeline con un override della revisione del codice sorgente (CLI)

Per avviare manualmente una pipeline ed eseguire l'ID di revisione di origine specificato per un artefatto attraverso una pipeline

1. Apri un terminale (Linux, macOS o Unix) o il prompt dei comandi (Windows) e usa AWS CLI per eseguire il start-pipeline-execution comando, specificando il nome della pipeline che desideri avviare. Utilizzate l'--source-revisionsargomento anche per fornire l'ID di revisione del codice sorgente. La revisione di origine è composta da ActionName, RevisionType e RevisionValue. I

valori RevisionType validi sono. `COMMIT_ID | IMAGE_DIGEST | S3_OBJECT_VERSION_ID | S3_OBJECT_KEY`

Nell'esempio seguente, per iniziare a eseguire la modifica specificata tramite una pipeline denominatacodecommit-pipeline, il comando seguente specifica il nome dell'azione di origine di Source, un tipo di revisione e un ID di `COMMIT_ID` commit di. `78a25c18755ccac3f2a9eec099dEXAMPLE`

```
aws codepipeline start-pipeline-execution --name codecommit-pipeline --source-
revisions
 actionName=Source,revisionType=COMMIT_ID,revisionValue=78a25c18755ccac3f2a9eec099dEXAMPLE
 --region us-west-1
```

2. Per verificare l'esito positivo, visualizzare l'oggetto restituito. Questo comando restituisce un ID esecuzione, simile al seguente:

```
{
    "pipelineExecutionId": "c53dbd42-This-Is-An-Example"
}
```

> **Note**
>
> Dopo aver avviato la pipeline, potete monitorarne l'avanzamento nella CodePipeline console o eseguendo il comando. get-pipeline-state Per ulteriori informazioni, consultare Visualizza le pipeline (console) e Visualizzazione dei dettagli e della cronologia della pipeline (CLI).

Interrompere l'esecuzione di una pipeline in CodePipeline

Quando l'esecuzione di una pipeline inizia a essere eseguita attraverso una pipeline, entra in uno stadio alla volta e blocca lo stage mentre tutte le esecuzioni di azioni nello stage sono in esecuzione. Queste azioni in corso devono essere gestite in modo che, quando l'esecuzione della pipeline viene interrotta, le azioni possano essere completate o abbandonate.

Esistono due modi per interrompere l'esecuzione di una pipeline:

- Stop and wait: AWS CodePipeline attende di interrompere l'esecuzione fino al completamento di tutte le azioni in corso (ovvero, le azioni hanno uno stato `Succeeded` or`Failed`). Questa opzione

conserva le azioni in corso. L'esecuzione è in uno stato `Stopping` fino al completamento delle azioni in corso. Quindi l'esecuzione è in uno stato `Stopped`. Lo stage si sblocca al termine delle azioni.

Se scegli di fermarti e aspettare e cambi idea mentre la tua esecuzione è ancora in uno stato `Stopping`, puoi scegliere di abbandonare.

- Arresta e abbandona: AWS CodePipeline interrompe l'esecuzione senza attendere il completamento delle azioni in corso. L'esecuzione è in uno stato `Stopping` per un tempo molto breve mentre le azioni in corso sono abbandonate. Dopo l'interruzione dell'esecuzione, l'esecuzione dell'azione è in uno stato `Abandoned` mentre l'esecuzione della pipeline è in uno stato `Stopped`. Lo stage si sblocca.

  Per l'esecuzione di una pipeline in uno stato `Stopped`, è possibile ripetere le azioni nella fase in cui l'esecuzione è interrotta.

  > **Warning**
  >
  > Questa opzione può portare ad attività non riuscite o fuori sequenza.

Argomenti

- Arresto dell'esecuzione di una pipeline (console)
- Interrompere un'esecuzione in entrata (console)
- Arresto dell'esecuzione di una pipeline (CLI)
- Interruzione di un'esecuzione in entrata (CLI)

Arresto dell'esecuzione di una pipeline (console)

È possibile utilizzare la console per interrompere l'esecuzione di una pipeline. Scegliete un'esecuzione, quindi scegliete il metodo per fermare l'esecuzione della pipeline.

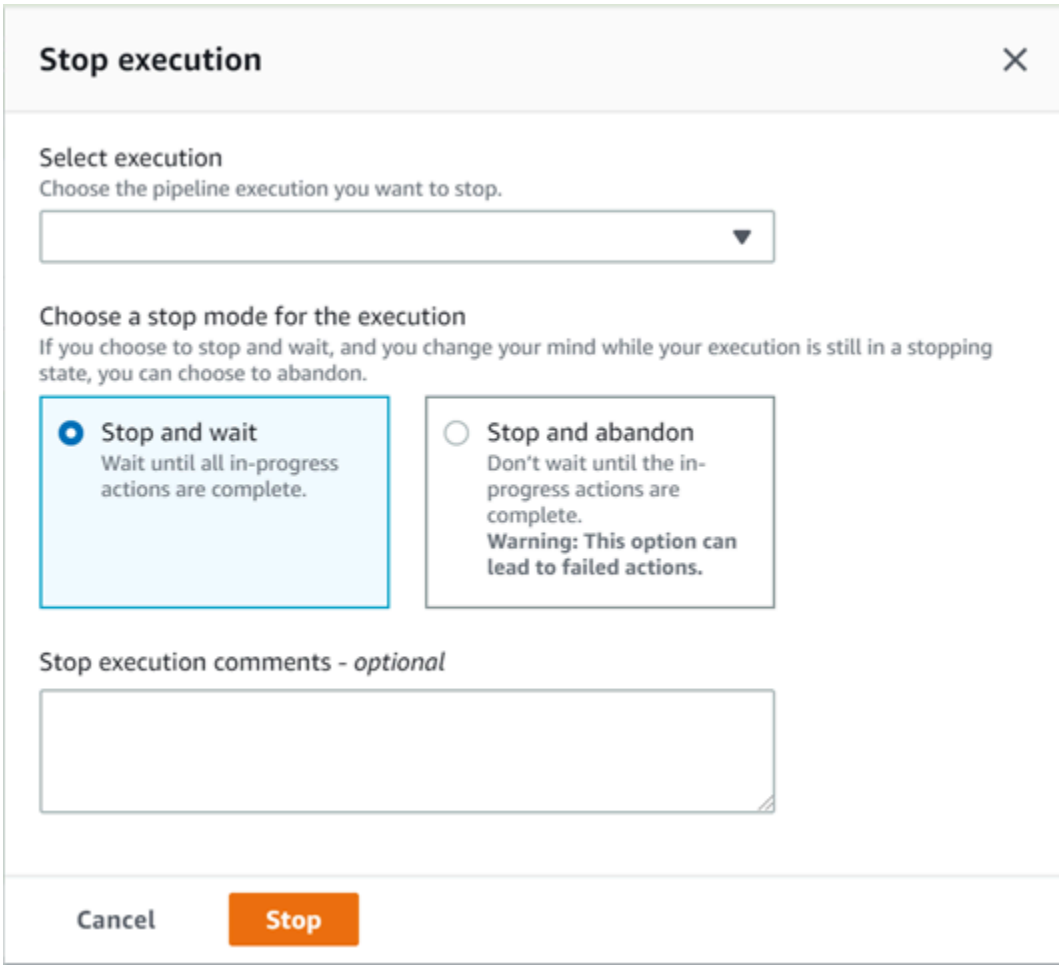
> **Note**
>
> È inoltre possibile interrompere l'esecuzione di una pipeline che è un'esecuzione in entrata. Per ulteriori informazioni sull'interruzione di un'esecuzione in entrata, consulta. Interrompere un'esecuzione in entrata (console)

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Esegui una di queste operazioni:

   > **Note**
   >
   > Prima di interrompere un'esecuzione, è consigliabile disabilitare la transizione davanti allo stage. In questo modo, quando lo stage si sblocca a causa dell'esecuzione interrotta, lo stage non accetta un'esecuzione successiva della pipeline.

   - In Nome, scegliere il nome della pipeline con l'esecuzione che si desidera interrompere. Nella pagina dei dettagli della pipeline scegliere Interrompi esecuzione.
   - Scegli View history (Visualizza cronologia). Nella pagina Cronologia scegliere Interrompi esecuzione.
3. Nella pagina Interrompi esecuzione in Seleziona esecuzione, scegliere l'esecuzione che si desidera interrompere.

   > **Note**
   >
   > L'esecuzione viene visualizzata solo se è ancora in corso. Le esecuzioni già completate non vengono visualizzate.



4. In Selezionare un'azione da applicare all'esecuzione, scegliere una delle seguenti opzioni:

   - Per assicurarsi che l'esecuzione non si interrompa fino al completamento di tutte le azioni in corso, scegliere Interrompi e attendi.

     > **Note**
     >
     > Non è possibile scegliere di interrompere e attendere se l'esecuzione è già in uno stato di arresto ma è possibile scegliere di interrompere e abbandonare.

   - Per interrompere senza attendere il completamento delle azioni in corso, scegliere Arresta e abbandona.

> **Warning**
>
> Questa opzione può portare ad attività non riuscite o fuori sequenza.

5. (Facoltativo) Inserisci commenti. Questi commenti, insieme allo stato di esecuzione, vengono visualizzati nella pagina della cronologia per l'esecuzione.
6. Scegli Stop (Arresta).

> **Important**
>
> Questa operazione non può essere annullata.

7. Visualizzare lo stato di esecuzione nella visualizzazione della pipeline come segue:

   - Se si sceglie di interrompere e attendere, l'esecuzione selezionata continua fino al completamento delle azioni in corso.
     - Il messaggio del banner di successo viene visualizzato nella parte superiore della console.
     - Nella fase attuale, le azioni in corso continuano in uno stato `InProgress`. Mentre le azioni sono in corso, l'esecuzione della pipeline è in uno stato `Stopping`.

       Al termine delle azioni (ovvero l'azione non riesce o riesce), l'esecuzione della pipeline cambia in uno stato `Stopped` e l'azione cambia in uno stato `Failed` o `Succeeded`. È inoltre possibile visualizzare lo stato dell'azione nella pagina dei dettagli dell'esecuzione. È possibile visualizzare lo stato di esecuzione nella pagina della cronologia dell'esecuzione o nella pagina dei dettagli dell'esecuzione.
     - L'esecuzione della pipeline cambia brevemente in uno stato `Stopping` e quindi cambia in uno stato `Stopped`. È possibile visualizzare lo stato di esecuzione nella pagina della cronologia dell'esecuzione o nella pagina dei dettagli dell'esecuzione.
   - Se si sceglie di interrompere e abbandonare, l'esecuzione non attende il completamento delle azioni in corso.
     - Il messaggio del banner di successo viene visualizzato nella parte superiore della console.
     - Nella fase attuale, le azioni in corso cambiano in uno stato di `Abandoned`. È inoltre possibile visualizzare lo stato dell'azione nella pagina dei dettagli dell'esecuzione.

- L'esecuzione della pipeline cambia brevemente in uno stato `Stopping` e quindi cambia in uno stato `Stopped`. È possibile visualizzare lo stato di esecuzione nella pagina della cronologia dell'esecuzione o nella pagina dei dettagli dell'esecuzione.

È possibile visualizzare lo stato di esecuzione della pipeline nella visualizzazione cronologia delle esecuzioni e nella visualizzazione cronologia dettagliata.

Interrompere un'esecuzione in entrata (console)

È possibile utilizzare la console per interrompere un'esecuzione in entrata. Un'esecuzione in entrata è un'esecuzione della pipeline in attesa di entrare in una fase in cui la transizione è stata disabilitata. Quando la transizione è abilitata, un'esecuzione in entrata `InProgress` continua a entrare nella fase. Un'esecuzione in entrata che `Stopped` lo è non entra nella fase.

> **Note**
>
> Dopo che un'esecuzione in entrata è stata interrotta, non può essere ritentata.

Se non viene visualizzata un'esecuzione in entrata, significa che non vi sono esecuzioni in sospeso in una fase di transizione disattivata.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Verranno visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. Scegli il nome della pipeline di cui desideri interrompere l'esecuzione in entrata, esegui una delle seguenti operazioni:

   - Nella vista Pipeline, scegliete l'ID di esecuzione in entrata, quindi scegliete di interrompere l'esecuzione.
   - Scegli la pipeline e scegli Visualizza cronologia. Nella cronologia di esecuzione, scegli l'ID di esecuzione in entrata, quindi scegli di interrompere l'esecuzione.
3. Nella modalità Interrompi l'esecuzione, segui i passaggi nella sezione precedente per selezionare l'ID di esecuzione e specificare il metodo di interruzione.

   Utilizzate il get-pipeline-state comando per visualizzare lo stato dell'esecuzione in entrata.

Arresto dell'esecuzione di una pipeline (CLI)

Per utilizzare il AWS CLI per arrestare manualmente una pipeline, utilizzate il stop-pipeline-execution comando con i seguenti parametri:

- ID esecuzione (obbligatorio)
- Commenti (facoltativo)
- Nome della pipeline (obbligatorio)
- Abbandona flag (facoltativo, il valore predefinito è falso)

Formato comando:

```
aws codepipeline stop-pipeline-execution --pipeline-name Pipeline_Name --pipeline-
execution-id Execution_ID [--abandon | --no-abandon] [--reason STOP_EXECUTION_REASON]
```

1. Apri un terminale (Linux, macOS o Unix) o prompt dei comandi (Windows).
2. Per interrompere l'esecuzione di una pipeline, scegliere una delle seguenti opzioni:

   - Per assicurarsi che l'esecuzione non si interrompa fino al completamento di tutte le azioni in corso, scegliere di interrompere e attendere. È possibile farlo includendo il parametro no-abandon. Se non si specifica il parametro, per default il comando viene impostato su arresto e attesa. Utilizzate il AWS CLI per eseguire il stop-pipeline-execution comando, specificando il nome della pipeline e l'ID di esecuzione. Ad esempio, per interrompere una pipeline denominata *MyFirstPipeline*con l'opzione stop and wait specificata:

     ```
     aws codepipeline stop-pipeline-execution --pipeline-name MyFirstPipeline --
     pipeline-execution-id d-EXAMPLE --no-abandon
     ```

     Ad esempio, per interrompere una pipeline denominata *MyFirstPipeline*, impostando di default l'opzione stop and wait e scegliendo di includere commenti:

     ```
     aws codepipeline stop-pipeline-execution --pipeline-name MyFirstPipeline --
     pipeline-execution-id d-EXAMPLE --reason "Stopping execution after the build
      action is done"
     ```

> **Note**
>
> Non è possibile scegliere di interrompere e attendere se l'esecuzione è già in uno stato di arresto . È possibile scegliere di interrompere e abbandonare un'esecuzione già in uno stato di arresto .

- Per fermarsi senza attendere il completamento delle azioni in corso, scegliere di interrompere e abbandonare. Includere il parametro abandon. Utilizzate il AWS CLI per eseguire il stop-pipeline-execution comando, specificando il nome della pipeline e l'ID di esecuzione.

  Ad esempio, per interrompere una pipeline denominata *MyFirstPipeline*, specificando l'opzione di abbandono e scegliendo di includere commenti:

```
aws codepipeline stop-pipeline-execution --pipeline-name MyFirstPipeline --
pipeline-execution-id d-EXAMPLE --abandon --reason "Stopping execution for a bug
 fix"
```

Interruzione di un'esecuzione in entrata (CLI)

È possibile utilizzare la CLI per interrompere un'esecuzione in entrata. Un'esecuzione in entrata è un'esecuzione della pipeline in attesa di entrare in una fase in cui la transizione è stata disabilitata. Quando la transizione è abilitata, un'esecuzione in entrata `InProgress` continua a entrare nella fase. Un'esecuzione in entrata che `Stopped` lo è non entra nella fase.

> **Note**
>
> Dopo che un'esecuzione in entrata è stata interrotta, non può essere ritentata.

Se non viene visualizzata un'esecuzione in entrata, significa che non vi sono esecuzioni in sospeso in una fase di transizione disattivata.

Per utilizzare il AWS CLI per interrompere manualmente un'esecuzione in entrata, utilizzate il stop-pipeline-execution comando con i seguenti parametri:

- ID di esecuzione in entrata (obbligatorio)
- Commenti (facoltativo)

- Nome della pipeline (obbligatorio)
- Abbandona flag (facoltativo, il valore predefinito è falso)

Formato comando:

```
aws codepipeline stop-pipeline-execution --pipeline-name Pipeline_Name --
pipeline-execution-id Inbound_Execution_ID [--abandon | --no-abandon] [--
reason STOP_EXECUTION_REASON]
```

Segui i passaggi della procedura precedente per immettere il comando e specificare il metodo di arresto.

Utilizzate il get-pipeline-state comando per visualizzare lo stato dell'esecuzione in entrata.

Creare una pipeline in CodePipeline

È possibile utilizzare la AWS CodePipeline console o il AWS CLI per creare una pipeline. Le pipeline devono avere almeno due fasi. La prima fase di una pipeline deve essere una fase di origine. La pipeline deve avere almeno un'altra fase che è una fase di compilazione o distribuzione.

Puoi aggiungere azioni alla tua pipeline che si trovano in una AWS regione diversa dalla tua pipeline. Un'azione interregionale è un'azione in cui an Servizio AWS è il fornitore di un'azione e il tipo di azione o il tipo di provider si trova in una AWS regione diversa dalla pipeline. Per ulteriori informazioni, consulta Aggiungere un'azione interregionale in CodePipeline.

Puoi anche creare pipeline per creare e distribuire applicazioni basate su container utilizzando Amazon ECS come fornitore di distribuzione. Prima di creare una pipeline che distribuisca applicazioni basate su container con Amazon ECS, devi creare un file di definizioni delle immagini come descritto in. Riferimento per il file di definizioni delle immagini

CodePipeline utilizza metodi di rilevamento delle modifiche per avviare la pipeline quando viene effettuata una modifica al codice sorgente. Questi metodi di rilevamento sono basati sul tipo di origine:

- CodePipeline utilizza Amazon CloudWatch Events per rilevare le modifiche nel repository e nella filiale di CodeCommit origine o nel bucket di origine S3.

> **Note**
>
> Quando utilizzi la console per creare o modificare una pipeline, le risorse di rilevamento delle modifiche vengono create automaticamente. Se utilizzi la AWS CLI per creare la pipeline, devi creare tu stesso le risorse aggiuntive. Per ulteriori informazioni, consulta CodeCommit azioni di origine e EventBridge.

Argomenti

Creazione di una pipeline (console)

Per creare una pipeline nella console, occorre specificare il percorso dei file di origine e le informazioni relative ai provider che verranno utilizzati per le operazioni.

Quando si utilizza la console per creare una pipeline, occorre includere una fase di origine e una o entrambe le fasi seguenti:

- Una fase di compilazione.
- Una fase di distribuzione.

Quando si utilizza la procedura guidata per la pipeline, CodePipeline crea i nomi delle fasi (source, build, staging). Questi nomi non possono essere modificati. Puoi usare nomi più specifici (ad esempio BuildToGamma o DeployToProd) per le fasi che aggiungi in un secondo momento.

Fase 1: creare e assegnare un nome alla pipeline

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Welcome (Benvenuto), seleziona Create pipeline (Crea pipeline).

   Se è la prima volta che lo usi CodePipeline, scegli Inizia.
3. Nella pagina Step 1: Choose pipeline settings (Fase 1: scelta delle impostazioni della pipeline), in Pipeline name (Nome pipeline), immettere il nome della pipeline.

   In un singolo AWS account, ogni pipeline creata in una AWS regione deve avere un nome univoco. I nomi possono essere riutilizzati per pipeline in regioni diverse.

   > **Note**
   >
   > Non è possibile modificare il nome di una pipeline dopo che è stata creata. Per informazioni su altre limitazioni, consulta Quote in AWS CodePipeline.
4. In Tipo di tubazione, scegliete una delle seguenti opzioni. I tipi di tubazioni differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.

   - Le pipeline di tipo V1 hanno una struttura JSON che contiene parametri standard di pipeline, fase e livello di azione.
   - Le pipeline di tipo V2 hanno la stessa struttura di un tipo V1, insieme al supporto di parametri aggiuntivi, come i trigger sui tag Git e le variabili a livello di pipeline.
5. In Service role (Ruolo del servizio), procedere in uno dei seguenti modi:

   - Scegli Nuovo ruolo di servizio per consentire la creazione di un nuovo ruolo di servizio CodePipeline in IAM.
   - Scegli Existing service role (Ruolo di servizio esistente) per usare un ruolo del servizio già creato in IAM. In Role ARN (ARN del ruolo), scegliere l'ARN del ruolo del servizio dall'elenco.

   > **Note**
   >
   > A seconda di quando è stato creato il ruolo di servizio, potrebbe essere necessario aggiornarne le autorizzazioni per supportarne altre Servizi AWS. Per informazioni, consulta Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline.

Per ulteriori informazioni sul ruolo del servizio e la relativa dichiarazione di policy, consulta Gestisci il ruolo CodePipeline del servizio.

6. (Facoltativo) In Variabili, scegliete Aggiungi variabile per aggiungere variabili a livello di pipeline.

   Per ulteriori informazioni sulle variabili a livello di pipeline, consulta. Variables Per un tutorial con una variabile a livello di pipeline che viene passata al momento dell'esecuzione della pipeline, consulta. Tutorial: utilizzare le variabili a livello di pipeline

   > **Note**
   >
   > Sebbene sia facoltativo aggiungere variabili a livello di pipeline, per una pipeline specificata con variabili a livello di pipeline in cui non vengono forniti valori, l'esecuzione della pipeline avrà esito negativo.

7. (Facoltativo) Espandi Advanced settings (Impostazioni avanzate).
8. In Artifact store (Archivio di artefatti), esegui una delle seguenti operazioni:

   a. Scegliete Posizione predefinita per utilizzare l'archivio di artefatti predefinito, ad esempio il bucket di artefatti S3 designato come predefinito, per la pipeline nell'area selezionata per la pipeline. Regione AWS

   b. Scegliere Custom location (Posizione personalizzata) se si dispone già di uno store di artefatti, ad esempio un bucket S3 dedicato agli artefatti, nella stessa regione della pipeline. In Bucket, scegli il nome del bucket.

   > **Note**
   >
   > Non si tratta del bucket di origine per il codice sorgente, ma dell'archivio artefatti per la pipeline. È richiesto un archivio artefatti separato, ad esempio un bucket S3, per ogni pipeline. Quando crei o modifichi una pipeline, devi avere un bucket di artefatti nella regione della pipeline e un bucket di artefatti per regione in cui stai eseguendo un'azione. AWS
   >
   > Per ulteriori informazioni, consulta Artefatti di input e output e CodePipeline riferimento alla struttura della tubazione.

9. Per Encryption key (Chiave di crittografia), esegui una delle operazioni seguenti:

a. Per utilizzare l' CodePipeline impostazione predefinita per AWS KMS key crittografare i dati nell'archivio degli artifatti della pipeline (bucket S3), scegli Chiave gestita predefinita. AWS

b. Per utilizzare la chiave gestita dal cliente per crittografare i dati nel pipeline Artifact Store (bucket S3), scegli Customer Managed Key. Scegli l'ID della chiave, l'ARN della chiave o l'alias ARN.

10. Seleziona Successivo.

Fase 2: creare una fase di origine

- Nella pagina Step 2: Add source stage (Fase 2: aggiunta fase di origine), nell'elenco a discesa Source provider (Provider di origine), scegli il tipo di repository in cui viene archiviato il codice sorgente, specifica le relative opzioni obbligatorie, quindi scegli Next step (Fase successiva).

  - Per Bitbucket Cloud, GitHub (versione 2), GitHub Enterprise Server, GitLab .com o autogestito: GitLab

    1. In Connessione, scegli una connessione esistente o creane una nuova. Per creare o gestire una connessione per l'azione GitHub sorgente, consultaGitHub connessioni.
    2. Scegli il repository che desideri utilizzare come posizione di origine per la tua pipeline.

       Scegli di aggiungere un trigger o un filtro in base ai tipi di trigger per avviare la pipeline. Per ulteriori informazioni sull'utilizzo dei trigger, consulta. Filtra i trigger nelle richieste push o pull di codice Per ulteriori informazioni sul filtraggio con pattern a glob, vedere. Lavorare con i modelli a globo nella sintassi
    3. In Formato degli artefatti di output, scegli il formato per i tuoi artefatti.
       - Per memorizzare gli artefatti di output derivanti dall' GitHub azione utilizzando il metodo predefinito, scegliete predefinito. CodePipeline L'azione accede ai file dal GitHub repository e archivia gli artefatti in un file ZIP nel Pipeline Artifact Store.
       - Per archiviare un file JSON contenente un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git, scegliere Full clone (Clone completo). Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild

         Se scegli questa opzione, dovrai aggiornare le autorizzazioni per il tuo ruolo di CodeBuild Project Service come mostrato in. Risoluzione dei problemi CodePipeline Per un tutorial

che mostra come usare l'opzione Full clone, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline

- Per Amazon S3:

  1. In Amazon S3 location (Percorso Amazon S3), fornisci il nome e il percorso del bucket S3 all'oggetto in un bucket con abilitata la funzione Versioni multiple. Il formato del nome e del percorso del bucket è simile al seguente:

     ```
     s3://bucketName/folderName/objectName
     ```

     > **Note**
     >
     > Se Amazon S3 è il fornitore di origine per la tua pipeline, puoi comprimere il file o i file sorgente in un unico .zip e caricare il file.zip nel tuo bucket di origine. È inoltre possibile caricare un singolo file decompresso; tuttavia, le operazioni a valle che si aspettano un file con estensione .zip avranno esito negativo.

  2. Dopo aver scelto il bucket di origine S3, CodePipeline crea la regola Amazon CloudWatch Events e il AWS CloudTrail percorso da creare per questa pipeline. Accettare le impostazioni predefinite in Change detection options (Opzioni di rilevamento delle modifiche). Ciò consente di CodePipeline utilizzare Amazon CloudWatch Events e AWS CloudTrail di rilevare le modifiche per la nuova pipeline. Seleziona Successivo.

- Per AWS CodeCommit:

  - In Nome repository, scegli il nome del CodeCommit repository che desideri utilizzare come posizione di origine per la tua pipeline. Nell'elenco a discesa in Branch name (Nome ramo), scegli il ramo che desideri utilizzare.
  - Nel Formato degli artefatti di output, scegli il formato per i tuoi artefatti.
    - Per memorizzare gli artefatti di output derivanti dall' CodeCommit azione utilizzando il metodo predefinito, scegliete predefinito. CodePipeline L'azione accede ai file dal CodeCommit repository e archivia gli artefatti in un file ZIP nel Pipeline Artifact Store.
    - Per archiviare un file JSON contenente un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git, scegliere Full clone (Clone completo). Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild

      Se scegli questa opzione, dovrai aggiungere l'`codecommit:GitPull`autorizzazione al tuo ruolo di CodeBuild servizio, come mostrato inAggiungi le CodeBuild GitClone

autorizzazioni per le azioni di origine CodeCommit. Dovrai anche aggiungere le `codecommit:GetRepository` autorizzazioni al tuo ruolo di CodePipeline servizio, come mostrato inAggiunta delle autorizzazioni dal ruolo di servizio CodePipeline. Per un tutorial che mostra come usare l'opzione Full clone, vedi. Tutorial: usa il clone completo con una sorgente di GitHub pipeline

- Dopo aver scelto il nome e il ramo del CodeCommit repository, nelle opzioni di rilevamento delle modifiche viene visualizzato un messaggio che mostra la regola Amazon CloudWatch Events da creare per questa pipeline. Accettare le impostazioni predefinite in Change detection options (Opzioni di rilevamento delle modifiche). Ciò consente di CodePipeline utilizzare Amazon CloudWatch Events per rilevare le modifiche alla tua nuova pipeline.

- Per Amazon ECR:
  - In Nome repository, scegli il nome del tuo repository Amazon ECR.
  - In Image tag (Tag immagine), specifica il nome e la versione dell'immagine, se diverso da LATEST.
  - In Elementi di output, scegli l'elemento di output predefinito, ad esempio quello che contiene il nome dell'immagine e le informazioni URI del repository che desideri utilizzare nella fase successiva. MyApp

    Per un tutorial sulla creazione di una pipeline per Amazon ECS con distribuzioni CodeDeploy blu-verdi che include una fase sorgente Amazon ECR, consulta. Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy

  Quando includi una fase sorgente di Amazon ECR nella tua pipeline, l'azione di origine genera un `imageDetail.json` file come artefatto di output quando esegui una modifica. Per ulteriori informazioni sul file `imageDetail.json`, consulta File ImageDetail.json per le azioni di distribuzione blu/verde di Amazon ECS.

> **Note**
>
> L'oggetto e il tipo di file devono essere compatibili con il sistema di distribuzione che intendi utilizzare (ad esempio, Elastic Beanstalk o). CodeDeploy I tipi di file supportati potrebbero includere i file .zip, .tar e .tgz. Per ulteriori informazioni sui tipi di contenitori supportati per Elastic Beanstalk, consulta Personalizzazione e configurazione degli ambienti Elastic Beanstalk e delle piattaforme supportate. Per ulteriori informazioni sulla

distribuzione delle revisioni con CodeDeploy, consulta Caricamento della revisione dell'applicazione e Preparazione di una revisione.

Fase 3: creare una fase di compilazione

Questa fase è facoltativa se prevedi di creare una fase di distribuzione.

- Nella pagina Step 3: Add build stage (Fase 3: aggiunta fase di compilazione), esegui una delle operazioni seguenti, quindi seleziona Next (Successivo):
  - Scegli Skip build stage (Salta la fase di compilazione) se prevedi di creare una fase di distribuzione.
  - Da Build provider (Provider di compilazione), scegli un provider di operazioni personalizzate di servizi di compilazione e fornisci i dettagli di configurazione per quel provider. Per un esempio di come aggiungere Jenkins come un provider di compilazione, consulta Tutorial: creazione di una pipeline a quattro fasi.
  - In Build provider (Provider compilazione), seleziona AWS CodeBuild.

    In Regione, scegli la regione in cui si trova la AWS risorsa. Il campo Regione indica dove vengono create le AWS risorse per questo tipo di azione e tipo di provider. Questo campo viene visualizzato solo per le azioni in cui il fornitore dell'azione è un Servizio AWS. Per impostazione predefinita, il campo Regione è la stessa AWS della pipeline.

    In Project name (Nome progetto), scegli il progetto di compilazione. Se hai già creato un progetto di compilazione in CodeBuild, sceglilo. Oppure puoi creare un progetto di compilazione CodeBuild e poi tornare a questa attività. Segui le istruzioni riportate in Creare una pipeline da utilizzare CodeBuild nella Guida per l'CodeBuildutente.

    In Variabili di ambiente, per aggiungere variabili di CodeBuild ambiente all'azione di compilazione, scegli Aggiungi variabile di ambiente. Ogni variabile è composta da tre elementi:
    - Name (Nome), immetti il nome o la chiave della variabile di ambiente.
    - Value (Valore), immetti il valore della variabile di ambiente. Se scegliete Parameter per il tipo di variabile, assicuratevi che questo valore sia il nome di un parametro che avete già memorizzato in AWS Systems Manager Parameter Store.

> **Note**
>
> Sconsigliamo vivamente l'uso di variabili di ambiente per archiviare valori sensibili, in particolare AWS le credenziali. Quando si utilizza la CodeBuild console o la AWS CLI, le variabili di ambiente vengono visualizzate in testo semplice. Per i valori sensibili, si consiglia di utilizzare invece il tipo Parameter (Parametro).

- (Facoltativo) In Type (Tipo), immetti il tipo di variabile di ambiente. I valori validi sono Plaintext (Testo non crittografato) o Parameter (Parametro). Il valore predefinito è Plaintext (Testo non crittografato).

(Facoltativo) In Tipo di build, scegliete una delle seguenti opzioni:

- Per eseguire ogni build in un'unica azione di compilazione, scegli Single build.
- Per eseguire più build nella stessa esecuzione della stessa azione di compilazione, scegli Creazione in Batch.

(Facoltativo) Se scegli di eseguire build in batch, puoi scegliere Combina tutti gli artefatti del batch in un'unica posizione per collocare tutti gli artefatti di build in un unico artefatto di output.

Fase 4: creare una fase di distribuzione

Questa fase è facoltativa se hai già creato una fase di compilazione.

- Nella pagina Step 4: Add deploy stage (Fase 4: aggiunta fase di distribuzione), esegui una delle operazioni seguenti, quindi scegli Next (Successivo):
  - Scegli Skip deploy stage (Ignora fase di distribuzione) se hai creato una fase di compilazione nella fase precedente.

    > **Note**
    >
    > Questa opzione non viene visualizzata se hai già ignorato la fase di compilazione.

  - In Deploy provider (Provider di distribuzione), scegli un'operazione personalizzata che hai creato per un provider di distribuzione.

    In Regione, solo per le azioni interregionali, scegliete la regione in cui viene creata la risorsa. AWS Il campo Regione indica dove vengono create le AWS risorse per questo tipo di azione e

tipo di provider. Questo campo viene visualizzato solo per le azioni in cui il fornitore dell'azione è un Servizio AWS. Per impostazione predefinita, il campo Regione è la stessa AWS della pipeline.

- In Deploy provider (Distribuisci provider), i campi sono disponibili per i provider predefiniti come segue:
  - CodeDeploy

    In Nome applicazione, inserisci o scegli il nome di un'applicazione esistente. CodeDeploy In Deployment group (Gruppo di distribuzione), immetti il nome di un gruppo di distribuzione per l'applicazione. Seleziona Successivo. È inoltre possibile creare un'applicazione, un gruppo di distribuzione o entrambi nella CodeDeploy console.
  - AWS Elastic Beanstalk

    In Nome applicazione, inserisci o scegli il nome di un'applicazione Elastic Beanstalk esistente. In Environment name (Nome ambiente), immetti un ambiente per l'applicazione. Seleziona Successivo. Puoi anche creare un'applicazione, un ambiente o entrambi nella console Elastic Beanstalk.
  - AWS OpsWorks Stacks

    In Stack, immetti o scegli il nome dello stack da utilizzare. In Layer (Livello), scegliere il livello cui appartengono le istanze di destinazione. In App, scegliere l'applicazione da aggiornare e distribuire. Se devi creare un'app, scegli Creane una nuova in. AWS OpsWorks

    Per informazioni sull'aggiunta di un'applicazione a uno stack e al layer in AWS OpsWorks, consulta Aggiungere app nella Guida per l'AWS OpsWorks utente.

    Per un end-to-end esempio di come utilizzare una pipeline semplice CodePipeline come sorgente per il codice da eseguire su AWS OpsWorks livelli, consulta Using CodePipeline with. AWS OpsWorks Stacks
  - AWS CloudFormation

    Esegui una di queste operazioni:

    - In modalità Azione, scegliete Crea o aggiorna uno stack, inserite il nome dello stack e il nome del file del modello, quindi scegliete il nome del ruolo da assumere. AWS CloudFormation Facoltativamente, inserisci il nome di un file di configurazione e scegli un'opzione di funzionalità IAM.

- In modalità Azione, scegli Crea o sostituisci un set di modifiche, inserisci il nome dello stack e il nome del set di modifiche, quindi scegli il nome del ruolo AWS CloudFormation da assumere. Facoltativamente, inserisci il nome di un file di configurazione e scegli un'opzione di funzionalità IAM.

Per informazioni sull'integrazione AWS CloudFormation delle funzionalità in una pipeline in CodePipeline, consulta Continuous Delivery with CodePipeline nella Guida per l'AWS CloudFormation utente.

- Amazon ECS

In Nome cluster, inserisci o scegli il nome di un cluster Amazon ECS esistente. In Service name (Nome servizio), immetti o scegli il nome del servizio in esecuzione sul cluster. Puoi anche creare un cluster e un servizio. In Image filename (Nome file di immagine), immetti il nome del file di definizioni delle immagini che descrive il container e l'immagine del servizio.

> **Note**
>
> L'azione di distribuzione di Amazon ECS richiede un `imagedefinitions.json` file come input per l'azione di distribuzione. Il nome file predefinito è imagedefinitions.json. Se scegli di utilizzare un nome di file diverso, è necessario fornirlo durante la creazione della fase di distribuzione della pipeline. Per ulteriori informazioni, consulta la pagina file imagedefinitions.json per le azioni di distribuzione standard di Amazon ECS.

Seleziona Next (Successivo).

> **Note**
>
> Assicurati che il tuo cluster Amazon ECS sia configurato con due o più istanze. I cluster Amazon ECS devono contenere almeno due istanze in modo che una venga mantenuta come istanza principale e l'altra venga utilizzata per ospitare nuove distribuzioni.

Per un tutorial sulla distribuzione di applicazioni basate su container con la tua pipeline, consulta Tutorial: Continuous Deployment with. CodePipeline

- Amazon ECS (Blu/verde)

  Inserisci l' CodeDeploy applicazione e il gruppo di distribuzione, la definizione delle attività di Amazon ECS e le informazioni sui AppSpec file, quindi scegli Avanti.

  > **Note**
  >
  > L'operazione Amazon ECS (Blue/Green) (Amazon ECS (blu/verde)) richiede un file imageDetail.json come artefatto di input per l'operazione di distribuzione. Poiché l'azione di origine di Amazon ECR crea questo file, non è necessario che le pipeline con un'azione di origine Amazon ECR forniscano un file. `imageDetail.json` Per ulteriori informazioni, consulta File ImageDetail.json per le azioni di distribuzione blu/verde di Amazon ECS.

  Per un tutorial sulla creazione di una pipeline per distribuzioni blu-verdi in un cluster Amazon ECS con, consulta. CodeDeploy Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy

- AWS Service Catalog

  Scegli Enter deployment configuration (Inserisci configurazione della distribuzione) se desideri utilizzare i campi nella console per specificare la configurazione oppure scegli Configuration file (File di configurazione) se disponi di un file di configurazione separato. Inserisci le informazioni di prodotto e configurazione, quindi scegli Next (Successivo).

  Per un tutorial sull'implementazione delle modifiche ai prodotti in Service Catalog con la tua pipeline, consulta. Tutorial: crea una pipeline da distribuire su Service Catalog

- Alexa Skills Kit

  In Alexa Skill ID (ID competenza Alexa), immetti l'ID competenza per la competenza Alexa. In Client ID (ID client) e Client secret (Segreto client), immetti le credenziali generate utilizzando un profilo di sicurezza Login with Amazon (LWA). In Refresh token (Token di aggiornamento), immetti il token di aggiornamento generato usando il comando dell'interfaccia a riga di comando ASK per il recupero di un token di aggiornamento. Seleziona Successivo.

Per un tutorial sulla distribuzione di competenze Alexa con la pipeline e la generazione di credenziali LWA, consulta Tutorial: creazione di una pipeline che distribuisce una competenza Amazon Alexa.

- Amazon S3

In Bucket, inserisci il nome del bucket S3 da usare. Scegli Extract file before deploy (Estrai file prima di distribuire) se l'artefatto di input per la fase di distribuzione è un file ZIP. Se Extract file before deploy (Estrai file prima di distribuire) è selezionato, è possibile inserire un valore per Deployment path (Percorso di distribuzione) dove il file ZIP verrà decompresso. Se non è selezionato, è necessario immettere un valore in S3 object key (Chiave oggetto S3).

> **Note**
>
> La maggior parte degli artefatti di output delle fasi di origine e compilazione vengono compressi. Tutti i fornitori di sorgenti di pipeline, ad eccezione di Amazon S3, comprimono i file sorgente prima di fornirli come elemento di input per l'azione successiva.

(Facoltativo) In ACL predefinito, inserisci l'ACL predefinito da applicare all'oggetto distribuito su Amazon S3.

> **Note**
>
> L'applicazione di una ACL predefinita sovrascrive le ACL esistenti applicate all'oggetto.

(Facoltativo) In Cache control (Controllo della cache), specifica i parametri del controllo della cache per le richieste di download degli oggetti dal bucket. Per un elenco dei valori validi, vedi il campo di intestazione `Cache-Control` per le operazioni HTTP. Per inserire più valori in Cache control (Controllo cache), utilizzare una virgola tra ogni valore. È possibile aggiungere uno spazio dopo ogni virgola (facoltativo), come mostrato in questo esempio.

La voce di esempio precedente viene visualizzata nell'interfaccia a riga di comando come segue:

```
"CacheControl": "public, max-age=0, no-transform"
```

Seleziona Successivo.

Per un tutorial sulla creazione di una pipeline con un provider di azioni di distribuzione Amazon S3, consulta. Tutorial: crea una pipeline che utilizzi Amazon S3 come provider di distribuzione

Fase 5: Rivedere la pipeline

- Nella pagina Step 5: Review (Fase 5: revisione), verifica la configurazione della pipeline e scegli Create pipeline (Crea pipeline) per creare la pipeline o Previous (Precedente) per tornare indietro e modificare le scelte. Per uscire dalla procedura guidata senza creare una pipeline, scegliere Cancel (Annulla).

Ora che hai creato la pipeline, puoi visualizzarla nella console. La pipeline viene avviata dopo che è stata creata. Per ulteriori informazioni, consulta Visualizza le pipeline e i dettagli in CodePipeline. Per ulteriori informazioni su come apportare modifiche alla pipeline, consulta Modificare una tubazione in CodePipeline.

Creazione di una pipeline (CLI)

Per utilizzare il AWS CLI per creare una pipeline, devi creare un file JSON per definire la struttura della pipeline, quindi eseguire il comando con il create-pipeline parametro. `--cli-input-json`

> **Important**
>
> Non è possibile utilizzare il AWS CLI per creare una pipeline che includa le azioni dei partner. È invece necessario utilizzare la CodePipeline console.

Per ulteriori informazioni sulla struttura della pipeline, consulta CodePipeline riferimento alla struttura della tubazione e create-pipeline nell'API Reference. CodePipeline

Per creare un file JSON, utilizza il file JSON della pipeline di esempio, modificarlo e quindi chiamare il file quando si esegue il comando create-pipeline.

Prerequisiti:

È necessario l'ARN del ruolo di servizio per CodePipeline cui è stato creato. Guida introduttiva con CodePipeline Si utilizza il ruolo CodePipeline di servizio ARN nel file JSON della pipeline quando si esegue il comando. create-pipeline Per ulteriori informazioni sulla creazione di un ruolo del servizio, consulta Creare il ruolo CodePipeline di servizio. A differenza della console, eseguendo il create-pipeline comando in AWS CLI non è possibile creare automaticamente il ruolo CodePipeline di servizio. Il ruolo del servizio deve essere già presente.

Hai bisogno del nome di un bucket S3 dove archiviare gli artefatti per la pipeline. Questo bucket deve trovarsi nella stessa regione della pipeline. Puoi usare il nome del bucket nel file JSON della pipeline quando esegui il comando create-pipeline. A differenza della console, l'esecuzione del create-pipeline comando in AWS CLI non crea un bucket S3 per l'archiviazione degli artefatti. Il bucket deve esistere già.

> **Note**
>
> Puoi anche utilizzare il comando get-pipeline per ottenere una copia della struttura JSON di tale pipeline, quindi modificare la struttura in un editor di testo semplice.

Per creare il file JSON

1. In un terminale (Linux, macOS o Unix) o al prompt dei comandi (Windows), create un nuovo file di testo in una directory locale.
2. (Facoltativo) È possibile aggiungere una o più variabili a livello di pipeline. È possibile fare riferimento a questo valore nella configurazione delle CodePipeline azioni. Puoi aggiungere i nomi e i valori delle variabili quando crei la pipeline e puoi anche scegliere di assegnare valori quando avvii la pipeline nella console.

   > **Note**
   >
   > Sebbene sia facoltativo aggiungere variabili a livello di pipeline, per una pipeline specificata con variabili a livello di pipeline in cui non vengono forniti valori, l'esecuzione della pipeline avrà esito negativo.

Una variabile a livello di pipeline viene risolta in fase di esecuzione della pipeline. Tutte le variabili sono immutabili, il che significa che non possono essere aggiornate dopo l'assegnazione di un valore. Le variabili a livello di pipeline con valori risolti verranno visualizzate nella cronologia di ogni esecuzione.

Le variabili vengono fornite a livello di pipeline utilizzando l'attributo variables nella struttura della pipeline. Nell'esempio seguente, la variabile `Variable1` ha un valore di. `Value1`

```
       "variables": [
            {
                "name": "Timeout",
                "defaultValue": "1000",
                "description": "description"
            }
        ]
```

Aggiungi questa struttura alla tua pipeline JSON o all'esempio JSON nel passaggio successivo. Per ulteriori informazioni sulle variabili, incluse le informazioni sullo spazio dei nomi, consulta. Variables

3. Aprire il file in un editor di testo semplice e modificare i valori in modo da riflettere la struttura da creare. Come minimo, occorre modificare il nome della pipeline. Occorre anche considerare se modificare:

   - Il bucket S3 in cui vengono archiviati artefatti per questa pipeline.
   - Il percorso di origine del codice.
   - Il provider di distribuzione.
   - La modalità di distribuzione del codice.
   - I tag per la pipeline.

   La seguente struttura della pipeline di esempio con due fasi evidenzia i valori che occorre modificare per la pipeline. La pipeline contiene probabilmente più di due fasi:

```
{
    "pipeline": {
        "roleArn": "arn:aws:iam::80398EXAMPLE::role/AWS-CodePipeline-Service",
        "stages": [
```

```
            {
                "name": "Source",
                "actions": [
                    {
                        "inputArtifacts": [],
                        "name": "Source",
                        "actionTypeId": {
                            "category": "Source",
                            "owner": "AWS",
                            "version": "1",
                            "provider": "S3"
                        },
                        "outputArtifacts": [
                            {
                                "name": "MyApp"
                            }
                        ],
                        "configuration": {
                            "S3Bucket": "awscodepipeline-demobucket-example-date",
                            "S3ObjectKey": "ExampleCodePipelineSampleBundle.zip",
                            "PollForSourceChanges": "false"
                        },
                        "runOrder": 1
                    }
                ]
            },
            {
                "name": "Staging",
                "actions": [
                    {
                        "inputArtifacts": [
                            {
                                "name": "MyApp"
                            }
                        ],
                        "name": "Deploy-CodeDeploy-Application",
                        "actionTypeId": {
                            "category": "Deploy",
                            "owner": "AWS",
                            "version": "1",
                            "provider": "CodeDeploy"
                        },
                        "outputArtifacts": [],
                        "configuration": {
```

```
                        "ApplicationName": "CodePipelineDemoApplication",
                        "DeploymentGroupName": "CodePipelineDemoFleet"
                    },
                    "runOrder": 1
                }
            ]
        }
    ],
    "artifactStore": {
        "type": "S3",
        "location": "codepipeline-us-east-2-250656481468"
    },
    "name": "MyFirstPipeline",
    "version": 1,
    "variables": [
        {
            "name": "Timeout",
            "defaultValue": "1000",
            "description": "description"
        }
      ]
    },
    "triggers": [
        {
            "providerType": "CodeStarSourceConnection",
            "gitConfiguration": {
                "sourceActionName": "Source",
                "push": [
                    {
                        "tags": {
                            "includes": [
                                "v1"
                            ],
                            "excludes": [
                                "v2"
                            ]
                        }
                    }
                ]
            }
        }
    ]
"metadata": {
```

```
        "pipelineArn": "arn:aws:codepipeline:us-
east-2:80398EXAMPLE:MyFirstPipeline",
        "updated": 1501626591.112,
        "created": 1501626591.112
    },
    "tags": [{
      "key": "Project",
      "value": "ProjectA"
    }]
}
```

In questo esempio viene aggiunto tagging alla pipeline, includendo in essa la chiave di tag `ProjectA` e il valore `Project`. Per ulteriori informazioni sull'etichettatura delle risorse, consulta CodePipeline. Assegnazione di tag alle risorse

Assicurati che il parametro `PollForSourceChanges` nel file JSON sia impostato come segue:

```
                "PollForSourceChanges": "false",
```

CodePipeline utilizza Amazon CloudWatch Events per rilevare le modifiche nel repository e nella filiale di CodeCommit origine o nel bucket di origine S3. Il passaggio successivo include le istruzioni per creare manualmente queste risorse per la pipeline. L'impostazione del flag su `false` disabilita i controlli periodici, che non sono necessari quando si utilizzano i metodi di rilevamento delle modifiche consigliati.

4. Per creare un'operazione di compilazione, test o distribuzione in una regione differente rispetto a quella della pipeline, devi aggiungere i seguenti elementi alla struttura della pipeline. Per istruzioni, consulta Aggiungere un'azione interregionale in CodePipeline.

   - Aggiungi il parametro `Region` alla struttura della pipeline dell'operazione.
   - Usa il `artifactStores` parametro per specificare un bucket di artefatti per ogni AWS regione in cui hai un'azione.

5. Al termine, salvare il file con un nome del tipo **`pipeline.json`**.

Per creare una pipeline

1. Eseguire il comando create-pipeline e utilizzare il parametro `--cli-input-json` per specificare il file JSON creato in precedenza.

Per creare una pipeline denominata *MySecondPipeline*con un file JSON denominato pipeline.json che include il nome "*MySecondPipeline*" come valore `name` nel JSON, il comando sarebbe simile al seguente:

```
aws codepipeline create-pipeline --cli-input-json file://pipeline.json
```

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

Questo comando restituisce la struttura dell'intera pipeline creata.

2. Per visualizzare la pipeline, apri la CodePipeline console e selezionala dall'elenco delle pipeline oppure usa il comando. get-pipeline-state Per ulteriori informazioni, consulta Visualizza le pipeline e i dettagli in CodePipeline.
3. Se si utilizza l'interfaccia a riga di comando per creare una pipeline, occorre creare manualmente le risorse di rilevamento delle modifiche consigliate per la pipeline:
   - Per una pipeline con un CodeCommit repository, è necessario creare manualmente la regola CloudWatch Events, come descritto in. Creare una EventBridge regola per un' CodeCommit origine (CLI)
   - Per una pipeline con una fonte Amazon S3, devi creare manualmente la regola AWS CloudTrail e CloudWatch il trail Events, come descritto in. Azioni di origine di Amazon S3 e con EventBridge AWS CloudTrail

Azioni e risorse relative ai sorgenti di Amazon ECR EventBridge

Per aggiungere un'azione sorgente Amazon ECR CodePipeline, puoi scegliere tra:

- Utilizza la CodePipeline console Create pipeline wizard (Creazione di una pipeline (console)) o la pagina Edit action per scegliere l'opzione del provider Amazon ECR. La console crea una EventBridge regola che avvia la pipeline quando cambia la fonte.
- Utilizza la CLI per aggiungere la configurazione dell'ECRazione e creare risorse aggiuntive come segue:

- Usa la configurazione dell'azione di ECR esempio in Amazon ECR per creare la tua azione come mostrato inCreazione di una pipeline (CLI).
- Per impostazione predefinita, il metodo di rilevamento delle modifiche avvia la pipeline interrogando la fonte. È necessario disabilitare i controlli periodici e creare manualmente la regola di rilevamento delle modifiche. Utilizzare uno dei seguenti metodi: Crea una EventBridge regola per una fonte Amazon ECR (console)Crea una EventBridge regola per un codice sorgente Amazon ECR (CLI), oCrea una EventBridge regola per una fonte Amazon ECR (AWS CloudFormation modello) .

Argomenti

- Crea una EventBridge regola per una fonte Amazon ECR (console)
- Crea una EventBridge regola per un codice sorgente Amazon ECR (CLI)
- Crea una EventBridge regola per una fonte Amazon ECR (AWS CloudFormation modello)

Crea una EventBridge regola per una fonte Amazon ECR (console)

Per creare una EventBridge regola da utilizzare nelle CodePipeline operazioni (fonte Amazon ECR)

1. Apri la EventBridge console Amazon all'indirizzo https://console.aws.amazon.com/events/.
2. Nel pannello di navigazione seleziona Events (Eventi).
3. Scegli Crea regola, quindi in Origine evento, da Nome servizio, scegli Elastic Container Registry (ECR).
4. In Event Source (Origine eventi) scegli Event Pattern (Modello di eventi).

   Scegli Modifica, quindi incolla il seguente modello di eventi di esempio nella finestra Origine eventi per un repository `eb-test` con una versione di immagine con tag di `cli-testing`:

```
{
    "detail-type": [
        "ECR Image Action"
    ],
    "source": [
        "aws.ecr"
    ],
    "detail": {
        "action-type": [
            "PUSH"
```

```
        ],
        "image-tag": [
            "latest"
        ],
        "repository-name": [
            "eb-test"
        ],
        "result": [
            "SUCCESS"
        ]
    }
}
```

> **Note**
>
> Per visualizzare il modello di eventi completo supportato per gli eventi Amazon ECR, consulta Amazon ECR Events e/o EventBridge Amazon Elastic Container Registry Events.

5. Selezionare Salva.

   Nel riquadro Event Pattern Preview (Anteprima modello eventi), visualizzare la regola.

6. In Targets, scegli. CodePipeline

7. Immettete l'ARN della pipeline per la pipeline da avviare in base a questa regola.

> **Note**
>
> Puoi trovare l'ARN della pipeline nell'output dei metadati dopo aver eseguito il comando get-pipeline. Il formato dell'ARN della pipeline è il seguente:
> *arn:aws:codepipeline: regione: account: nome-pipeline*
> ARN della pipeline di esempio:
> `arn:aws:codepipeline:us-east-2:80398EXAMPLE:MyFirstPipeline`

8. Crea o specifica un ruolo del servizio IAM che conceda EventBridge le autorizzazioni per richiamare la destinazione associata alla tua regola (in questo caso, la destinazione è). EventBridge CodePipeline

   - Scegli Crea un nuovo ruolo per questa risorsa specifica per creare un ruolo di servizio che ti dia EventBridge le autorizzazioni per avviare le esecuzioni della pipeline.

- Scegli Usa il ruolo esistente per inserire un ruolo di servizio che ti dia EventBridge le autorizzazioni per avviare le esecuzioni della pipeline.

9. Rivedere la configurazione delle regole per accertarsi che soddisfi i requisiti.
10. Scegli Configura dettagli.
11. Nella pagina Configure rule details (Configura dettagli della regola), immetti un nome e una descrizione per la regola e quindi scegli State (Stato) per abilitare la regola.
12. Se la regola ti soddisfa, scegli Create rule (Crea regola).

Crea una EventBridge regola per un codice sorgente Amazon ECR (CLI)

Chiama il comando put-rule, specificando:

- Un nome che identifica in modo univoco la regola che stai creando. Questo nome deve essere univoco in tutte le pipeline che crei e CodePipeline associate al tuo account. AWS
- Il modello eventi per i campi di origine e di dettaglio utilizzati dalla regola. Per ulteriori informazioni, consulta Amazon EventBridge e Event Patterns.

Per creare una EventBridge regola con Amazon ECR come origine dell'evento e CodePipeline come destinazione

1. Aggiungi le autorizzazioni EventBridge da utilizzare per CodePipeline richiamare la regola. Per ulteriori informazioni, consulta Utilizzo delle politiche basate sulle risorse per Amazon. EventBridge

   a. Usa l'esempio seguente per creare la politica di fiducia che consente di EventBridge assumere il ruolo di servizio. Denomina la policy di attendibilità `trustpolicyforEB.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "events.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
```

```
}
```

b. Utilizza il seguente comando per creare il ruolo `Role-for-MyRule` e collegare la policy di attendibilità.

```
aws iam create-role --role-name Role-for-MyRule --assume-role-policy-document
 file://trustpolicyforEB.json
```

c. Crea il JSON della policy delle autorizzazioni, come mostrato in questo esempio, per la pipeline denominata `MyFirstPipeline`. Denomina la policy delle autorizzazioni `permissionspolicyforEB.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:StartPipelineExecution"
            ],
            "Resource": [
                "arn:aws:codepipeline:us-west-2:80398EXAMPLE:MyFirstPipeline"
            ]
        }
    ]
}
```

d. Utilizza il comando seguente per collegare la policy delle autorizzazioni `CodePipeline-Permissions-Policy-for-EB` al ruolo `Role-for-MyRule`.

Perché occorre apportare questa modifica? L'aggiunta di questa politica al ruolo crea le autorizzazioni per EventBridge.

```
aws iam put-role-policy --role-name Role-for-MyRule --policy-name CodePipeline-
Permissions-Policy-For-EB --policy-document file://permissionspolicyforEB.json
```

2. Richiama il comando put-rule e includi i parametri `--name`, `--event-pattern` e `--role-arn`.

Perché occorre apportare questa modifica? È necessario creare un evento con una regola che specifichi come deve essere eseguito il push di un'immagine e un obiettivo che nomini la pipeline da avviare dall'evento.

Il seguente comando di esempio crea una regola denominata `MyECRRepoRule`.

```
aws events put-rule --name "MyECRRepoRule" --event-pattern "{\"detail-type\":[\"ECR
 Image Action\"],\"source\":[\"aws.ecr\"],\"detail\":{\"action-type\":[\"PUSH\"],
\"image-tag\":[\"latest\"],\"repository-name\":[\"eb-test\"],\"result\":[\"SUCCESS
\"]}}}" --role-arn "arn:aws:iam::ACCOUNT_ID:role/Role-for-MyRule"
```

> **Note**
>
> Per visualizzare il modello di eventi completo supportato per gli eventi Amazon ECR, consulta Amazon ECR Events e/o EventBridge Amazon Elastic Container Registry Events.

3. Per aggiungerlo CodePipeline come destinazione, chiama il put-targets comando e includi i seguenti parametri:

   - Il parametro `--rule` viene utilizzato con il `rule_name` che hai creato utilizzando put-rule.
   - Il parametro `--targets` viene utilizzato con l'`Id` elenco della destinazione nell'elenco delle destinazioni e l'ARN della pipeline di destinazione.

   Il comando di esempio seguente specifica che per la regola denominata `MyECRRepoRule`, la destinazione `Id` è composta dal numero uno, per indicare che in un elenco di destinazioni per la regola questa è la destinazione 1. Il comando di esempio specifica anche un esempio `Arn` per la pipeline e l'esempio `RoleArn` per la regola. La pipeline si avvia quando si verifica una modifica nel repository.

   ```
   aws events put-targets --rule MyECRRepoRule --targets
    Id=1,Arn=arn:aws:codepipeline:us-
   west-2:80398EXAMPLE:TestPipeline,RoleArn=arn:aws:iam::80398EXAMPLE:role/Role-for-
   MyRule
   ```

Crea una EventBridge regola per una fonte Amazon ECR (AWS CloudFormation modello)

AWS CloudFormation Per creare una regola, usa lo snippet di modello come mostrato qui.

Per aggiornare il AWS CloudFormation modello di pipeline e creare una regola EventBridge

1. Nel modello, sotto`Resources`, utilizza la `AWS::IAM::Role` AWS CloudFormation risorsa per configurare il ruolo IAM che consente all'evento di avviare la pipeline. Questa voce crea un ruolo che utilizza due policy:

   - La prima policy consente di assumere quel ruolo.
   - La seconda policy fornisce le autorizzazioni per avviare la pipeline.

   Perché occorre apportare questa modifica? Devi creare un ruolo che possa essere assunto da EventBridge per avviare un'esecuzione nella nostra pipeline.

   YAML

```
  EventRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          -
            Effect: Allow
            Principal:
              Service:
                - events.amazonaws.com
            Action: sts:AssumeRole
      Path: /
      Policies:
        -
          PolicyName: eb-pipeline-execution
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              -
                Effect: Allow
                Action: codepipeline:StartPipelineExecution
                Resource: !Sub arn:aws:codepipeline:${AWS::Region}:
${AWS::AccountId}:${AppPipeline}
```

JSON

```
{
    "EventRole": {
        "Type": "AWS::IAM::Role",
        "Properties": {
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {
                            "Service": [
                                "events.amazonaws.com"
                            ]
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "Path": "/",
            "Policies": [
                {
                    "PolicyName": "eb-pipeline-execution",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": "codepipeline:StartPipelineExecution",
                                "Resource": {
                                    "Fn::Sub": "arn:aws:codepipeline:
${AWS::Region}:${AWS::AccountId}:${AppPipeline}"
                                }
                            }
                        ]
                    }
                }
            ]
        }
    }
}
```

```
...
```

2. Nel modello, sotto`Resources`, utilizza la `AWS::Events::Rule` AWS CloudFormation risorsa per aggiungere una EventBridge regola per il codice sorgente Amazon ECR. Questo modello di eventi crea un evento che monitora i commit nel tuo repository. Quando EventBridge rileva una modifica dello stato del repository, la regola viene `StartPipelineExecution` richiamata sulla pipeline di destinazione.

   Perché sto apportando questa modifica? È necessario creare un evento con una regola che specifichi come deve essere eseguito il push di un'immagine e un target che nomini la pipeline da avviare dall'evento.

   Questo frammento utilizza un'immagine denominata `eb-test` con un tag `latest`.

   YAML

```
EventRule:
  Type: 'AWS::Events::Rule'
  Properties:
    EventPattern:
      detail:
        action-type: [PUSH]
        image-tag: [latest]
        repository-name: [eb-test]
        result: [SUCCESS]
      detail-type: [ECR Image Action]
      source: [aws.ecr]
    Targets:
      - Arn: !Sub arn:aws:codepipeline:${AWS::Region}:${AWS::AccountId}:
${AppPipeline}
        RoleArn: !GetAtt
          - EventRole
          - Arn
        Id: codepipeline-AppPipeline
```

   JSON

```
{
    "EventRule": {
        "Type": "AWS::Events::Rule",
        "Properties": {
```

```
            "EventPattern": {
                "detail": {
                    "action-type": [
                        "PUSH"
                    ],
                    "image-tag": [
                        "latest"
                    ],
                    "repository-name": [
                        "eb-test"
                    ],
                    "result": [
                        "SUCCESS"
                    ]
                },
                "detail-type": [
                    "ECR Image Action"
                ],
                "source": [
                    "aws.ecr"
                ]
            },
            "Targets": [
                {
                    "Arn": {
                        "Fn::Sub": "arn:aws:codepipeline:${AWS::Region}:
${AWS::AccountId}:${AppPipeline}"
                    },
                    "RoleArn": {
                        "Fn::GetAtt": [
                            "EventRole",
                            "Arn"
                        ]
                    },
                    "Id": "codepipeline-AppPipeline"
                }
            ]
        }
    }
},
```

> **Note**
>
> Per visualizzare il modello di eventi completo supportato per gli eventi Amazon ECR, consulta Amazon ECR Events e/o EventBridge Amazon Elastic Container Registry Events.

3. Salva il modello aggiornato nel computer locale e quindi apri la console AWS CloudFormation .
4. Seleziona lo stack e scegli Create Change Set for Current Stack (Crea set di modifiche per lo stack corrente).
5. Carica il modello e quindi visualizza le modifiche elencate in AWS CloudFormation. Queste sono le modifiche da apportare allo stack. Le nuove risorse dovrebbero essere visibili nell'elenco.
6. Scegliere Execute (Esegui).

Azioni di origine di Amazon S3 e con EventBridge AWS CloudTrail

Per aggiungere un'azione sorgente Amazon S3 CodePipeline, puoi scegliere tra:

- Utilizza la CodePipeline console Create pipeline wizard (Creazione di una pipeline (console)) o la pagina Edit action per scegliere l'opzione del provider S3. La console crea una EventBridge regola e un CloudTrail percorso che avviano la pipeline quando cambia la fonte.
- Usa AWS CLI per aggiungere la configurazione dell'`S3`azione e creare risorse aggiuntive come segue:
  - Usa la configurazione dell'azione di `S3` esempio in Azione all'origine di Amazon S3 per creare l'azione come mostrato inCreazione di una pipeline (CLI).
  - Per impostazione predefinita, il metodo di rilevamento delle modifiche avvia la pipeline interrogando la fonte. È necessario disabilitare i controlli periodici e creare manualmente la regola di rilevamento delle modifiche e la traccia. Utilizzate uno dei seguenti metodi: Crea una EventBridge regola per un codice sorgente Amazon S3 (console)Crea una EventBridge regola per un codice sorgente Amazon S3 (CLI), oCrea una EventBridge regola per un codice sorgente Amazon S3 (modello)AWS CloudFormation.

AWS CloudTrail è un servizio che registra e filtra gli eventi sul tuo bucket di origine Amazon S3. Il trail invia le modifiche alla fonte filtrate alla regola. EventBridge La EventBridge regola rileva la modifica all'origine e quindi avvia la pipeline.

Requisiti:

- Se non stai creando un trail, usa un AWS CloudTrail trail esistente per registrare gli eventi nel tuo bucket di origine Amazon S3 e inviare eventi filtrati alla regola. EventBridge
- Crea o usa un bucket S3 esistente in cui AWS CloudTrail archiviare i relativi file di registro. AWS CloudTrail deve disporre delle autorizzazioni necessarie per inviare i file di log a un bucket Amazon S3. Il bucket non può essere configurato come bucket con pagamento a carico del richiedente. Quando crei un bucket Amazon S3 come parte della creazione o dell'aggiornamento di un trail nella console, AWS CloudTrail assegna automaticamente le autorizzazioni necessarie a un bucket. Per ulteriori informazioni, consulta Amazon S3 Bucket Policy per. CloudTrail

Crea una EventBridge regola per un codice sorgente Amazon S3 (console)

Prima di configurare una regola in EventBridge, devi creare un AWS CloudTrail percorso. Per ulteriori informazioni, consulta Creazione di un trail nella console.

> **Important**
>
> Se usi la console per creare o modificare la pipeline, la EventBridge regola e il AWS CloudTrail percorso vengono creati automaticamente.

Per creare un trail

1. Apri la AWS CloudTrail console.
2. Nel riquadro di navigazione selezionare Trails (Percorso).
3. Scegliere Create trail (Creare trail). In Trail name (Nome trail), immetti un nome per il trail.
4. In Storage location (Percorso di storage), creare o specificare il bucket da utilizzare per archiviare i file di log. Per impostazione predefinita, i bucket e gli oggetti Amazon S3 sono privati. Solo il proprietario della risorsa (l' AWS account che ha creato il bucket) può accedere al bucket e ai suoi oggetti. Il bucket deve disporre di una politica delle risorse che consenta AWS CloudTrail le autorizzazioni di accesso agli oggetti nel bucket.
5. In Trail log bucket and folder, specifica un bucket Amazon S3 e il prefisso dell'oggetto (nome della cartella) per registrare gli eventi relativi ai dati per tutti gli oggetti nella cartella. Per ogni trail puoi aggiungere fino a 250 oggetti Amazon S3. Completa le informazioni sulla chiave di crittografia richieste e scegli Avanti.
6. Per Tipo di evento, scegli Eventi di gestione.

7. Per gli eventi di gestione, scegli Scrivi. Il percorso registra l'attività dell'API a livello di oggetto di Amazon S3 (ad esempio `GetObject` e`PutObject`) sul bucket e sul prefisso specificati.
8. Scegliere Write (Scrivi).
9. Se sei soddisfatto del percorso, scegli Crea percorso.

Per creare una EventBridge regola che abbia come target la tua pipeline con una fonte Amazon S3

1. Apri la EventBridge console Amazon all'indirizzo https://console.aws.amazon.com/events/.
2. Nel pannello di navigazione, scegli Regole. Lascia selezionato il bus predefinito o scegli un bus per eventi. Scegli Crea regola.
3. In Nome, inserisci un nome per la regola.
4. In Tipo di regola, scegli Regola con un modello di evento. Seleziona Successivo.
5. In Origine evento, scegli AWS eventi o eventi EventBridge partner.
6. In Tipo di evento di esempio, scegli AWS eventi.
7. In Eventi di esempio, digita S3 come parola chiave su cui filtrare. Scegli la chiamata AWS API tramite CloudTrail.
8. In Metodo di creazione, scegli Customer pattern (editor JSON).

   Incolla lo schema di eventi fornito di seguito. Assicurati di aggiungere il nome del bucket e la chiave dell'oggetto S3 (o nome chiave) che identifica in modo univoco l'oggetto nel bucket come. `requestParameters` In questo esempio, viene creata una regola per un bucket denominato `my-bucket` e una chiave oggetto di. `my-files.zip` Quando utilizzi la finestra Edit (Modifica) per specificare le risorse, la regola viene aggiornata per l'utilizzo di un modello eventi personalizzato.

   Di seguito è riportato un esempio di modello di eventi da copiare e incollare:

```
{
    "source": [
        "aws.s3"
    ],
    "detail-type": [
        "AWS API Call via CloudTrail"
    ],
    "detail": {
        "eventSource": [
            "s3.amazonaws.com"
```

```
        ],
        "eventName": [
            "CopyObject",
            "CompleteMultipartUpload",
            "PutObject"
        ],
        "requestParameters": {
            "bucketName": [
                "my-bucket"
            ],
            "key": [
                "my-files.zip"
            ]
        }
    }
}
```

9. Seleziona Successivo.
10. In Tipi di Target, scegli AWS service.
11. In Seleziona un obiettivo, scegli CodePipeline. In Pipeline ARN, immettete l'ARN della pipeline per la pipeline da avviare in base a questa regola.

> **Note**
>
> Per ottenere l'ARN della pipeline, esegui il comando get-pipeline. L'ARN della pipeline viene visualizzato nell'output. Il formato è il seguente:
> *arn:aws:codepipeline: regione: account: nome-pipeline*
> ARN della pipeline di esempio:
> arn:aws:codepipeline:us-east- 2:80398 ESEMPIO: MyFirstPipeline

12. Per creare o specificare un ruolo di servizio IAM che conceda le EventBridge autorizzazioni per richiamare il target associato alla regola (in questo caso, l'obiettivo è): EventBridge CodePipeline
    - Scegli Crea un nuovo ruolo per questa risorsa specifica per creare un ruolo di servizio che ti dia EventBridge le autorizzazioni per avviare le esecuzioni della pipeline.
    - Scegli Usa il ruolo esistente per inserire un ruolo di servizio che ti dia EventBridge le autorizzazioni per avviare le esecuzioni della pipeline.
13. Seleziona Successivo.
14. Nella pagina Tag, scegli Avanti.

15. Nella pagina Rivedi e crea, esamina la configurazione della regola. Se la regola ti soddisfa, scegli Create rule (Crea regola).

Crea una EventBridge regola per un codice sorgente Amazon S3 (CLI)

Per creare un AWS CloudTrail percorso e abilitare la registrazione

Per utilizzare il AWS CLI per creare una traccia, chiamate il create-trail comando, specificando:

- Il nome del trail.
- Il bucket nel quale hai già applicato le policy del bucket per AWS CloudTrail.

Per ulteriori informazioni, vedere Creazione di un percorso con l'interfaccia a riga di AWS comando.

1. Chiama il comando create-trail e includi i parametri `--name` e `--s3-bucket-name`.

   Perché occorre apportare questa modifica? Questo crea il CloudTrail percorso richiesto per il tuo bucket di origine S3.

   Il comando seguente utilizza `--name` e `--s3-bucket-name` per creare un trail denominato `my-trail` e un bucket denominato `myBucket`.

   ```
   aws cloudtrail create-trail --name my-trail --s3-bucket-name myBucket
   ```

2. Chiama il comando start-logging e includi il parametro `--name`.

   Perché sto apportando questa modifica? Questo comando avvia la CloudTrail registrazione per il bucket di origine e invia gli eventi a. EventBridge

   Esempio:

   Il comando seguente utilizza `--name` per avviare la registrazione su un trail denominato `my-trail`.

   ```
   aws cloudtrail start-logging --name my-trail
   ```

3. Chiama il comando put-event-selectors e includi i parametri `--trail-name` e `--event-selectors`. Usa i selettori di eventi per specificare che desideri che il tuo trail registri gli eventi di dati per il tuo bucket di origine e invii gli eventi alla regola. EventBridge

Perché sto apportando questa modifica? Questo comando filtra gli eventi.

Esempio:

Il comando seguente utilizza `--trail-name` e `--event-selectors` per specificare gli eventi dati per un bucket e un prefisso di origine denominati `myBucket/myFolder`.

```
aws cloudtrail put-event-selectors --trail-name my-trail --event-selectors
 '[{ "ReadWriteType": "WriteOnly", "IncludeManagementEvents":false,
 "DataResources": [{ "Type": "AWS::S3::Object", "Values": ["arn:aws:s3:::myBucket/
myFolder/file.zip"] }] }]'
```

Per creare una EventBridge regola con Amazon S3 come origine dell'evento e CodePipeline come destinazione e applicare la politica delle autorizzazioni

1. Concedi le autorizzazioni EventBridge da utilizzare per CodePipeline richiamare la regola. Per ulteriori informazioni, consulta Utilizzo delle politiche basate sulle risorse per Amazon. EventBridge

   a. Usa l'esempio seguente per creare la politica di fiducia che consente di EventBridge assumere il ruolo di servizio. Denominalo `trustpolicyforEB.json`.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Principal": {
                   "Service": "events.amazonaws.com"
               },
               "Action": "sts:AssumeRole"
           }
       ]
   }
   ```

   b. Utilizza il seguente comando per creare il ruolo `Role-for-MyRule` e collegare la policy di attendibilità.

      Perché occorre apportare questa modifica? L'aggiunta di questa politica di fiducia al ruolo crea le autorizzazioni per EventBridge.

```
aws iam create-role --role-name Role-for-MyRule --assume-role-policy-document
 file://trustpolicyforEB.json
```

c. Crea il JSON della policy delle autorizzazioni, come mostrato qui per la pipeline denominata `MyFirstPipeline`. Denomina la policy delle autorizzazioni `permissionspolicyforEB.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:StartPipelineExecution"
            ],
            "Resource": [
                "arn:aws:codepipeline:us-west-2:80398EXAMPLE:MyFirstPipeline"
            ]
        }
    ]
}
```

d. Utilizza il comando seguente per collegare la nuova policy delle autorizzazioni `CodePipeline-Permissions-Policy-for-EB` al ruolo `Role-for-MyRule` che hai creato.

```
aws iam put-role-policy --role-name Role-for-MyRule --policy-name CodePipeline-
Permissions-Policy-For-EB --policy-document file://permissionspolicyforEB.json
```

2. Richiama il comando put-rule e includi i parametri `--name`, `--event-pattern` e `--role-arn`.

   Il seguente comando di esempio crea una regola denominata `MyS3SourceRule`.

```
aws events put-rule --name "MyS3SourceRule" --event-pattern "{\"source\":
[\"aws.s3\"],\"detail-type\":[\"AWS API Call via CloudTrail\"],\"detail\":
{\"eventSource\":[\"s3.amazonaws.com\"],\"eventName\":[\"CopyObject\",\"PutObject
\",\"CompleteMultipartUpload\"],\"requestParameters\":{\"bucketName\":[\"my-bucket
\"],\"key\":[\"my-key\"]}}}
 --role-arn "arn:aws:iam::ACCOUNT_ID:role/Role-for-MyRule"
```

3. Per aggiungere CodePipeline come destinazione, chiamate il put-targets comando e includete i `--targets` parametri `--rule` and.

   Il comando seguente specifica che per la regola denominata `MyS3SourceRule`, la destinazione `Id` è composta dal numero uno, per indicare che in un elenco di destinazioni per la regola questa è la destinazione 1. Il comando specifica anche un esempio di ARN per la pipeline. La pipeline si avvia quando si verifica una modifica nel repository.

   ```
   aws events put-targets --rule MyS3SourceRule --targets
    Id=1,Arn=arn:aws:codepipeline:us-west-2:80398EXAMPLE:TestPipeline
   ```

Per modificare il parametro della PollForSourceChanges pipeline

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

1. Esegui il comando get-pipeline per copiare la struttura della pipeline in un file JSON. Ad esempio, per una pipeline denominata `MyFirstPipeline`, esegui il seguente comando:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   ```

   Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.

2. Apri il file JSON in qualsiasi editor di testo normale e modifica la fase di origine modificando il parametro `PollForSourceChanges` per un bucket denominato `storage-bucketfalse` come mostrato nell'esempio seguente.

   Perché occorre apportare questa modifica? L'impostazione del parametro su `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

```
"configuration": {
    "S3Bucket": "storage-bucket",
    "PollForSourceChanges": "false",
    "S3ObjectKey": "index.zip"
},
```

3. Se stai utilizzando la struttura della pipeline recuperata tramite il comando get-pipeline, devi rimuovere le righe `metadata` dal file JSON. In caso contrario, il comando update-pipeline non è in grado di utilizzarlo. Rimuovi le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`.

   Ad esempio, rimuovere dalla struttura le seguenti righe:

   ```
   "metadata": {
       "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
       "created": "date",
       "updated": "date"
   },
   ```

   Salvare il file.

4. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline:

   Important

   Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

   ```
   aws codepipeline update-pipeline --cli-input-json file://pipeline.json
   ```

   Questo comando restituisce l'intera struttura della pipeline modificata.

   Note

   Il comando update-pipeline arresta la pipeline. Se è in corso di elaborazione una versione durante l'esecuzione del comando update-pipeline, tale elaborazione viene arrestata. Per elaborare tale versione utilizzando la pipeline aggiornata, devi avviare

manualmente la pipeline. Utilizza il comando start-pipeline-execution per avviare manualmente la pipeline.

Crea una EventBridge regola per un codice sorgente Amazon S3 (modello)AWS CloudFormation

Per utilizzarla AWS CloudFormation per creare una regola, aggiorna il modello come mostrato qui.

Per creare una EventBridge regola con Amazon S3 come origine dell'evento e CodePipeline come destinazione e applicare la politica delle autorizzazioni

1. Nel modello, sotto`Resources`, utilizza la `AWS::IAM::Role` AWS CloudFormation risorsa per configurare il ruolo IAM che consente all'evento di avviare la pipeline. Questa voce crea un ruolo che utilizza due policy:

   - La prima policy consente di assumere quel ruolo.
   - La seconda policy fornisce le autorizzazioni per avviare la pipeline.

   Perché occorre apportare questa modifica? L'aggiunta di `AWS::IAM::Role` risorse consente AWS CloudFormation di creare autorizzazioni per. EventBridge Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

```
EventRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: 2012-10-17
      Statement:
        -
          Effect: Allow
          Principal:
            Service:
              - events.amazonaws.com
          Action: sts:AssumeRole
    Path: /
    Policies:
      -
```

```
          PolicyName: eb-pipeline-execution
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              -
                Effect: Allow
                Action: codepipeline:StartPipelineExecution
                Resource: !Join [ '', [ 'arn:aws:codepipeline:', !Ref
 'AWS::Region', ':', !Ref 'AWS::AccountId', ':', !Ref AppPipeline ] ]


...
```

JSON

```
 "EventRole": {
   "Type": "AWS::IAM::Role",
   "Properties": {
     "AssumeRolePolicyDocument": {
       "Version": "2012-10-17",
       "Statement": [
         {
           "Effect": "Allow",
           "Principal": {
             "Service": [
               "events.amazonaws.com"
             ]
           },
           "Action": "sts:AssumeRole"
         }
       ]
     },
     "Path": "/",
     "Policies": [
       {
         "PolicyName": "eb-pipeline-execution",
         "PolicyDocument": {
           "Version": "2012-10-17",
           "Statement": [
             {
               "Effect": "Allow",
               "Action": "codepipeline:StartPipelineExecution",
               "Resource": {
```

```
                "Fn::Join": [
                  "",
                  [
                    "arn:aws:codepipeline:",
                    {
                      "Ref": "AWS::Region"
                    },
                    ":",
                    {
                      "Ref": "AWS::AccountId"
                    },
                    ":",
                    {
                      "Ref": "AppPipeline"
                    }
                  ]
                ]

...
```

2. Usa la `AWS::Events::Rule` AWS CloudFormation risorsa per aggiungere una EventBridge regola. Questo modello di eventi crea un evento che monitora `CopyObject` `PutObject` e `CompleteMultipartUpload` sul tuo bucket di origine Amazon S3. Inoltre, include una destinazione della pipeline. Quando si verifica `CopyObject`, `PutObject` o `CompleteMultipartUpload`, questa regola richiama `StartPipelineExecution` sulla pipeline di destinazione.

   Perché occorre apportare questa modifica? L'aggiunta della `AWS::Events::Rule` risorsa consente di AWS CloudFormation creare l'evento. Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

```
  EventRule:
    Type: AWS::Events::Rule
    Properties:
      EventPattern:
        source:
          - aws.s3
        detail-type:
          - 'AWS API Call via CloudTrail'
        detail:
```

```
          eventSource:
            - s3.amazonaws.com
          eventName:
            - CopyObject
            - PutObject
            - CompleteMultipartUpload
          requestParameters:
            bucketName:
              - !Ref SourceBucket
            key:
              - !Ref SourceObjectKey
      Targets:
        -
          Arn:
            !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region', ':', !Ref
 'AWS::AccountId', ':', !Ref AppPipeline ] ]
          RoleArn: !GetAtt EventRole.Arn
          Id: codepipeline-AppPipeline


...
```

JSON

```
 "EventRule": {
   "Type": "AWS::Events::Rule",
   "Properties": {
     "EventPattern": {
       "source": [
         "aws.s3"
       ],
       "detail-type": [
         "AWS API Call via CloudTrail"
       ],
       "detail": {
         "eventSource": [
           "s3.amazonaws.com"
         ],
         "eventName": [
           "CopyObject",
           "PutObject",
           "CompleteMultipartUpload"
```

```
          ],
          "requestParameters": {
            "bucketName": [
              {
                "Ref": "SourceBucket"
              }
            ],
            "key": [
              {
                "Ref": "SourceObjectKey"
              }
            ]
          }
        }
      },
      "Targets": [
        {
          "Arn": {
            "Fn::Join": [
              "",
              [
                "arn:aws:codepipeline:",
                {
                  "Ref": "AWS::Region"
                },
                ":",
                {
                  "Ref": "AWS::AccountId"
                },
                ":",
                {
                  "Ref": "AppPipeline"
                }
              ]
            ]
          },
          "RoleArn": {
            "Fn::GetAtt": [
              "EventRole",
              "Arn"
            ]
          },
          "Id": "codepipeline-AppPipeline"
        }
```

```
      ]
    }
  }
},

...
```

3. Aggiungere questo snippet di codice al primo modello per consentire la funzionalità tra stack:

   YAML

   ```
   Outputs:
     SourceBucketARN:
       Description: "S3 bucket ARN that Cloudtrail will use"
       Value: !GetAtt SourceBucket.Arn
       Export:
         Name: SourceBucketARN
   ```

   JSON

   ```
     "Outputs" : {
       "SourceBucketARN" : {
         "Description" : "S3 bucket ARN that Cloudtrail will use",
         "Value" : { "Fn::GetAtt": ["SourceBucket", "Arn"] },
         "Export" : {
           "Name" : "SourceBucketARN"
         }
       }

   ...
   ```

4. Salva il modello aggiornato sul computer locale e apri la AWS CloudFormation console.
5. Seleziona lo stack e scegli Create Change Set for Current Stack (Crea set di modifiche per lo stack corrente).
6. Caricare il modello aggiornato e quindi visualizzare le modifiche elencate in AWS CloudFormation. Queste sono le modifiche che verranno apportate allo stack. Le nuove risorse dovrebbero essere visibili nell'elenco.
7. Scegliere Execute (Esegui).

Per modificare i parametri della PollForSourceChanges pipeline

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

- Nel modello, modifica `PollForSourceChanges` in `false`. Se non hai incluso `PollForSourceChanges` nella definizione della pipeline, aggiungilo e impostalo su `false`.

  Perché occorre apportare questa modifica? La modifica di `PollForSourceChanges` in `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

  YAML

```
Name: Source
Actions:
  -
    Name: SourceAction
    ActionTypeId:
      Category: Source
      Owner: AWS
      Version: 1
      Provider: S3
    OutputArtifacts:
      - Name: SourceOutput
    Configuration:
      S3Bucket: !Ref SourceBucket
      S3ObjectKey: !Ref SourceObjectKey
      PollForSourceChanges: false
    RunOrder: 1
```

JSON

```
{
   "Name": "SourceAction",
   "ActionTypeId": {
     "Category": "Source",
     "Owner": "AWS",
     "Version": 1,
     "Provider": "S3"
   },
   "OutputArtifacts": [
     {
       "Name": "SourceOutput"
     }
   ],
   "Configuration": {
     "S3Bucket": {
       "Ref": "SourceBucket"
     },
     "S3ObjectKey": {
       "Ref": "SourceObjectKey"
     },
     "PollForSourceChanges": false
   },
   "RunOrder": 1
 }
```

Per creare un secondo modello per le risorse della tua pipeline Amazon S3 CloudTrail

- In un modello separato, sotto`Resources`, utilizza le `AWS::CloudTrail::Trail` AWS CloudFormation risorse `AWS::S3::BucketAWS::S3::BucketPolicy`, e per fornire una definizione e un percorso semplici per il bucket. CloudTrail

  Perché sto apportando questa modifica? Dato l'attuale limite di cinque percorsi per account, il CloudTrail percorso deve essere creato e gestito separatamente. (Vedi Limiti in AWS CloudTrail.) Tuttavia, puoi includere molti bucket Amazon S3 in un singolo trail, in modo da poter creare il trail una sola volta e poi aggiungere bucket Amazon S3 per altre pipeline, se necessario. Incolla quanto segue nel secondo file di modello di esempio.

YAML

```
###############################################################################
# Prerequisites:
#   - S3 SourceBucket and SourceObjectKey must exist
###############################################################################

Parameters:
  SourceObjectKey:
    Description: 'S3 source artifact'
    Type: String
    Default: SampleApp_Linux.zip

Resources:
  AWSCloudTrailBucketPolicy:
    Type: AWS::S3::BucketPolicy
    Properties:
      Bucket: !Ref AWSCloudTrailBucket
      PolicyDocument:
        Version: 2012-10-17
        Statement:
          -
            Sid: AWSCloudTrailAclCheck
            Effect: Allow
            Principal:
              Service:
                - cloudtrail.amazonaws.com
            Action: s3:GetBucketAcl
            Resource: !GetAtt AWSCloudTrailBucket.Arn
          -
            Sid: AWSCloudTrailWrite
            Effect: Allow
            Principal:
              Service:
                - cloudtrail.amazonaws.com
            Action: s3:PutObject
            Resource: !Join [ '', [ !GetAtt AWSCloudTrailBucket.Arn, '/
AWSLogs/', !Ref 'AWS::AccountId', '/*' ] ]
            Condition:
              StringEquals:
                s3:x-amz-acl: bucket-owner-full-control
  AWSCloudTrailBucket:
    Type: AWS::S3::Bucket
```

```
    DeletionPolicy: Retain
  AwsCloudTrail:
    DependsOn:
      - AWSCloudTrailBucketPolicy
    Type: AWS::CloudTrail::Trail
    Properties:
      S3BucketName: !Ref AWSCloudTrailBucket
      EventSelectors:
        -
          DataResources:
            -
              Type: AWS::S3::Object
              Values:
                - !Join [ '', [ !ImportValue SourceBucketARN, '/', !Ref
 SourceObjectKey ] ]
          ReadWriteType: WriteOnly
          IncludeManagementEvents: false
      IncludeGlobalServiceEvents: true
      IsLogging: true
      IsMultiRegionTrail: true


...
```

JSON

```
{
  "Parameters": {
    "SourceObjectKey": {
      "Description": "S3 source artifact",
      "Type": "String",
      "Default": "SampleApp_Linux.zip"
    }
  },
  "Resources": {
    "AWSCloudTrailBucket": {
      "Type": "AWS::S3::Bucket",
        "DeletionPolicy": "Retain"
    },
    "AWSCloudTrailBucketPolicy": {
      "Type": "AWS::S3::BucketPolicy",
      "Properties": {
        "Bucket": {
```

```
          "Ref": "AWSCloudTrailBucket"
        },
        "PolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "AWSCloudTrailAclCheck",
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "cloudtrail.amazonaws.com"
                ]
              },
              "Action": "s3:GetBucketAcl",
              "Resource": {
                "Fn::GetAtt": [
                  "AWSCloudTrailBucket",
                  "Arn"
                ]
              }
            },
            {
              "Sid": "AWSCloudTrailWrite",
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "cloudtrail.amazonaws.com"
                ]
              },
              "Action": "s3:PutObject",
              "Resource": {
                "Fn::Join": [
                  "",
                  [
                    {
                      "Fn::GetAtt": [
                        "AWSCloudTrailBucket",
                        "Arn"
                      ]
                    },
                    "/AWSLogs/",
                    {
                      "Ref": "AWS::AccountId"
                    },
```

```
                "/*"
              ]
            ]
          },
          "Condition": {
            "StringEquals": {
              "s3:x-amz-acl": "bucket-owner-full-control"
            }
          }
        }
      ]
    }
  }
},
"AwsCloudTrail": {
  "DependsOn": [
    "AWSCloudTrailBucketPolicy"
  ],
  "Type": "AWS::CloudTrail::Trail",
  "Properties": {
    "S3BucketName": {
      "Ref": "AWSCloudTrailBucket"
    },
    "EventSelectors": [
      {
        "DataResources": [
          {
            "Type": "AWS::S3::Object",
            "Values": [
              {
                "Fn::Join": [
                  "",
                  [
                    {
                      "Fn::ImportValue": "SourceBucketARN"
                    },
                    "/",
                    {
                      "Ref": "SourceObjectKey"
                    }
                  ]
                ]
              }
            ]
```

```
              }
            ],
            "ReadWriteType": "WriteOnly",
            "IncludeManagementEvents": false
          }
        ],
        "IncludeGlobalServiceEvents": true,
        "IsLogging": true,
        "IsMultiRegionTrail": true
      }
    }
  }
}

...
```

Connessioni Bitbucket Cloud

Le connessioni ti consentono di autorizzare e stabilire configurazioni che associano il tuo provider di terze parti alle tue risorse. AWS Per associare il repository di terze parti come fonte per la pipeline, si utilizza una connessione.

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hong Kong), Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Africa (Città del Capo), Medio Oriente (Bahrain), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna), Europa (Zurigo), Israele (Tel Aviv) o AWS GovCloud (Stati Uniti occidentali). Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Per aggiungere un'azione sorgente di Bitbucket Cloud in CodePipeline, puoi scegliere tra:

- Utilizza la procedura guidata di creazione della pipeline della CodePipeline console o la pagina Modifica azione per scegliere l'opzione del provider Bitbucket. Vedi per aggiungere Crea una

connessione a Bitbucket Cloud (console) l'azione. La console ti aiuta a creare una risorsa di connessioni.

> **Note**
>
> È possibile creare connessioni a un repository Bitbucket Cloud. I tipi di provider Bitbucket installati, ad esempio Bitbucket Server, non sono supportati.

- Utilizza la CLI per aggiungere la configurazione dell'azione per l'`CreateSourceConnection`azione con il `Bitbucket` provider come segue:
  - Per creare le tue risorse di connessione, consulta Crea una connessione a Bitbucket Cloud (CLI) Creare una risorsa di connessione con la CLI.
  - Usa l'`CreateSourceConnection`esempio di configurazione dell'azione in CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite per aggiungere la tua azione come mostrato inCreazione di una pipeline (CLI).

> **Note**
>
> Puoi anche creare una connessione utilizzando la console Developer Tools in Impostazioni. Vedi Creare una connessione.

Prima di iniziare:

- Devi aver creato un account con il provider del repository di terze parti, come Bitbucket Cloud.
- Devi aver già creato un repository di codice di terze parti, ad esempio un repository Bitbucket Cloud.

> **Note**
>
> Le connessioni Bitbucket Cloud forniscono l'accesso solo agli archivi di proprietà dell'account Bitbucket Cloud utilizzato per creare la connessione.

Se l'applicazione viene installata in un'area di lavoro Bitbucket Cloud, sono necessarie le autorizzazioni di amministrazione dell'area di lavoro. In caso contrario, l'opzione per installare l'app non verrà visualizzata.

Argomenti

Crea una connessione a Bitbucket Cloud (console)

Segui questi passaggi per utilizzare la CodePipeline console per aggiungere un'azione di connessione per il tuo repository Bitbucket.

Note

È possibile creare connessioni a un repository Bitbucket Cloud. I tipi di provider Bitbucket installati, ad esempio Bitbucket Server, non sono supportati.

Passaggio 1: crea o modifica la tua pipeline

Per creare o modificare la tua pipeline

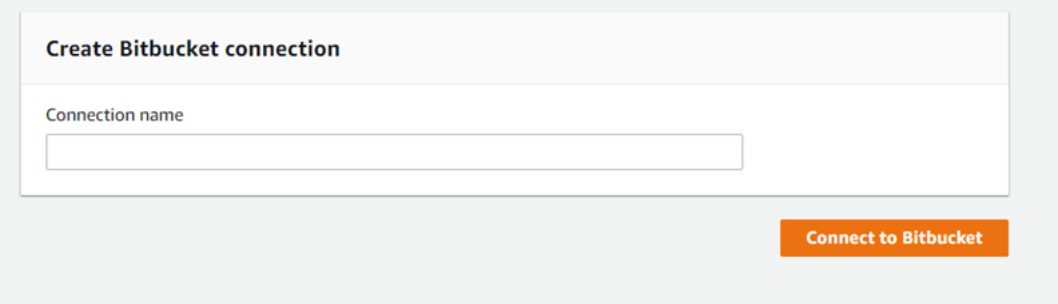
1. Accedi alla CodePipeline console.
2. Scegliere una delle seguenti opzioni.
   - Scegli di creare una pipeline. Segui i passaggi descritti in Crea una pipeline per completare la prima schermata e scegli Avanti. Nella pagina Source, in Source Provider, scegli Bitbucket.
   - Scegli di modificare una pipeline esistente. Scegliete Modifica, quindi scegliete Modifica fase. Scegli di aggiungere o modificare l'azione sorgente. Nella pagina Modifica azione, in Nome azione, inserisci il nome dell'azione. Nel provider Action, scegli Bitbucket.
3. Esegui una di queste operazioni:
   - In Connessione, se non hai già creato una connessione al tuo provider, scegli Connetti a Bitbucket. Procedi al passaggio 2: crea una connessione a Bitbucket.

- In Connessione, se hai già creato una connessione al tuo provider, scegli la connessione. Procedi al passaggio 3: Salva l'azione di origine per la tua connessione.

Passaggio 2: crea una connessione a Bitbucket Cloud

Per creare una connessione a Bitbucket Cloud

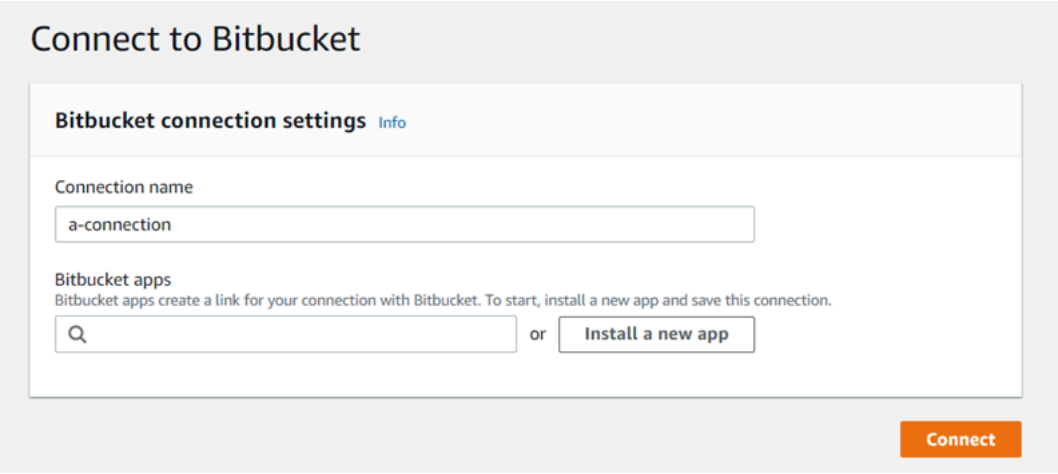
1. Nella pagina delle impostazioni di Connect to Bitbucket, inserisci il nome della connessione e scegli Connetti a Bitbucket.



   Viene visualizzato il campo delle app Bitbucket.

2. In Bitbucket apps (App Bitbucket), selezionare l'installazione di un'app o Install a new app (Installa una nuova app) per crearne una.

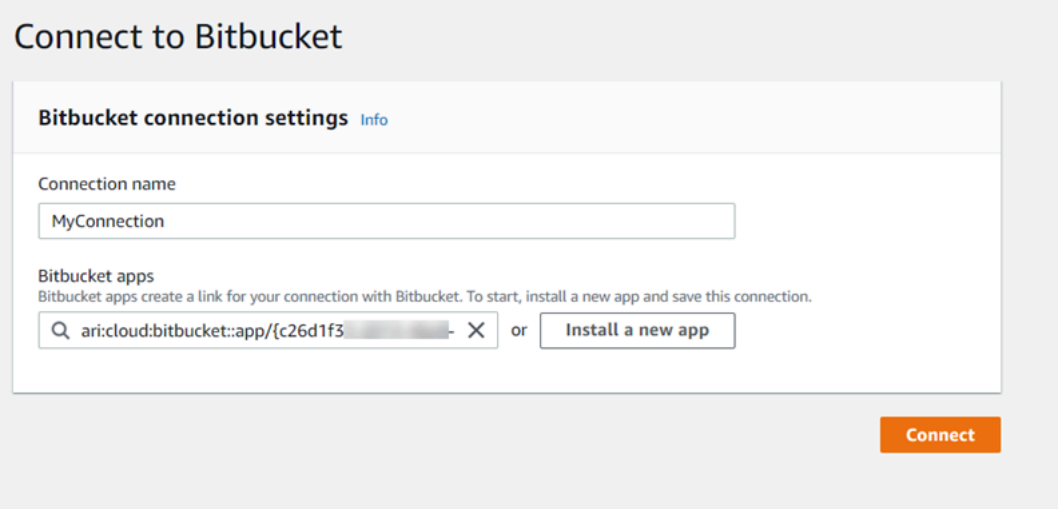
> **Note**
>
> L'app viene installata una sola volta per ogni spazio di lavoro o account Bitbucket Cloud. Se hai già installato l'app Bitbucket, sceglila e vai al passaggio 4.



3. Se viene visualizzata la pagina di accesso per Bitbucket Cloud, accedi con le tue credenziali e scegli di continuare.
4. Nella pagina di installazione dell'app, un messaggio indica che l' AWS CodeStar app sta tentando di connettersi al tuo account Bitbucket.

   Se si sta usando un workspace Bitbucket, modificare l'opzione Authorize for (Autorizza) per il workspace. Verranno visualizzati solo i workspace ai quali è possibile accedere come amministratore.

   Selezionare Grant access (Concedi accesso).
5. In the connection ID for your new installation is displayed. (App Bitbucket), viene visualizzato l'ID di connessione per la nuova installazione. Scegli Connetti. La connessione creata viene visualizzata nell'elenco delle connessioni.



Passaggio 3: salva l'azione sorgente di Bitbucket Cloud

Utilizza questi passaggi nella procedura guidata o nella pagina Modifica azione per salvare l'azione di origine con le informazioni di connessione.

Per completare e salvare l'azione sorgente con la connessione

1. In Repository name (Nome repository), scegliere il nome del repository di terze parti.
2. In Trigger Pipeline puoi aggiungere trigger se la tua azione è un'azione. CodeConnections Per configurare la configurazione dei trigger della pipeline e, facoltativamente, filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice
3. In Output artifact format (Formato artefatto di output), occorre scegliere il formato degli artefatti.

- Per memorizzare gli artefatti di output dall'azione Bitbucket Cloud utilizzando il metodo predefinito, scegli default. CodePipeline L'azione accede ai file dal repository Bitbucket Cloud e archivia gli artefatti in un file ZIP nel pipeline artifact store.
- Per archiviare un file JSON contenente un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git, scegliere Full clone (Clone completo). Questa opzione può essere utilizzata solo per azioni a valle. CodeBuild

  Se scegli questa opzione, dovrai aggiornare le autorizzazioni per il tuo ruolo di CodeBuild Project Service come mostrato in. Aggiungi le autorizzazioni per le connessioni a Bitbucket, Enterprise Server o.com CodeBuild GitClone GitHub GitHub GitLab

4. Scegli Avanti nella procedura guidata o Salva nella pagina Modifica azione.

Crea una connessione a Bitbucket Cloud (CLI)

Puoi usare il AWS Command Line Interface (AWS CLI) per creare una connessione.

> **Note**
>
> È possibile creare connessioni a un repository Bitbucket Cloud. I tipi di provider Bitbucket installati, ad esempio Bitbucket Server, non sono supportati.

Per farlo, utilizzare il comando create-connection.

> **Important**
>
> Per impostazione predefinita, una connessione creata tramite AWS CLI o AWS CloudFormation è in `PENDING` stato. Dopo aver creato una connessione con la CLI o AWS CloudFormation, utilizza la console per modificare la connessione e definirne lo stato. `AVAILABLE`

Per creare una connessione

1. Apri un terminale (Linux, macOS o Unix) o prompt dei comandi (Windows). Usa il AWS CLI per eseguire il create-connection comando, specificando l'`--provider-typee --`

`connection-name` per la tua connessione. In questo esempio, il nome del provider di terze parti è `Bitbucket` e il nome della connessione specificato è `MyConnection`.

```
aws codestar-connections create-connection --provider-type Bitbucket --connection-
name MyConnection
```

In caso di esito positivo, questo comando restituisce informazioni dell'ARN della connessione simili alle seguenti.

```
{
    "ConnectionArn": "arn:aws:codestar-connections:us-west-2:account_id:connection/
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f"
}
```

2. Utilizzare la console per completare la connessione. Per ulteriori informazioni, consulta Aggiornare una connessione in sospeso.
3. Per impostazione predefinita, la pipeline rileva le modifiche al codice inviato al repository delle sorgenti di connessione. Per configurare la configurazione del trigger della pipeline per il rilascio manuale o per i tag Git, esegui una delle seguenti operazioni:

   - Per configurare la configurazione del trigger della pipeline in modo che inizi solo con una versione manuale, aggiungi la seguente riga alla configurazione:

     ```
     "DetectChanges": "false",
     ```

   - Per configurare la configurazione del trigger della pipeline per filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice Ad esempio, quanto segue aggiunge tag Git al livello di pipeline della definizione JSON della pipeline. In questo esempio, `release-v0` e `release-v1` sono i tag Git da includere e `release-v2` il tag Git da escludere.

     ```
     "triggers": [
                 {
                     "providerType": "CodeStarSourceConnection",
                     "gitConfiguration": {
                         "sourceActionName": "Source",
                         "push": [
                             {
                                 "tags": {
                                     "includes": [
     ```

```
                                "release-v0", "release-v1"
                            ],
                            "excludes": [
                                "release-v2"
                            ]
                        }
                    }
                ]
            }
        }
    ]
```

CodeCommit azioni di origine e EventBridge

Per aggiungere un'azione CodeCommit sorgente in CodePipeline, puoi scegliere tra:

- Utilizza la procedura guidata di creazione della pipeline della CodePipeline console (Creazione di una pipeline (console)) o la pagina Modifica azione per scegliere l'opzione del CodeCommitprovider. La console crea una EventBridge regola che avvia la pipeline quando cambia la fonte.
- Usa AWS CLI per aggiungere la configurazione dell'`CodeCommit`azione e creare risorse aggiuntive come segue:
  - Usa la configurazione dell'azione di `CodeCommit` esempio in CodeCommit per creare l'azione come mostrato inCreazione di una pipeline (CLI).
  - Per impostazione predefinita, il metodo di rilevamento delle modifiche avvia la pipeline interrogando la fonte. È necessario disabilitare i controlli periodici e creare manualmente la regola di rilevamento delle modifiche. Utilizzare uno dei seguenti metodi: Crea una EventBridge regola per una CodeCommit fonte (console)Creare una EventBridge regola per un' CodeCommit origine (CLI), oCrea una EventBridge regola per una CodeCommit fonte (AWS CloudFormation modello) .

Argomenti

- Crea una EventBridge regola per una CodeCommit fonte (console)
- Creare una EventBridge regola per un' CodeCommit origine (CLI)
- Crea una EventBridge regola per una CodeCommit fonte (AWS CloudFormation modello)

Crea una EventBridge regola per una CodeCommit fonte (console)

> **Important**
>
> Se usi la console per creare o modificare la tua pipeline, la EventBridge regola viene creata automaticamente.

Per creare una EventBridge regola da utilizzare nelle operazioni CodePipeline

1. Apri la EventBridge console Amazon all'indirizzo https://console.aws.amazon.com/events/.
2. Nel pannello di navigazione, scegli Regole. Lascia selezionato il bus predefinito o scegli un bus per eventi. Scegli Crea regola.
3. In Nome, inserisci un nome per la regola.
4. In Tipo di regola, scegli Regola con un modello di evento. Seleziona Successivo.
5. In Origine evento, scegli AWS eventi o eventi EventBridge partner.
6. In Tipo di evento di esempio, scegli AWS eventi.
7. In Eventi di esempio, digita CodeCommit come parola chiave in base alla quale filtrare. Scegli CodeCommit Repository State Change.
8. In Metodo di creazione, scegli Customer pattern (editor JSON).

   Incolla lo schema di eventi fornito di seguito. Di seguito è riportato un modello di CodeCommit evento di esempio nella finestra Evento per un `MyTestRepo` repository con un ramo denominato`main`:

```
{
  "source": [
    "aws.codecommit"
  ],
  "detail-type": [
    "CodeCommit Repository State Change"
  ],
  "resources": [
    "arn:aws:codecommit:us-west-2:80398EXAMPLE:MyTestRepo"
  ],
  "detail": {
    "referenceType": [
```

```
      "branch"
    ],
    "referenceName": [
      "main"
    ]
  }
}
```

9. In Target, scegliete CodePipeline.
10. Immettete l'ARN della pipeline per la pipeline da avviare in base a questa regola.

> **Note**
>
> Puoi trovare l'ARN della pipeline nell'output dei metadati dopo aver eseguito il comando get-pipeline. Il formato dell'ARN della pipeline è il seguente:
> *arn:aws:codepipeline: regione: account: nome-pipeline*
> ARN della pipeline di esempio:
> `arn:aws:codepipeline:us-east-2:80398EXAMPLE:MyFirstPipeline`

11. Per creare o specificare un ruolo di servizio IAM che conceda EventBridge le autorizzazioni per richiamare la destinazione associata alla regola (in questo caso, la destinazione è): EventBridge CodePipeline
    - Scegli Crea un nuovo ruolo per questa risorsa specifica per creare un ruolo di servizio che ti dia EventBridge le autorizzazioni per avviare le esecuzioni della pipeline.
    - Scegli Usa il ruolo esistente per inserire un ruolo di servizio che ti dia EventBridge le autorizzazioni per avviare le esecuzioni della pipeline.
12. Seleziona Successivo.
13. Nella pagina Tag, scegli Avanti.
14. Nella pagina Rivedi e crea, esamina la configurazione della regola. Se la regola ti soddisfa, scegli Create rule (Crea regola).

Creare una EventBridge regola per un' CodeCommit origine (CLI)

Chiama il comando put-rule, specificando:

- Un nome che identifica in modo univoco la regola che stai creando. Questo nome deve essere univoco in tutte le pipeline che crei e CodePipeline associate al tuo AWS account.

- Il modello eventi per i campi di origine e di dettaglio utilizzati dalla regola. Per ulteriori informazioni, consulta Amazon EventBridge e Event Patterns.

Per creare una EventBridge regola con CodeCommit come origine dell'evento e CodePipeline come destinazione

1. Aggiungi le autorizzazioni EventBridge da utilizzare per CodePipeline richiamare la regola. Per ulteriori informazioni, consulta Utilizzo delle politiche basate sulle risorse per Amazon. EventBridge

   a. Usa l'esempio seguente per creare la politica di fiducia che consente di EventBridge assumere il ruolo di servizio. Denomina la policy di attendibilità `trustpolicyforEB.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "events.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

   b. Utilizza il seguente comando per creare il ruolo `Role-for-MyRule` e collegare la policy di attendibilità.

```
aws iam create-role --role-name Role-for-MyRule --assume-role-policy-document
 file://trustpolicyforEB.json
```

   c. Crea il JSON della policy delle autorizzazioni, come mostrato in questo esempio, per la pipeline denominata `MyFirstPipeline`. Denomina la policy delle autorizzazioni `permissionspolicyforEB.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "codepipeline:StartPipelineExecution"
            ],
            "Resource": [
                "arn:aws:codepipeline:us-west-2:80398EXAMPLE:MyFirstPipeline"
            ]
        }
    ]
}
```

d. Utilizza il comando seguente per collegare la policy delle autorizzazioni `CodePipeline-Permissions-Policy-for-EB` al ruolo `Role-for-MyRule`.

   Perché occorre apportare questa modifica? L'aggiunta di questa politica al ruolo crea le autorizzazioni per EventBridge.

```
aws iam put-role-policy --role-name Role-for-MyRule --policy-name CodePipeline-Permissions-Policy-For-EB --policy-document file://permissionspolicyforEB.json
```

2. Richiama il comando put-rule e includi i parametri `--name`, `--event-pattern` e `--role-arn`.

   Perché occorre apportare questa modifica? Questo comando consente a AWS CloudFormation di creare l'evento.

   Il seguente comando di esempio crea una regola denominata `MyCodeCommitRepoRule`.

```
aws events put-rule --name "MyCodeCommitRepoRule" --event-pattern "{\"source\":[\"aws.codecommit\"],\"detail-type\":[\"CodeCommit Repository State Change\"],\"resources\":[\"repository-ARN\"],\"detail\":{\"referenceType\":[\"branch\"],\"referenceName\":[\"main\"]}}" --role-arn "arn:aws:iam::ACCOUNT_ID:role/Role-for-MyRule"
```

3. Per aggiungerlo CodePipeline come destinazione, chiamate il put-targets comando e includete i seguenti parametri:

   - Il parametro `--rule` viene utilizzato con il `rule_name` che hai creato utilizzando put-rule.
   - Il parametro `--targets` viene utilizzato con l'`Id` elenco della destinazione nell'elenco delle destinazioni e l'ARN della pipeline di destinazione.

   Il comando di esempio seguente specifica che per la regola denominata `MyCodeCommitRepoRule`, la destinazione `Id` è composta dal numero uno, per indicare che

in un elenco di destinazioni per la regola questa è la destinazione 1. Il comando di esempio specifica anche un esempio di ARN per la pipeline. La pipeline si avvia quando si verifica una modifica nel repository.

```
aws events put-targets --rule MyCodeCommitRepoRule --targets
 Id=1,Arn=arn:aws:codepipeline:us-west-2:80398EXAMPLE:TestPipeline
```

Per modificare il parametro della PollForSourceChanges pipeline

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

1. Esegui il comando get-pipeline per copiare la struttura della pipeline in un file JSON. Ad esempio, per una pipeline denominata `MyFirstPipeline`, esegui il seguente comando:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   ```

   Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.

2. Apri il file JSON in qualsiasi editor di testo normale e modifica la fase di origine modificando il parametro `PollForSourceChanges` su `false`, come illustrato in questo esempio.

   Perché occorre apportare questa modifica? La modifica di questo parametro in `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

   ```
   "configuration": {
       "PollForSourceChanges": "false",
       "BranchName": "main",
       "RepositoryName": "MyTestRepo"
   ```

```
},
```

3. Se stai utilizzando la struttura della pipeline recuperata tramite il comando get-pipeline, rimuovi le righe `metadata` dal file JSON. In caso contrario, il comando update-pipeline non è in grado di utilizzarlo. Rimuovi le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`.

   Ad esempio, rimuovere dalla struttura le seguenti righe:

```
"metadata": {
    "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
    "created": "date",
    "updated": "date"
},
```

   Salvare il file.

4. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline:

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

```
aws codepipeline update-pipeline --cli-input-json file://pipeline.json
```

Questo comando restituisce l'intera struttura della pipeline modificata.

> **Note**
>
> Il comando update-pipeline arresta la pipeline. Se è in corso di elaborazione una versione durante l'esecuzione del comando update-pipeline, tale elaborazione viene arrestata. Per elaborare tale versione utilizzando la pipeline aggiornata, devi avviare manualmente la pipeline. Utilizza il comando **`start-pipeline-execution`** per avviare manualmente la pipeline.

Crea una EventBridge regola per una CodeCommit fonte (AWS CloudFormation modello)

Per AWS CloudFormation utilizzarla per creare una regola, aggiorna il modello come mostrato qui.

Per aggiornare il AWS CloudFormation modello di pipeline e creare una regola EventBridge

1. Nel modello, sotto`Resources`, utilizza la `AWS::IAM::Role` AWS CloudFormation risorsa per configurare il ruolo IAM che consente all'evento di avviare la pipeline. Questa voce crea un ruolo che utilizza due policy:

   - La prima policy consente di assumere quel ruolo.
   - La seconda policy fornisce le autorizzazioni per avviare la pipeline.

   Perché occorre apportare questa modifica? L'aggiunta della `AWS::IAM::Role` risorsa consente di AWS CloudFormation creare autorizzazioni per. EventBridge Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

```
EventRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: 2012-10-17
      Statement:
        -
          Effect: Allow
          Principal:
            Service:
              - events.amazonaws.com
          Action: sts:AssumeRole
    Path: /
    Policies:
      -
        PolicyName: eb-pipeline-execution
        PolicyDocument:
          Version: 2012-10-17
          Statement:
            -
```

```
                Effect: Allow
                Action: codepipeline:StartPipelineExecution
                Resource: !Join [ '', [ 'arn:aws:codepipeline:', !Ref
 'AWS::Region', ':', !Ref 'AWS::AccountId', ':', !Ref AppPipeline ] ]
```

JSON

```
"EventRole": {
  "Type": "AWS::IAM::Role",
  "Properties": {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
            "Service": [
              "events.amazonaws.com"
            ]
          },
          "Action": "sts:AssumeRole"
        }
      ]
    },
    "Path": "/",
    "Policies": [
      {
        "PolicyName": "eb-pipeline-execution",
        "PolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Action": "codepipeline:StartPipelineExecution",
              "Resource": {
                "Fn::Join": [
                  "",
                  [
                    "arn:aws:codepipeline:",
                    {
                      "Ref": "AWS::Region"
                    },
                    ":",
```

```
                    {
                      "Ref": "AWS::AccountId"
                    },
                    ":",
                    {
                      "Ref": "AppPipeline"
                    }
                  ]

...
```

2. Nel modello, sotto`Resources`, usa la `AWS::Events::Rule` AWS CloudFormation risorsa per aggiungere una EventBridge regola. Questo modello di eventi crea un evento che monitora le modifiche push al tuo repository. Quando EventBridge rileva una modifica dello stato del repository, la regola viene `StartPipelineExecution` richiamata sulla pipeline di destinazione.

   Perché sto apportando questa modifica? L'aggiunta della `AWS::Events::Rule` risorsa AWS CloudFormation consente di creare l'evento. Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

```
  EventRule:
    Type: AWS::Events::Rule
    Properties:
      EventPattern:
        source:
          - aws.codecommit
        detail-type:
          - 'CodeCommit Repository State Change'
        resources:
          - !Join [ '', [ 'arn:aws:codecommit:', !Ref 'AWS::Region', ':', !Ref
'AWS::AccountId', ':', !Ref RepositoryName ] ]
        detail:
          event:
            - referenceCreated
            - referenceUpdated
          referenceType:
            - branch
          referenceName:
            - main
```

```
      Targets:
        -
          Arn:
            !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region', ':', !Ref
 'AWS::AccountId', ':', !Ref AppPipeline ] ]
          RoleArn: !GetAtt EventRole.Arn
          Id: codepipeline-AppPipeline
```

JSON

```
"EventRule": {
  "Type": "AWS::Events::Rule",
  "Properties": {
    "EventPattern": {
      "source": [
        "aws.codecommit"
      ],
      "detail-type": [
        "CodeCommit Repository State Change"
      ],
      "resources": [
        {
          "Fn::Join": [
            "",
            [
              "arn:aws:codecommit:",
              {
                "Ref": "AWS::Region"
              },
              ":",
              {
                "Ref": "AWS::AccountId"
              },
              ":",
              {
                "Ref": "RepositoryName"
              }
            ]
          ]
        }
      ],
      "detail": {
```

```
        "event": [
          "referenceCreated",
          "referenceUpdated"
        ],
        "referenceType": [
          "branch"
        ],
        "referenceName": [
          "main"
        ]
      }
    },
    "Targets": [
      {
        "Arn": {
          "Fn::Join": [
            "",
            [
              "arn:aws:codepipeline:",
              {
                "Ref": "AWS::Region"
              },
              ":",
              {
                "Ref": "AWS::AccountId"
              },
              ":",
              {
                "Ref": "AppPipeline"
              }
            ]
          ]
        },
        "RoleArn": {
          "Fn::GetAtt": [
            "EventRole",
            "Arn"
          ]
        },
        "Id": "codepipeline-AppPipeline"
      }
    ]
  }
```

```
},
```

3. Salva il modello aggiornato nel computer locale e quindi apri la console AWS CloudFormation .
4. Seleziona lo stack e scegli Create Change Set for Current Stack (Crea set di modifiche per lo stack corrente).
5. Carica il modello e quindi visualizza le modifiche elencate in AWS CloudFormation. Queste sono le modifiche da apportare allo stack. Le nuove risorse dovrebbero essere visibili nell'elenco.
6. Scegliere Execute (Esegui).

Per modificare i parametri della pipeline PollForSourceChanges

> **Important**
>
> In molti casi, il parametro `PollForSourceChanges` è preimpostato su "true" al momento della creazione di una pipeline. Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

- Nel modello, modifica `PollForSourceChanges` in `false`. Se non hai incluso `PollForSourceChanges` nella definizione della pipeline, aggiungilo e impostalo su `false`.

  Perché occorre apportare questa modifica? La modifica di questo parametro in `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

  YAML

```
        Name: Source
        Actions:
          -
            Name: SourceAction
            ActionTypeId:
              Category: Source
              Owner: AWS
              Version: 1
              Provider: CodeCommit
```

```
            OutputArtifacts:
              - Name: SourceOutput
            Configuration:
              BranchName: !Ref BranchName
              RepositoryName: !Ref RepositoryName
              PollForSourceChanges: false
            RunOrder: 1
```

JSON

```
{
  "Name": "Source",
  "Actions": [
    {
      "Name": "SourceAction",
      "ActionTypeId": {
        "Category": "Source",
        "Owner": "AWS",
        "Version": 1,
        "Provider": "CodeCommit"
      },
      "OutputArtifacts": [
        {
          "Name": "SourceOutput"
        }
      ],
      "Configuration": {
        "BranchName": {
          "Ref": "BranchName"
        },
        "RepositoryName": {
          "Ref": "RepositoryName"
        },
        "PollForSourceChanges": false
      },
      "RunOrder": 1
    }
  ]
},
```

GitHub connessioni

Utilizzi le connessioni per autorizzare e stabilire configurazioni che associano il provider di terze parti alle tue AWS risorse.

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hong Kong), Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Africa (Città del Capo), Medio Oriente (Bahrain), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna), Europa (Zurigo), Israele (Tel Aviv) o AWS GovCloud (Stati Uniti occidentali). Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Per aggiungere un'azione sorgente per il tuo repository GitHub o per GitHub Enterprise Cloud CodePipeline, puoi scegliere tra:

- Utilizza la procedura guidata di creazione della pipeline della CodePipeline console o la pagina Modifica azione per scegliere l'opzione del provider GitHub (versione 2). Vedi Creare una connessione a GitHub Enterprise Server (console) per aggiungere l'azione. La console ti aiuta a creare una risorsa di connessioni.

  > **Note**
  >
  > Per un tutorial che illustra come aggiungere una GitHub connessione e utilizzare l'opzione Full clone nella pipeline, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline

- Utilizza la CLI per aggiungere la configurazione dell'azione per l'`CodeStarSourceConnection`azione con il `GitHub` provider con i passaggi CLI mostrati in. Creazione di una pipeline (CLI)

> **Note**
>
> Puoi anche creare una connessione utilizzando la console Developer Tools in Impostazioni. Vedi Creare una connessione.

Prima di iniziare:

- Devi aver creato un account con GitHub.
- Devi aver già creato un repository di GitHub codice.
- Se il ruolo CodePipeline di servizio è stato creato prima del 18 dicembre 2019, potrebbe essere necessario aggiornarne le autorizzazioni da utilizzare `codestar-connections:UseConnection` per AWS CodeStar le connessioni. Per istruzioni, consulta Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline.

> **Note**
>
> Per creare la connessione, devi essere il proprietario dell' GitHub organizzazione. Per i repository che non appartengono a un'organizzazione, è necessario esserne il proprietario.

Argomenti

- Crea una connessione a GitHub (console)
- Creare una connessione a GitHub (CLI)

Crea una connessione a GitHub (console)

Segui questi passaggi per utilizzare la CodePipeline console per aggiungere un'azione di connessione per il tuo repository GitHub o per GitHub Enterprise Cloud.

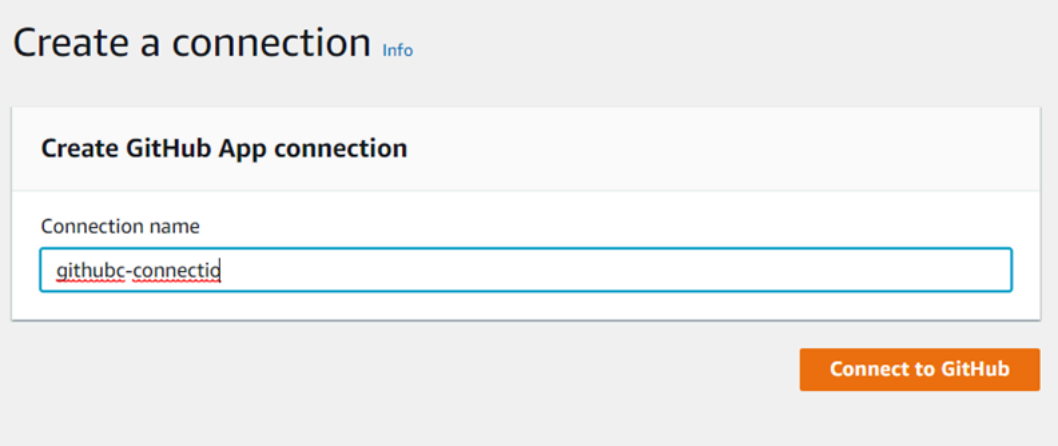
> **Note**
>
> In questi passaggi, puoi selezionare repository specifici in Accesso al repository. Tutti i repository non selezionati non saranno accessibili o visibili da. CodePipeline

Passaggio 1: crea o modifica la tua pipeline

1. Accedi alla CodePipeline console.
2. Scegliere una delle seguenti opzioni.
   - Scegli di creare una pipeline. Segui i passaggi descritti in Crea una pipeline per completare la prima schermata e scegli Avanti. Nella pagina Sorgente, in Source Provider, scegli GitHub (Versione 2).
   - Scegliete di modificare una pipeline esistente. Scegliete Modifica, quindi scegliete Modifica fase. Scegli di aggiungere o modificare l'azione sorgente. Nella pagina Modifica azione, in Nome azione, inserisci il nome dell'azione. Nel provider Action, scegli GitHub (Versione 2).
3. Esegui una di queste operazioni:
   - In Connessione, se non hai già creato una connessione al tuo provider, scegli Connetti a GitHub. Procedi al passaggio 2: Crea una connessione a GitHub.
   - In Connessione, se hai già creato una connessione al tuo provider, scegli la connessione. Procedi al passaggio 3: Salva l'azione di origine per la tua connessione.

Passaggio 2: Creare una connessione a GitHub

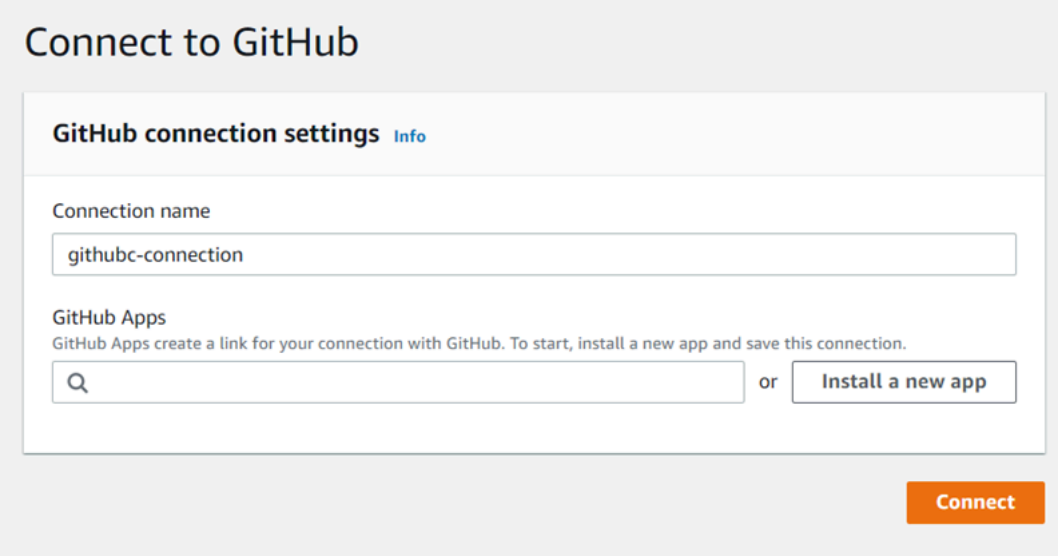
Dopo aver scelto di creare la connessione, viene visualizzata la GitHub pagina Connetti a.



Per creare una connessione a GitHub

1. Nelle impostazioni di GitHub connessione, il nome della connessione viene visualizzato in Nome connessione. Scegli Connect a GitHub. Viene visualizzata la pagina di richiesta di accesso.

2. Scegli Autorizza AWS connettore per GitHub. La pagina di connessione visualizza e mostra il campo GitHub App.



3. In GitHub App, scegli l'installazione di un'app o scegli Installa una nuova app per crearne una.

> **Note**
>
> È sufficiente installare una sola app per tutte le connessioni a un provider specifico. Se hai già installato il AWS Connector for GitHub app, sceglilo e salta questo passaggio.

4. Nella GitHub pagina Installa AWS Connector per, scegli l'account in cui desideri installare l'app.

> **Note**
>
> Puoi installare l'app solo una volta per ogni GitHub account. Se hai già installato l'app, puoi scegliere Configure (Configura) per passare a una pagina di modifica per l'installazione dell'app oppure è possibile utilizzare il pulsante Indietro per tornare alla console.

5. Nella GitHub pagina Install AWS Connector per, lascia le impostazioni predefinite e scegli Installa.
6. Nella GitHub pagina Connect to, l'ID di connessione per la nuova installazione viene visualizzato in GitHub App. Scegli Connetti.

Passaggio 3: Salva l'azione GitHub sorgente

Utilizza questi passaggi nella pagina Modifica azione per salvare l'azione di origine con le informazioni di connessione.

Per salvare l'azione GitHub sorgente

1. In Repository name (Nome repository), scegliere il nome del repository di terze parti.
2. In Trigger Pipeline puoi aggiungere trigger se la tua azione è un'azione. CodeConnections Per configurare la configurazione dei trigger della pipeline e, facoltativamente, filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice
3. In Output artifact format (Formato artefatto di output), occorre scegliere il formato degli artefatti.
   - Per memorizzare gli artefatti di output dell' GitHub azione utilizzando il metodo predefinito, scegliete default. CodePipeline L'azione accede ai file dal GitHub repository e archivia gli artefatti in un file ZIP nell'archivio degli artefatti della pipeline.
   - Per archiviare un file JSON contenente un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git, scegliere Full clone (Clone completo). Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild

     Se scegli questa opzione, dovrai aggiornare le autorizzazioni per il tuo ruolo di CodeBuild Project Service come mostrato in. Aggiungi le autorizzazioni per le connessioni a Bitbucket, Enterprise Server o.com CodeBuild GitClone GitHub GitHub GitLab Per un tutorial che mostra come usare l'opzione Full clone, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline
4. Scegli Avanti nella procedura guidata o Salva nella pagina Modifica azione.

Creare una connessione a GitHub (CLI)

È possibile utilizzare AWS Command Line Interface (AWS CLI) per creare una connessione.

Per farlo, utilizzare il comando create-connection.

> **Important**
>
> Per impostazione predefinita, una connessione creata tramite AWS CLI o AWS CloudFormation è in `PENDING` stato. Dopo aver creato una connessione con la CLI o AWS

CloudFormation, utilizza la console per modificare la connessione e definirne lo stato. `AVAILABLE`

Per creare una connessione

1. Apri un terminale (Linux, macOS o Unix) o prompt dei comandi (Windows). Usa il AWS CLI per eseguire il create-connection comando, specificando l'`--provider-typee --connection-name` per la tua connessione. In questo esempio, il nome del provider di terze parti è `GitHub` e il nome della connessione specificato è `MyConnection`.

   ```
   aws codestar-connections create-connection --provider-type GitHub --connection-name
    MyConnection
   ```

   In caso di esito positivo, questo comando restituisce informazioni dell'ARN della connessione simili alle seguenti.

   ```
   {
       "ConnectionArn": "arn:aws:codestar-connections:us-west-2:account_id:connection/
   aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f"
   }
   ```

2. Utilizzare la console per completare la connessione. Per ulteriori informazioni, consulta Aggiornare una connessione in sospeso.
3. Per impostazione predefinita, la pipeline rileva le modifiche al codice inviato al repository delle sorgenti di connessione. Per configurare la configurazione del trigger della pipeline per il rilascio manuale o per i tag Git, esegui una delle seguenti operazioni:

   - Per configurare la configurazione del trigger della pipeline in modo che inizi solo con una versione manuale, aggiungi la seguente riga alla configurazione:

     ```
     "DetectChanges": "false",
     ```

   - Per configurare la configurazione del trigger della pipeline per filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice Ad esempio, quanto segue si aggiunge al livello di pipeline della definizione JSON della pipeline. In questo esempio, `release-v0` e `release-v1` sono i tag Git da includere e `release-v2` il tag Git da escludere.

```
"triggers": [
            {
                "providerType": "CodeStarSourceConnection",
                "gitConfiguration": {
                    "sourceActionName": "Source",
                    "push": [
                        {
                            "tags": {
                                "includes": [
                                    "release-v0", "release-v1"
                                ],
                                "excludes": [
                                    "release-v2"
                                ]
                            }
                        }
                    ]
                }
            }
        ]
```

GitHub Connessioni Enterprise Server

Le connessioni consentono di autorizzare e stabilire configurazioni che associano il provider di terze parti alle risorse. AWS Per associare il repository di terze parti come fonte per la pipeline, si utilizza una connessione.

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hong Kong), Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Africa (Città del Capo), Medio Oriente (Bahrain), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna), Europa (Zurigo), Israele (Tel Aviv) o AWS GovCloud (Stati Uniti occidentali). Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Per aggiungere un'azione di origine di GitHub Enterprise Server in CodePipeline, è possibile scegliere tra:

- Utilizza la procedura guidata Crea pipeline della CodePipeline console o la pagina Modifica azione per scegliere l'opzione del provider GitHub Enterprise Server. Vedi Creare una connessione a GitHub Enterprise Server (console) per aggiungere l'azione. La console ti aiuta a creare una risorsa host e una risorsa di connessione.
- Utilizza la CLI per aggiungere la configurazione dell'azione per l'`CreateSourceConnection`azione con il `GitHubEnterpriseServer` provider e creare le tue risorse:
  - Per creare le tue risorse di connessione, consulta Creare un host e una connessione a GitHub Enterprise Server (CLI) Creare una risorsa host e una risorsa di connessione con la CLI.
  - Usa l'`CreateSourceConnection`esempio di configurazione dell'azione in CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite per aggiungere la tua azione come mostrato inCreazione di una pipeline (CLI).

> **Note**
>
> Puoi anche creare una connessione utilizzando la console Developer Tools in Impostazioni. Vedi Creare una connessione.

Prima di iniziare:

- È necessario aver creato un account con GitHub Enterprise Server e installato l'istanza di GitHub Enterprise Server sull'infrastruttura.

  > **Note**
  >
  > Ogni VPC può essere associato a un solo host (istanza GitHub Enterprise Server) alla volta.

- È necessario aver già creato un repository di codice con GitHub Enterprise Server.

Argomenti

- Creare una connessione a GitHub Enterprise Server (console)
- Creare un host e una connessione a GitHub Enterprise Server (CLI)

Creare una connessione a GitHub Enterprise Server (console)

Segui questi passaggi per utilizzare la CodePipeline console per aggiungere un'azione di connessione per il tuo repository GitHub Enterprise Server.

> **Note**
>
> GitHub Le connessioni Enterprise Server forniscono l'accesso solo ai repository di proprietà dell'account GitHub Enterprise Server utilizzato per creare la connessione.

Prima di iniziare:

Per una connessione host a GitHub Enterprise Server, è necessario aver completato i passaggi per creare una risorsa host per la connessione. Vedere Gestire gli host per le connessioni.

Passaggio 1: crea o modifica la tua pipeline

Per creare o modificare la tua pipeline

1. Accedi alla CodePipeline console.
2. Scegliere una delle seguenti opzioni.
   - Scegli di creare una pipeline. Segui i passaggi descritti in Crea una pipeline per completare la prima schermata e scegli Avanti. Nella pagina Sorgente, in Provider di origine, scegli GitHub Enterprise Server.
   - Scegliete di modificare una pipeline esistente. Scegliete Modifica, quindi scegliete Modifica fase. Scegli di aggiungere o modificare l'azione sorgente. Nella pagina Modifica azione, in Nome azione, inserisci il nome dell'azione. In Action provider, scegli GitHub Enterprise Server.
3. Esegui una di queste operazioni:
   - In Connessione, se non hai già creato una connessione al tuo provider, scegli Connetti a GitHub Enterprise Server. Procedi al passaggio 2: creazione di una connessione a GitHub Enterprise Server.

- In Connessione, se hai già creato una connessione al tuo provider, scegli la connessione. Procedi al passaggio 3: Salva l'azione di origine per la tua connessione.

Creare una connessione a GitHub Enterprise Server

Dopo aver scelto di creare la connessione, viene visualizzata la pagina Connect to GitHub Enterprise Server.

> **Important**
>
> AWS CodeConnections non supporta la versione 2.22.0 di GitHub Enterprise Server a causa di un problema noto nella versione. Per connetterti, esegui l'aggiornamento alla versione 2.22.1 o all'ultima versione disponibile.

Per connettersi a Enterprise Server GitHub

1. In Connection Name (Nome connessione), immetti un nome per la connessione.
2. In URL, immetti l'endpoint per il server.

   > **Note**
   >
   > Se l'URL fornito è già stato utilizzato per configurare un GitHub Enterprise Server per una connessione, verrà richiesto di scegliere l'ARN della risorsa host creato in precedenza per quell'endpoint.

3. Se hai lanciato il tuo server in un VPC Amazon e vuoi connetterti con il tuo VPC, scegli Use a VPC (Utilizzo di un VPC) e completa quanto segue.

   a. In VPC ID (ID VPC), seleziona l'ID VPC. Assicurati di scegliere il VPC per l'infrastruttura in cui è installata l'istanza di GitHub Enterprise Server o un VPC con accesso all'istanza GitHub Enterprise Server tramite VPN o Direct Connect.

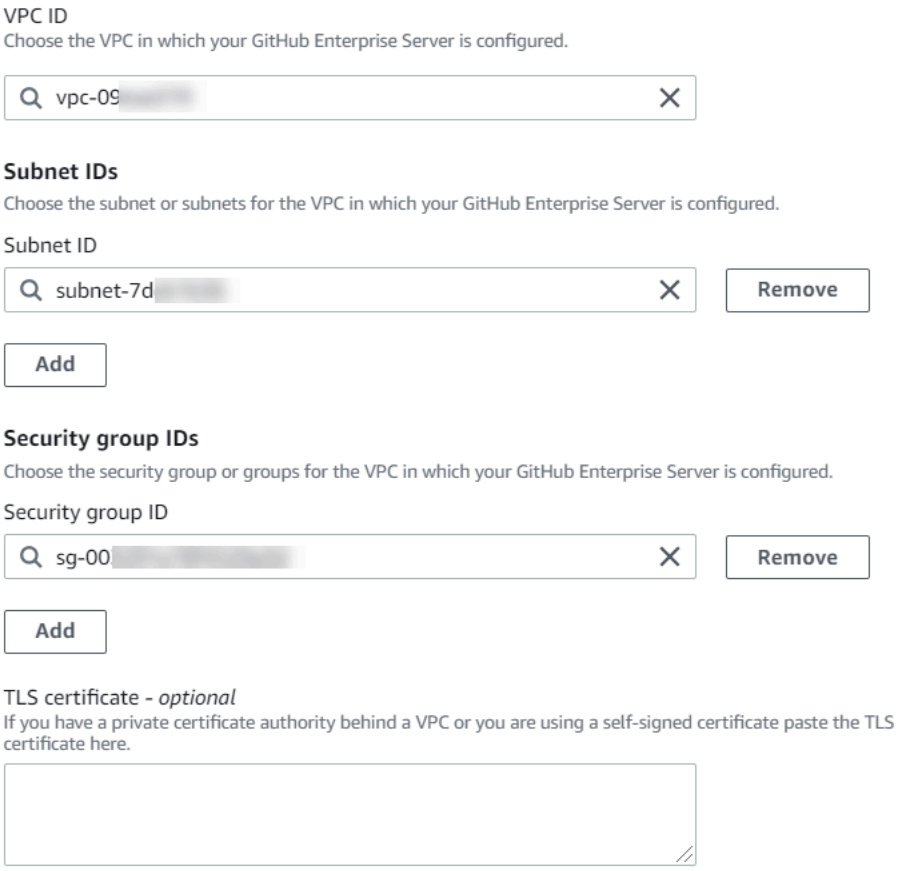
   b. In Subnet ID (ID sottorete), scegliere Add (Aggiungi). Nel campo, selezionare l'ID della sottorete che si desidera utilizzare per l'host. È possibile scegliere fino a 10 sottoreti.

      Assicurati di scegliere la sottorete per l'infrastruttura in cui è installata l'istanza di GitHub Enterprise Server o una sottorete con accesso all'istanza GitHub Enterprise Server installata tramite VPN o Direct Connect.

c. In Security group IDs (ID gruppo di sicurezza), scegliere Add (Aggiungi). Nel campo, selezionare il gruppo di sicurezza che si desidera utilizzare per l'host. È possibile creare fino a 10 gruppi di sicurezza.

   Assicurati di scegliere il gruppo di sicurezza per l'infrastruttura in cui è installata l'istanza di GitHub Enterprise Server o un gruppo di sicurezza con accesso all'istanza GitHub Enterprise Server installata tramite VPN o Direct Connect.

d. Se hai configurato un VPC privato e hai configurato l'istanza di GitHub Enterprise Server per eseguire la convalida TLS utilizzando un'autorità di certificazione non pubblica, nel certificato TLS inserisci l'ID del certificato. Il valore del Certificato TLS deve essere la chiave pubblica del certificato.



4. Scegli Connect to GitHub Enterprise Server. La connessione creata viene mostrata con uno stato Pending (In attesa). Viene creata una risorsa host per la connessione con le informazioni sul server fornite. Per il nome dell'host, viene utilizzato l'URL.
5. Scegli Update pending connection (Aggiornare la connessione in attesa).
6. Se richiesto, nella pagina di accesso GitHub Enterprise, accedi con le tue credenziali GitHub Enterprise.

7. Nella pagina Crea GitHub app, scegli un nome per la tua app.
8. <app-name>Nella pagina di GitHub autorizzazione, scegli Autorizza.
9. Nella pagina di installazione dell'app, un messaggio indica che l'app Connector è pronta per essere installata. Se disponi di più organizzazioni, potrebbe essere richiesto di scegliere l'organizzazione in cui si desidera installare l'app.

   Scegli le impostazioni del repository in cui desideri installare l'app. Scegli Installa.
10. La pagina di connessione mostra la connessione creata in uno stato Available (Disponibile).

Fase 3: Salvare l'azione sorgente di GitHub Enterprise Server

Utilizzare questi passaggi nella procedura guidata o nella pagina Modifica azione per salvare l'azione di origine con le informazioni di connessione.

Per completare e salvare l'azione sorgente con la connessione

1. In Repository name (Nome repository), scegliere il nome del repository di terze parti.
2. In Trigger Pipeline puoi aggiungere trigger se la tua azione è un'azione. CodeConnections Per configurare la configurazione dei trigger della pipeline e, facoltativamente, filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice
3. In Output artifact format (Formato artefatto di output), occorre scegliere il formato degli artefatti.
   - Per memorizzare gli artefatti di output dell'azione GitHub Enterprise Server utilizzando il metodo predefinito, scegliete default. CodePipeline L'azione accede ai file dal repository di GitHub Enterprise Server e archivia gli artefatti in un file ZIP nell'archivio degli artefatti della pipeline.
   - Per archiviare un file JSON contenente un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git, scegliere Full clone (Clone completo). Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild
4. Scegli Avanti nella procedura guidata o Salva nella pagina Modifica azione.

Creare un host e una connessione a GitHub Enterprise Server (CLI)

È possibile utilizzare AWS Command Line Interface (AWS CLI) per creare una connessione.

Per farlo, utilizzare il comando create-connection.

> **Important**
>
> Per impostazione predefinita, una connessione creata tramite AWS CLI o AWS CloudFormation è in `PENDING` stato. Dopo aver creato una connessione con la CLI o AWS CloudFormation, utilizza la console per modificare la connessione e definirne lo stato. `AVAILABLE`

È possibile utilizzare AWS Command Line Interface (AWS CLI) per creare un host per le connessioni installate.

> **Note**
>
> È possibile creare un host solo una volta per account GitHub Enterprise Server. Tutte le connessioni a un account GitHub Enterprise Server specifico utilizzeranno lo stesso host.

È possibile utilizzare un host per rappresentare l'endpoint per l'infrastruttura in cui è installato il provider di terze parti. Dopo aver completato la creazione dell'host con la CLI, lo stato dell'host è in sospeso. Quindi configuri, o registri, l'host per spostarlo allo stato Disponibile. Quando l'host sarà disponibile, completa i passaggi per creare una connessione.

Per farlo, utilizzare il comando create-host.

> **Important**
>
> Per impostazione predefinita, un host creato tramite `Pending` lo stato AWS CLI è attivo. Dopo aver creato un host con la CLI, utilizza la console o la CLI per configurare l'host e impostarne lo stato. `Available`

Per creare un host

1. Apri un terminale (Linux, macOS o Unix) o prompt dei comandi (Windows). Usa il AWS CLI per eseguire il create-host comando, specificando e `--provider-endpoint` per `--name` `--provider-type` la tua connessione. In questo esempio, il nome del provider di terze parti è `GitHubEnterpriseServer` e l'endpoint è `my-instance.dev`.

```
aws codestar-connections create-host --name MyHost --provider-type
 GitHubEnterpriseServer --provider-endpoint "https://my-instance.dev"
```

In caso di esito positivo, questo comando restituisce informazioni dell'Amazon Resource Name (ARN) dell'host simili alle seguenti.

```
{
    "HostArn": "arn:aws:codestar-connections:us-west-2:account_id:host/My-
Host-28aef605"
}
```

Dopo questo passaggio, l'host è nello stato `PENDING`.

2. Utilizza la console per completare la configurazione dell'host e sposta l'host nello stato `Available`.

Per creare una connessione a GitHub Enterprise Server

1. Apri un terminale (Linux, macOS o Unix) o prompt dei comandi (Windows). Utilizzare il AWS CLI per eseguire il create-connection comando, specificando `--host-arn` e `--connection-name` per la connessione.

```
aws codestar-connections create-connection --host-arn arn:aws:codestar-
connections:us-west-2:account_id:host/MyHost-234EXAMPLE --connection-name
 MyConnection
```

In caso di esito positivo, questo comando restituisce informazioni dell'ARN della connessione simili alle seguenti.

```
{
    "ConnectionArn": "arn:aws:codestar-connections:us-west-2:account_id:connection/
aEXAMPLE-8aad"
}
```

2. Utilizzare la console per configurare la connessione in attesa.
3. Per impostazione predefinita, la pipeline rileva le modifiche al codice inviato al repository delle sorgenti di connessione. Per configurare la configurazione del trigger della pipeline per il rilascio manuale o per i tag Git, esegui una delle seguenti operazioni:

- Per configurare la configurazione del trigger della pipeline in modo che inizi solo con una versione manuale, aggiungi la seguente riga alla configurazione:

```
"DetectChanges": "false",
```

- Per configurare la configurazione del trigger della pipeline per filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice Ad esempio, quanto segue si aggiunge al livello di pipeline della definizione JSON della pipeline. In questo esempio, `release-v0` e `release-v1` sono i tag Git da includere e `release-v2` il tag Git da escludere.

```
"triggers": [
            {
                "providerType": "CodeStarSourceConnection",
                "gitConfiguration": {
                    "sourceActionName": "Source",
                    "push": [
                        {
                            "tags": {
                                "includes": [
                                    "release-v0", "release-v1"
                                ],
                                "excludes": [
                                    "release-v2"
                                ]
                            }
                        }
                    ]
                }
            }
        ]
```

GitLabconnessioni.com

Le connessioni consentono di autorizzare e stabilire configurazioni che associano il provider di terze parti alle risorse dell'utente AWS . Per associare il repository di terze parti come fonte per la pipeline, si utilizza una connessione.

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hong Kong), Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Africa (Città del Capo), Medio Oriente (Bahrain), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna), Europa (Zurigo), Israele (Tel Aviv) o AWS GovCloud (Stati Uniti occidentali). Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Per aggiungere un'azione GitLab source.com in CodePipeline, puoi scegliere tra:

- Utilizza la procedura guidata di creazione della pipeline della CodePipeline console o la pagina Modifica azione per scegliere l'opzione del GitLabprovider. Vedi Crea una connessione a .com (console) GitLab per aggiungere l'azione. La console ti aiuta a creare una risorsa di connessioni.
- Utilizza la CLI per aggiungere la configurazione dell'azione per l'`CreateSourceConnection`azione con il `GitLab` provider come segue:
  - Per creare le tue risorse di connessione, consulta Creare una connessione a GitLab .com (CLI) Creare una risorsa di connessione con la CLI.
  - Usa l'`CreateSourceConnection`esempio di configurazione dell'azione in CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite per aggiungere la tua azione come mostrato inCreazione di una pipeline (CLI).

> **Note**
>
> Puoi anche creare una connessione utilizzando la console Developer Tools in Impostazioni. Vedi Creare una connessione.

> **Note**
>
> Autorizzando l'installazione di questa connessione GitLab su.com, concedi al nostro servizio le autorizzazioni per elaborare i tuoi dati accedendo al tuo account e puoi revocare le autorizzazioni in qualsiasi momento disinstallando l'applicazione.

Prima di iniziare:

- Devi aver già creato un account con.com. GitLab

> **Note**
>
> Le connessioni forniscono l'accesso solo ai repository di proprietà dell'account utilizzato per creare e autorizzare la connessione.

> **Note**
>
> È possibile creare connessioni a un repository in cui si ricopre il ruolo di proprietario e quindi la connessione può essere utilizzata con il repository con risorse come. GitLab CodePipeline Per i repository nei gruppi, non è necessario essere il proprietario del gruppo.

- Per specificare una fonte per la tua pipeline, devi aver già creato un repository su gitlab.com.

Argomenti

Crea una connessione a .com (console) GitLab

Usa questi passaggi per utilizzare la CodePipeline console e aggiungere un'azione di connessione per il tuo progetto (repository) in GitLab.

Per creare o modificare la tua pipeline

1. Accedi alla CodePipeline console.

2. Scegliere una delle seguenti opzioni.

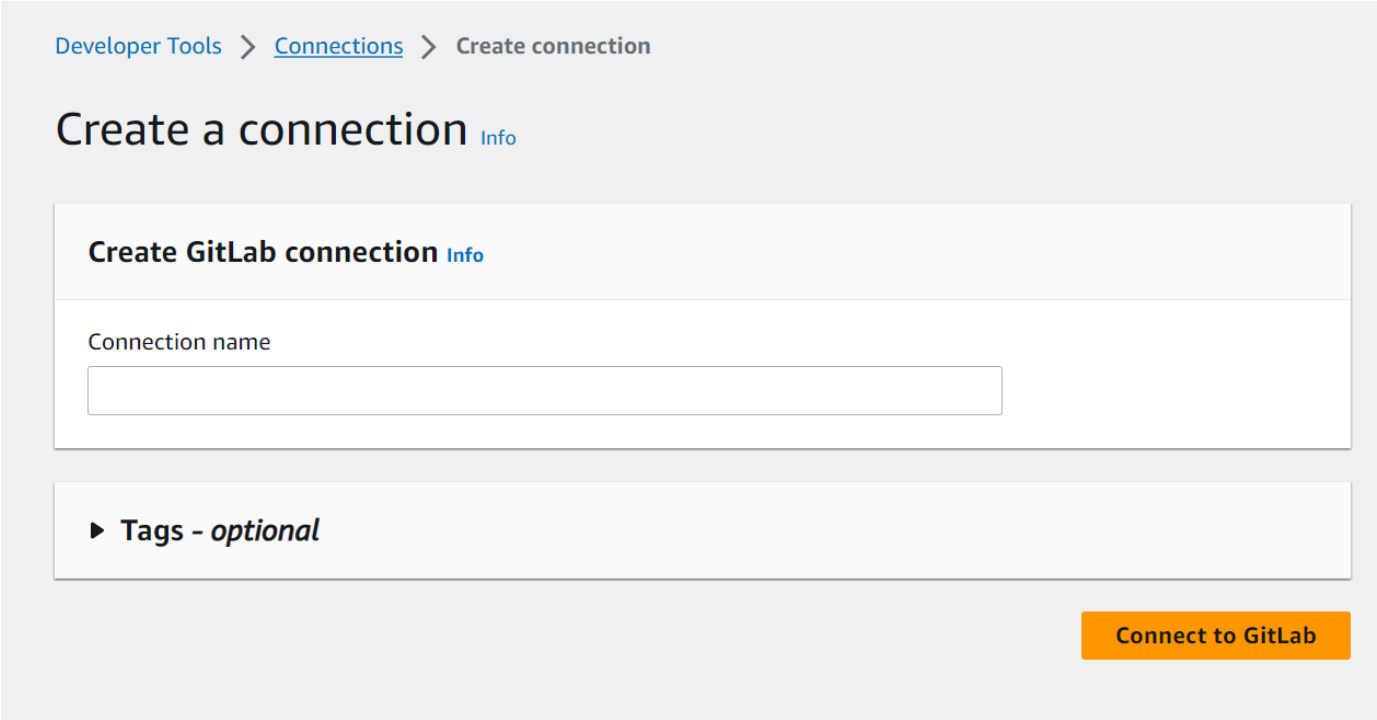
   - Scegli di creare una pipeline. Segui i passaggi descritti in Crea una pipeline per completare la prima schermata e scegli Avanti. Nella pagina Sorgente, in Source Provider, scegli GitLab.
   - Scegliete di modificare una pipeline esistente. Scegliete Modifica, quindi scegliete Modifica fase. Scegli di aggiungere o modificare l'azione sorgente. Nella pagina Modifica azione, in Nome azione, inserisci il nome dell'azione. In Provider Action, scegli GitLab.

3. Esegui una di queste operazioni:

   - In Connessione, se non hai già creato una connessione con il tuo provider, scegli Connetti a GitLab. Procedi al passaggio 4 per creare la connessione.
   - In Connessione, se hai già creato una connessione con il tuo provider, scegli la connessione. Procedi al passaggio 9.

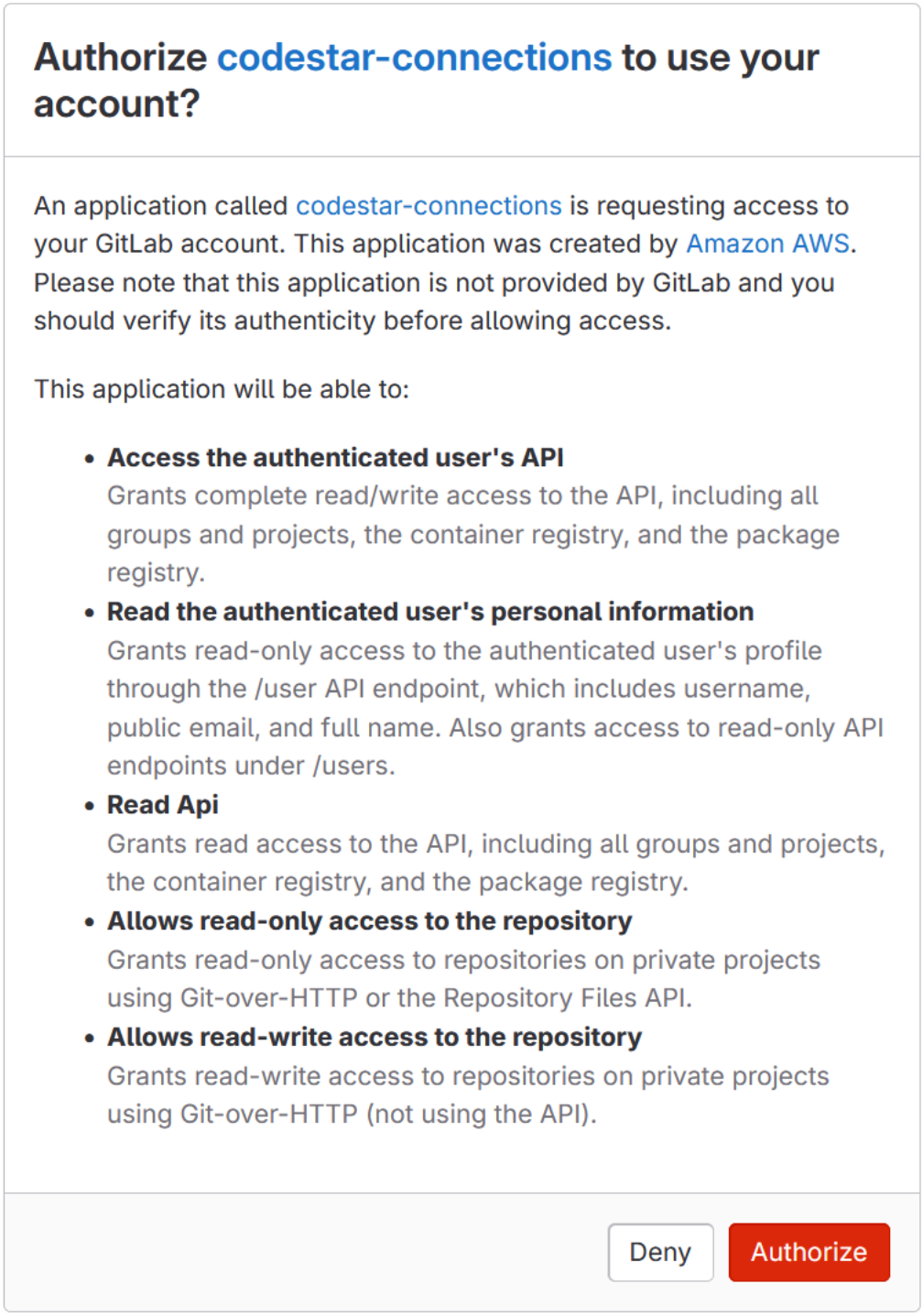
   > **Note**
   >
   > Se chiudi la finestra pop-up prima che venga creata una connessione GitLab .com, devi aggiornare la pagina.

4. Per creare una connessione a un repository GitLab .com, in Seleziona un provider, scegli. GitLab In Connection name (Nome connessione), immetti il nome della connessione che desideri creare. Scegli Connect a GitLab.



5. Quando viene visualizzata la pagina di accesso per GitLab .com, accedi con le tue credenziali, quindi scegli Accedi.
6. Se è la prima volta che autorizzi la connessione, viene visualizzata una pagina di autorizzazione con un messaggio che richiede l'autorizzazione per la connessione per accedere al tuo account.com. GitLab

   Seleziona Authorize (Autorizza).



7. Il browser torna alla pagina della console delle connessioni. In Crea GitLab connessione, la nuova connessione viene mostrata in Nome connessione.
8. Scegli Connect a GitLab.

   Verrai reindirizzato alla CodePipeline console.

> **Note**
>
> Dopo aver creato correttamente una connessione GitLab .com, nella finestra principale verrà visualizzato un banner di successo.
> Se non hai precedentemente effettuato l'accesso GitLab al computer corrente, dovrai chiudere manualmente la finestra pop-up.

9. In Repository name, scegli il nome del tuo progetto in GitLab specificando il percorso del progetto con lo spazio dei nomi. Ad esempio, per un repository a livello di gruppo, inserisci il nome del repository nel seguente formato:. `group-name/repository-name` Per ulteriori informazioni sul percorso e sullo spazio dei nomi, consultate il campo in https://docs.gitlab.com/ee/api/projects.html #. `path_with_namespace` get-single-project Per ulteriori informazioni sullo spazio dei nomi in GitLab, vedi https://docs.gitlab.com/ee/user/namespace/.

> **Note**
>
> Per i gruppi in GitLab, è necessario specificare manualmente il percorso del progetto con lo spazio dei nomi. Ad esempio, per un repository denominato `myrepo` in un gruppo`mygroup`, inserisci quanto segue:. `mygroup/myrepo` Puoi trovare il percorso del progetto con lo spazio dei nomi nell'URL in. GitLab

10. In Pipeline triggers puoi aggiungere trigger se la tua azione è un'azione. CodeConnections Per configurare la configurazione dei trigger della pipeline e, facoltativamente, filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice
11. In Branch name (Nome ramo), scegliere il ramo in cui si desidera che la pipeline rilevi le modifiche dell'origine.

> **Note**
>
> Se il nome del ramo non viene compilato automaticamente, non hai accesso come proprietario al repository. Il nome del progetto non è valido oppure la connessione utilizzata non ha accesso al progetto/repository.

12. In Output artifact format (Formato artefatto di output), occorre scegliere il formato degli artefatti.

- Per memorizzare gli elementi di output dall'azione GitLab .com utilizzando il metodo predefinito, scegliete default. CodePipeline L'azione accede ai file dal repository GitLab .com e archivia gli artefatti in un file ZIP nell'archivio degli artefatti della pipeline.
- Per archiviare un file JSON contenente un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git, scegliere Full clone (Clone completo). Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild

  Se scegli questa opzione, dovrai aggiornare le autorizzazioni per il tuo ruolo di CodeBuild Project Service come mostrato in. Aggiungi le autorizzazioni per le connessioni a Bitbucket, Enterprise Server o.com CodeBuild GitClone GitHub GitHub GitLab Per un tutorial che mostra come usare l'opzione Full clone, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline

13. Scegliete di salvare l'azione sorgente e continuate.

Creare una connessione a GitLab .com (CLI)

È possibile utilizzare AWS Command Line Interface (AWS CLI) per creare una connessione.

Per farlo, utilizzare il comando create-connection.

> **Important**
>
> Per impostazione predefinita, una connessione creata tramite AWS CLI o AWS CloudFormation è in `PENDING` stato. Dopo aver creato una connessione con la CLI o AWS CloudFormation, utilizza la console per modificare la connessione e definirne lo stato. `AVAILABLE`

Per creare una connessione

1. Apri un terminale (Linux, macOS o Unix) o prompt dei comandi (Windows). Usa il AWS CLI per eseguire il create-connection comando, specificando l'`--provider-typee --connection-name` per la tua connessione. In questo esempio, il nome del provider di terze parti è `GitLab` e il nome della connessione specificato è `MyConnection`.

   ```
   aws codestar-connections create-connection --provider-type GitLab --connection-name
    MyConnection
   ```

In caso di esito positivo, questo comando restituisce informazioni dell'ARN della connessione simili alle seguenti.

```
{
    "ConnectionArn": "arn:aws:codestar-connections:us-west-2:account_id:connection/
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f"
}
```

2. Utilizzare la console per completare la connessione. Per ulteriori informazioni, consulta Aggiornare una connessione in sospeso.
3. Per impostazione predefinita, la pipeline rileva le modifiche al codice inviato al repository delle sorgenti di connessione. Per configurare la configurazione del trigger della pipeline per il rilascio manuale o per i tag Git, esegui una delle seguenti operazioni:

   - Per configurare la configurazione del trigger della pipeline in modo che inizi solo con una versione manuale, aggiungi la seguente riga alla configurazione:

     ```
     "DetectChanges": "false",
     ```

   - Per configurare la configurazione del trigger della pipeline per filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice Ad esempio, quanto segue si aggiunge al livello di pipeline della definizione JSON della pipeline. In questo esempio, `release-v0` e `release-v1` sono i tag Git da includere e `release-v2` il tag Git da escludere.

     ```
     "triggers": [
                {
                    "providerType": "CodeStarSourceConnection",
                    "gitConfiguration": {
                        "sourceActionName": "Source",
                        "push": [
                            {
                                "tags": {
                                    "includes": [
                                        "release-v0", "release-v1"
                                    ],
                                    "excludes": [
                                        "release-v2"
                                    ]
                                }
     ```

```
                    }
                ]
            }
        }
    ]
```

Connessioni per la GitLab gestione automatica

Le connessioni consentono di autorizzare e stabilire configurazioni che associano il provider di terze parti alle risorse. AWS Per associare il repository di terze parti come fonte per la pipeline, si utilizza una connessione.

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hong Kong), Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Africa (Città del Capo), Medio Oriente (Bahrain), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna), Europa (Zurigo), Israele (Tel Aviv) o AWS GovCloud (Stati Uniti occidentali). Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Per aggiungere un'azione sorgente GitLab autogestita in CodePipeline, puoi scegliere tra:

- Utilizza la procedura guidata Crea pipeline della CodePipeline console o la pagina Modifica azione per scegliere l'opzione del provider GitLabautogestito. Vedi Crea una connessione a una rete GitLab autogestita (console) per aggiungere l'azione. La console ti aiuta a creare una risorsa host e una risorsa di connessione.
- Utilizza la CLI per aggiungere la configurazione dell'azione per l'`CreateSourceConnection`azione con il `GitLabSelfManaged` provider e creare le tue risorse:
  - Per creare le tue risorse di connessione, consulta Creazione di un host e connessione a sistemi GitLab autogestiti (CLI) Creare una risorsa host e una risorsa di connessione con la CLI.
  - Usa l'`CreateSourceConnection`esempio di configurazione dell'azione in CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com

e GitLab azioni autogestite per aggiungere la tua azione come mostrato inCreazione di una pipeline (CLI).

> **Note**
>
> Puoi anche creare una connessione utilizzando la console Developer Tools in Impostazioni. Vedi Creare una connessione.

Prima di iniziare:

- È necessario aver già creato un account con GitLab GitLab Enterprise Edition o GitLab Community Edition con installazione autogestita. Per ulteriori informazioni, consulta https://docs.gitlab.com/ee/subscriptions/self_managed/.

  > **Note**
  >
  > Le connessioni forniscono l'accesso solo all'account utilizzato per creare e autorizzare la connessione.

  > **Note**
  >
  > È possibile creare connessioni a un repository in cui si ricopre il ruolo di proprietario e quindi la connessione può essere utilizzata con risorse come. GitLab CodePipeline Per i repository nei gruppi, non è necessario essere il proprietario del gruppo.

- È necessario aver già creato un token di accesso GitLab personale (PAT) solo con la seguente autorizzazione limitata: api. Per ulteriori informazioni, consulta https://docs.gitlab.com/ee/user/profile/personal_access_tokens.html. Devi essere un amministratore per creare e utilizzare il PAT.

  > **Note**
  >
  > Il PAT viene utilizzato per autorizzare l'host e non viene altrimenti archiviato o utilizzato dalle connessioni. Per configurare un host, puoi creare un PAT temporaneo e quindi, dopo aver configurato l'host, puoi eliminare il PAT.

- Puoi scegliere di configurare il tuo host in anticipo. Puoi configurare un host con o senza un VPC. Per dettagli sulla configurazione del VPC e informazioni aggiuntive sulla creazione di un host, consulta Creare un host.

Argomenti

Crea una connessione a una rete GitLab autogestita (console)

Segui questi passaggi per utilizzare la CodePipeline console e aggiungere un'azione di connessione per il tuo repository GitLab autogestito.

> **Note**
>
> GitLab le connessioni autogestite forniscono l'accesso solo agli archivi di proprietà dell'account GitLab autogestito utilizzato per creare la connessione.

Prima di iniziare:

Per una connessione host a GitLab gestione automatica, è necessario aver completato i passaggi per creare una risorsa host per la connessione. Vedi Gestire gli host per le connessioni.

Passaggio 1: crea o modifica la tua pipeline

Per creare o modificare la tua pipeline

1. Accedi alla CodePipeline console.
2. Scegliere una delle seguenti opzioni.
   - Scegli di creare una pipeline. Segui i passaggi descritti in Crea una pipeline per completare la prima schermata e scegli Avanti. Nella pagina Sorgente, in Provider di origine, scegli Gestione GitLab automatica.
   - Scegli di modificare una pipeline esistente. Scegliete Modifica, quindi scegliete Modifica fase. Scegli di aggiungere o modificare l'azione sorgente. Nella pagina Modifica azione, in Nome azione, inserisci il nome dell'azione. In Action provider, scegli GitLab Autogestito.
3. Esegui una di queste operazioni:

- In Connessione, se non hai già creato una connessione con il tuo provider, scegli Connetti a GitLab gestione automatica. Procedi al passaggio 2: Crea una connessione a gestione GitLab automatica.
- In Connessione, se hai già creato una connessione al tuo provider, scegli la connessione. Procedi al passaggio 3: Salva l'azione di origine per la tua connessione.

Crea una connessione a gestione GitLab automatica

Dopo aver scelto di creare la connessione, viene visualizzata la pagina Connect to GitLab self-managed.

Per connettersi a una rete autogestita GitLab

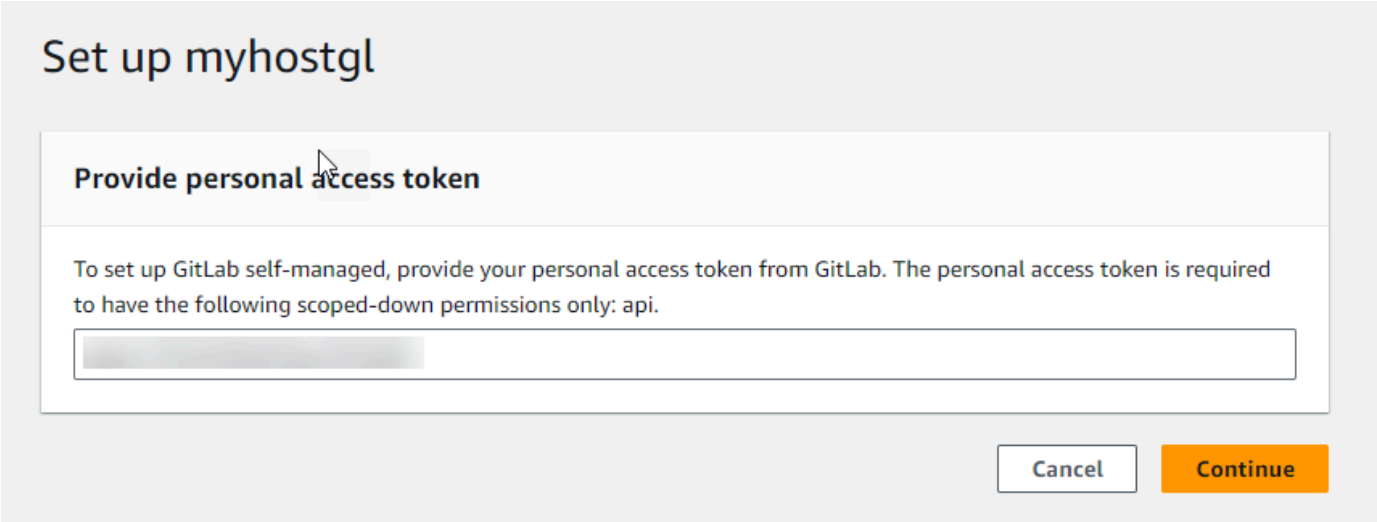
1. In Connection Name (Nome connessione), immetti un nome per la connessione.
2. In URL, immetti l'endpoint per il server.

   > **Note**
   >
   > Se l'URL fornito è già stato utilizzato per configurare un host per una connessione, ti verrà richiesto di scegliere l'ARN della risorsa host creato in precedenza per quell'endpoint.

3. Se hai avviato il tuo server in un Amazon VPC e desideri connetterti con il tuo VPC, scegli Usa un VPC e completa le informazioni per il VPC.
4. Scegli Connect to GitLab self-managed. La connessione creata viene mostrata con uno stato Pending (In attesa). Viene creata una risorsa host per la connessione con le informazioni sul server fornite. Per il nome dell'host, viene utilizzato l'URL.
5. Scegli Update pending connection (Aggiornare la connessione in attesa).
6. Se si apre una pagina con un messaggio di reindirizzamento che conferma che desideri continuare a contattare il provider, scegli Continua. Inserisci l'autorizzazione per il provider.
7. Viene visualizzata la pagina Configura ***`nome_host`***. In Fornisci token di accesso personale, fornisci GitLab al tuo PAT solo la seguente autorizzazione limitata:. `api`

   > **Note**
   >
   > Solo un amministratore può creare e utilizzare il PAT.

Scegli Continua.



8. La pagina di connessione mostra la connessione creata in uno stato Available (Disponibile).

Fase 3: Salva l'azione sorgente GitLab autogestita

Utilizza questi passaggi nella procedura guidata o nella pagina Modifica azione per salvare l'azione di origine con le informazioni di connessione.

Per completare e salvare l'azione sorgente con la connessione

1. In Repository name (Nome repository), scegliere il nome del repository di terze parti.
2. In Trigger Pipeline puoi aggiungere trigger se la tua azione è un'azione. CodeConnections Per configurare la configurazione dei trigger della pipeline e, facoltativamente, filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice
3. In Output artifact format (Formato artefatto di output), occorre scegliere il formato degli artefatti.
   - Per memorizzare gli artefatti di output derivanti dall'azione GitLab autogestita utilizzando il metodo predefinito, scegliete default. CodePipeline L'azione accede ai file dal repository e archivia gli artefatti in un file ZIP nell'archivio degli artefatti della pipeline.
   - Per archiviare un file JSON contenente un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git, scegliere Full clone (Clone completo). Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild
4. Scegli Avanti nella procedura guidata o Salva nella pagina Modifica azione.

Creazione di un host e connessione a sistemi GitLab autogestiti (CLI)

È possibile utilizzare AWS Command Line Interface (AWS CLI) per creare una connessione.

Per farlo, utilizzare il comando create-connection.

> **Important**
>
> Per impostazione predefinita, una connessione creata tramite AWS CLI o AWS CloudFormation è in `PENDING` stato. Dopo aver creato una connessione con la CLI o AWS CloudFormation, utilizza la console per modificare la connessione e definirne lo stato. `AVAILABLE`

È possibile utilizzare AWS Command Line Interface (AWS CLI) per creare un host per le connessioni installate.

È possibile utilizzare un host per rappresentare l'endpoint per l'infrastruttura in cui è installato il provider di terze parti. Dopo aver completato la creazione dell'host con la CLI, lo stato dell'host è in sospeso. Quindi configuri, o registri, l'host per spostarlo allo stato Disponibile. Quando l'host sarà disponibile, completa i passaggi per creare una connessione.

Per farlo, utilizzare il comando create-host.

> **Important**
>
> Per impostazione predefinita, un host creato tramite `Pending` lo stato AWS CLI è attivo. Dopo aver creato un host con la CLI, utilizza la console o la CLI per configurare l'host e impostarne lo stato. `Available`

Per creare un host

1. Apri un terminale (Linux, macOS o Unix) o prompt dei comandi (Windows). Usa il AWS CLI per eseguire il create-host comando, specificando e `--provider-endpoint` per `--name` `--provider-type` la tua connessione. In questo esempio, il nome del provider di terze parti è `GitLabSelfManaged` e l'endpoint è `my-instance.dev`.

   ```
   aws codestar-connections create-host --name MyHost --provider-type
    GitLabSelfManaged --provider-endpoint "https://my-instance.dev"
   ```

In caso di esito positivo, questo comando restituisce informazioni dell'Amazon Resource Name (ARN) dell'host simili alle seguenti.

```
{
    "HostArn": "arn:aws:codestar-connections:us-west-2:account_id:host/My-
Host-28aef605"
}
```

Dopo questo passaggio, l'host è nello stato `PENDING`.

2. Utilizza la console per completare la configurazione dell'host e sposta l'host nello stato `Available`.

Per creare una connessione a gestione automatica GitLab

1. Apri un terminale (Linux, macOS o Unix) o prompt dei comandi (Windows). Usa il AWS CLI per eseguire il create-connection comando, specificando l'`--host-arn` e `--connection-name` per la tua connessione.

```
aws codestar-connections create-connection --host-arn arn:aws:codestar-
connections:us-west-2:account_id:host/MyHost-234EXAMPLE --connection-name
 MyConnection
```

In caso di esito positivo, questo comando restituisce informazioni dell'ARN della connessione simili alle seguenti.

```
{
    "ConnectionArn": "arn:aws:codestar-connections:us-west-2:account_id:connection/
aEXAMPLE-8aad"
}
```

2. Utilizzare la console per configurare la connessione in attesa.
3. Per impostazione predefinita, la pipeline rileva le modifiche al codice inviato al repository delle sorgenti di connessione. Per configurare la configurazione del trigger della pipeline per il rilascio manuale o per i tag Git, esegui una delle seguenti operazioni:

   - Per configurare la configurazione del trigger della pipeline in modo che inizi solo con una versione manuale, aggiungi la seguente riga alla configurazione:

```
"DetectChanges": "false",
```

- Per configurare la configurazione del trigger della pipeline per filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice Ad esempio, quanto segue si aggiunge al livello di pipeline della definizione JSON della pipeline. In questo esempio, `release-v0` e `release-v1` sono i tag Git da includere e `release-v2` il tag Git da escludere.

```
"triggers": [
            {
                "providerType": "CodeStarSourceConnection",
                "gitConfiguration": {
                    "sourceActionName": "Source",
                    "push": [
                        {
                            "tags": {
                                "includes": [
                                    "release-v0", "release-v1"
                                ],
                                "excludes": [
                                    "release-v2"
                                ]
                            }
                        }
                    ]
                }
            }
        ]
```

Modificare una tubazione in CodePipeline

Una pipeline descrive il processo di rilascio AWS CodePipeline da seguire, incluse le fasi e le azioni che devono essere completate. Puoi modificare una pipeline per aggiungere o rimuovere tali elementi. Tuttavia, quando modifichi una pipeline, elementi come il nome della pipeline o i suoi metadati non possono essere modificati.

Puoi modificare il tipo di pipeline, le variabili e i trigger utilizzando la pagina di modifica della pipeline. Puoi anche aggiungere o modificare fasi e azioni nella tua pipeline.

A differenza della creazione di una pipeline, la modifica di una pipeline non avvia l'esecuzione della versione più recente tramite la pipeline stessa. Se vuoi avviare l'elaborazione della versione più recente tramite una pipeline che hai appena modificato, è necessario riavviarla manualmente. In caso contrario, la pipeline modificata viene eseguita la volta successiva alla modifica di un percorso di origine configurato nella fase di origine. Per informazioni, consulta Avvio manuale di una pipeline.

Puoi aggiungere azioni alla tua pipeline che si trovano in una AWS regione diversa dalla tua pipeline. Quando un Servizio AWS è il provider di un'azione e questo tipo di azione/tipo di provider si trova in una AWS regione diversa dalla pipeline, si tratta di un'azione interregionale. Per ulteriori informazioni sulle operazioni tra regioni, consulta Aggiungere un'azione interregionale in CodePipeline.

CodePipeline utilizza metodi di rilevamento delle modifiche per avviare la pipeline quando viene effettuata una modifica al codice sorgente. Questi metodi di rilevamento sono basati sul tipo di origine:

- CodePipeline utilizza Amazon CloudWatch Events per rilevare le modifiche nel tuo repository di CodeCommit origine o nel tuo bucket di origine Amazon S3.

> **Note**
>
> Le risorse per il rilevamento delle modifiche vengono create automaticamente al momento dell'utilizzo della console. Quando utilizzi la console per creare o modificare una pipeline, le risorse aggiuntive vengono create per tuo conto. Se utilizzi il AWS CLI per creare la pipeline, devi creare tu stesso le risorse aggiuntive. Per ulteriori informazioni sulla creazione o l'aggiornamento di una CodeCommit pipeline, consulta. Creare una EventBridge regola per un' CodeCommit origine (CLI) Per ulteriori informazioni sull'utilizzo della CLI per creare o aggiornare una pipeline Amazon S3, consulta. Crea una EventBridge regola per un codice sorgente Amazon S3 (CLI)

Argomenti

- Modifica di una pipeline (console)
- Modifica di una pipeline (AWS CLI)

Modifica di una pipeline (console)

Puoi utilizzare la CodePipeline console per aggiungere, modificare o rimuovere fasi in una pipeline e per aggiungere, modificare o rimuovere azioni in una fase.

Quando aggiorni una pipeline, CodePipeline completa correttamente tutte le azioni in esecuzione e quindi fallisce le fasi e le esecuzioni della pipeline in cui sono state completate le azioni in esecuzione. Quando una pipeline viene aggiornata, è necessario eseguirla nuovamente. Per ulteriori informazioni sull'esecuzione di una pipeline, consulta. Avvio manuale di una pipeline

Per modificare una pipeline

1. Accedere AWS Management Console e aprire la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegliere il nome della pipeline da modificare. Questa operazione apre una visualizzazione dettagliata della pipeline. compreso lo stato di ciascuna delle operazioni in ciascuna fase della pipeline.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. Per modificare il tipo di tubazione, scegliete Modifica nella scheda Modifica: proprietà della tubazione. Scegliete una delle seguenti opzioni, quindi scegliete Fine.

   - Le pipeline di tipo V1 hanno una struttura JSON che contiene parametri standard di pipeline, stage e a livello di azione.
   - Le pipeline di tipo V2 hanno la stessa struttura di un tipo V1, oltre al supporto di parametri aggiuntivi, come trigger e variabili a livello di pipeline.

   I tipi di tubazioni differiscono per caratteristiche e prezzo. Per ulteriori informazioni, consulta Tipi di pipeline.
5. Per modificare le variabili della pipeline, scegliete Modifica variabili nella scheda Modifica: variabili. Aggiungi o modifica le variabili per il livello di pipeline, quindi scegli Fine.

   Per ulteriori informazioni sulle variabili a livello di pipeline, consulta. Variables Per un tutorial con una variabile a livello di pipeline che viene passata al momento dell'esecuzione della pipeline, consulta. Tutorial: utilizzare le variabili a livello di pipeline

> **Note**
>
> Sebbene sia facoltativo aggiungere variabili a livello di pipeline, per una pipeline specificata con variabili a livello di pipeline in cui non vengono forniti valori, l'esecuzione della pipeline avrà esito negativo.

6. Per modificare i trigger della pipeline, scegli Modifica trigger nella scheda Modifica: Triggers. Aggiungi o modifica i trigger, quindi scegli Fine.

   Per ulteriori informazioni sull'aggiunta di trigger, consulta i passaggi per creare una connessione a Bitbucket Cloud, GitHub (versione 2), GitHub Enterprise Server, GitLab .com o GitLab autogestita, ad esempio. GitHub connessioni

7. Per modificare le fasi e le azioni nella pagina Modifica, esegui una delle seguenti operazioni:

   - Per modificare una fase, scegliere Edit stage (Modifica fase). Puoi aggiungere operazioni in serie o in parallelo con le operazioni esistenti:

     In questa visualizzazione puoi anche modificare le operazioni scegliendo l'icona di modifica associata a tali operazioni. Per eliminare un'operazione, scegliere l'icona di eliminazione su tale operazione.

   - Per modificare un'operazione, scegliere l'icona di modifica per tale operazione e quindi modificare i valori in Edit action (Modifica operazione). Gli elementi contrassegnati con un asterisco (*) sono obbligatori.
     - Per il nome e il ramo di un CodeCommit repository, viene visualizzato un messaggio che mostra la regola Amazon CloudWatch Events da creare per questa pipeline. Se rimuovi la CodeCommit fonte, viene visualizzato un messaggio che mostra la regola di Amazon CloudWatch Events da eliminare.
     - Per un bucket di origine Amazon S3, viene visualizzato un messaggio che mostra la regola e il AWS CloudTrail percorso di Amazon CloudWatch Events da creare per questa pipeline. Se rimuovi la fonte Amazon S3, viene visualizzato un messaggio che mostra la regola e il AWS CloudTrail percorso di Amazon CloudWatch Events da eliminare. Se il AWS CloudTrail percorso è utilizzato da altre pipeline, il percorso non viene rimosso e l'evento relativo ai dati viene eliminato.
   - Per aggiungere una fase, scegliere + Add stage (+ Aggiungi fase) nel punto della pipeline in cui si desidera aggiungere una fase. Indicare un nome per la fase e quindi aggiungere a essa almeno un'operazione. Gli elementi contrassegnati con un asterisco (*) sono obbligatori.

- Per eliminare una fase, scegliere l'icona di eliminazione associata a tale fase. La fase e tutte le relative operazioni vengono eliminate.
- Per configurare una fase per il rollback automatico in caso di errore, scegli Modifica fase, quindi scegli la casella di controllo Configura il rollback automatico in caso di errore sullo stage.

Ad esempio, per aggiungere un'operazione seriale a una fase in una pipeline:

1. Nella fase in cui si desidera aggiungere l'operazione, scegliere Edit stage (Modifica fase), quindi selezionare + Add action group (Aggiungi gruppo di operazioni).
2. In Edit action (Modifica operazione), in Action name (Nome operazione), immettere il nome dell'operazione. L'elenco Action provider (Provider operazione) visualizza le opzioni provider per categoria. Cercare la categoria (ad esempio, Deploy (Distribuzione)). Nella categoria, scegli il provider (ad esempio, AWS CodeDeploy). In Regione, scegli la AWS regione in cui viene creata la risorsa o in cui intendi crearla. Il campo Regione indica dove vengono create le AWS risorse per questo tipo di azione e tipo di provider. Questo campo viene visualizzato solo per le azioni in cui il fornitore dell'azione è un Servizio AWS. Per impostazione predefinita, il campo Regione è la stessa AWS della pipeline.

   Per ulteriori informazioni sui requisiti per le azioni in CodePipeline, inclusi i nomi degli artefatti di input e output e su come vengono utilizzati, consulta. Requisiti della struttura delle azioni in CodePipeline Per esempi di aggiunta di provider di operazioni e di utilizzo di campi predefiniti per ogni provider, consulta Creazione di una pipeline (console).

   Per aggiungere CodeBuild come azione di compilazione o azione di test a una fase, consulta Use CodePipeline with CodeBuild to Test Code e Run Builds nella Guida per l'CodeBuild utente.

   > **Note**
   >
   > Alcuni provider di azioni, ad esempio GitHub, richiedono la connessione al sito Web del provider prima di poter completare la configurazione dell'azione. Quando si effettua la connessione a un sito web di un provider, utilizzare le credenziali proprie di tale sito web. Non utilizzare AWS le tue credenziali.

3. Una volta completata la configurazione dell'operazione, scegliere Save (Salva).

> **Note**
>
> Non è possibile rinominare una fase nella visualizzazione all'interno della console. È possibile aggiungere una fase con il nome che si desidera modificare, quindi eliminare la precedente. Verifica di aver aggiunto tutte le operazioni che desideri in tale fase prima di eliminare la precedente.

8. Al termine della modifica della pipeline, scegliere Save (Salva) per tornare alla pagina di riepilogo.

> **Important**
>
> Dopo aver salvato le modifiche, non è possibile annullarle. È necessario modificare di nuovo la pipeline. Se al momento del salvataggio delle modifiche è in esecuzione una versione tramite la pipeline, l'esecuzione non viene portata a termine. Se vuoi che un determinato commit o una modifica siano elaborati dalla pipeline modificata, devi avviare manualmente l'elaborazione tramite la pipeline. In caso contrario, saranno elaborati automaticamente dalla pipeline il commit o la modifica successivi.

9. Per testare la tua azione, scegli Release change per elaborare il commit attraverso la pipeline e confermare una modifica alla fonte specificata nella fase di origine della pipeline. Oppure segui i passaggi Avvio manuale di una pipeline per utilizzare il per AWS CLI rilasciare manualmente una modifica.

Modifica di una pipeline (AWS CLI)

Per modificare una pipeline utilizzare il comando update-pipeline.

Quando aggiorni una pipeline, CodePipeline completa correttamente tutte le azioni in esecuzione e quindi fallisce le fasi e le esecuzioni della pipeline in cui sono state completate le azioni in esecuzione. Quando una pipeline viene aggiornata, è necessario eseguirla nuovamente. Per ulteriori informazioni sull'esecuzione di una pipeline, consulta. Avvio manuale di una pipeline

> **Important**
>
> Sebbene sia possibile utilizzare la AWS CLI per modificare le pipeline che includono azioni partner, non è necessario modificare manualmente il JSON di un'azione partner. Facendolo, l'operazione del partner ha esito negativo dopo l'aggiornamento della pipeline.

Per modificare una pipeline

1. Apri una sessione terminale (Linux, macOS o Unix) o un prompt dei comandi (Windows) ed esegui il get-pipeline comando per copiare la struttura della pipeline in un file JSON. Ad esempio, per una pipeline denominata **`MyFirstPipeline`**, immettere il seguente comando:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   ```

   Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.

2. Aprire il file JSON in qualsiasi editor di testo e modificare la struttura del file in base alle modifiche che si desidera apportare alla pipeline. Ad esempio, è possibile aggiungere o rimuovere fasi, oppure aggiungerne un'altra operazione a una fase.

   L'esempio seguente mostra come aggiungere un'altra fase di distribuzione nel file pipeline.json. Questa fase viene eseguita dopo la prima fase di distribuzione denominata *`Staging`*.

   > **Note**
   >
   > Questa è solo una parte del file, non rappresenta l'intera struttura. Per ulteriori informazioni, consulta CodePipeline riferimento alla struttura della tubazione.

   ```
   ,
           {
               "name": "Staging",
               "actions": [
                       {
                           "inputArtifacts": [
                               {
                                   "name": "MyApp"
   ```

```
                        }
                    ],
                    "name": "Deploy-CodeDeploy-Application",
                    "actionTypeId": {
                        "category": "Deploy",
                        "owner": "AWS",
                        "version": "1",
                        "provider": "CodeDeploy"
                    },
                    "outputArtifacts": [],
                    "configuration": {
                        "ApplicationName": "CodePipelineDemoApplication",
                        "DeploymentGroupName": "CodePipelineDemoFleet"
                    },
                    "runOrder": 1
                }
            ]
        },
    {
        "name": "Production",
        "actions":  [
                {
                    "inputArtifacts": [
                        {
                            "name": "MyApp"
                        }
                    ],
                    "name": "Deploy-Second-Deployment",
                    "actionTypeId": {
                        "category": "Deploy",
                        "owner": "AWS",
                        "version": "1",
                        "provider": "CodeDeploy"
                    },
                    "outputArtifacts": [],
                    "configuration": {
                    "ApplicationName": "CodePipelineDemoApplication",
                    "DeploymentGroupName": "CodePipelineProductionFleet"
                    },
                    "runOrder": 1
                }
            ]
    }
]
```

```
}
```

Per informazioni su come usare l'interfaccia a riga di comando per aggiungere un'operazione di approvazione a una pipeline, consulta Aggiungi un'azione di approvazione manuale a una pipeline in CodePipeline .

Assicurati che il parametro `PollForSourceChanges` nel file JSON sia impostato come segue:

```
                "PollForSourceChanges": "false",
```

CodePipeline utilizza Amazon CloudWatch Events per rilevare le modifiche nel repository e nella filiale di CodeCommit origine o nel bucket di origine Amazon S3. Il passaggio successivo include le istruzioni per creare tali risorse in modo manuale. L'impostazione del flag su `false` disabilita i controlli periodici, che non sono necessari quando si utilizzano i metodi di rilevamento delle modifiche consigliati.

3. Per aggiungere un'operazione di compilazione, test o distribuzione in una regione differente rispetto a quella della pipeline, è necessario aggiungere i seguenti elementi alla struttura della pipeline. Per istruzioni dettagliate, vedi Aggiungere un'azione interregionale in CodePipeline.

   - Aggiungi il parametro `Region` alla struttura della pipeline dell'operazione.
   - Utilizza il parametro `artifactStores` per specificare un bucket di artefatti per ogni regione in cui disponi di un'operazione.

4. Se stai utilizzando la struttura della pipeline recuperata utilizzando il comando get-pipeline, questa deve essere modificata nel file JSON. Rimuovi le righe `metadata` dal file per consentire al comando update-pipeline di utilizzarlo. Rimuovere la sezione dalla struttura della pipeline nel file JSON (le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`)

   Ad esempio, rimuovere dalla struttura le seguenti righe:

```
"metadata": {
  "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
  "created": "date",
  "updated": "date"
  }
```

   Salvare il file.

5. Se si utilizza l'interfaccia a riga di comando per modificare una pipeline, occorre gestire manualmente le risorse di rilevamento delle modifiche consigliate per la pipeline:

   - Per un CodeCommit repository, devi creare la regola CloudWatch Events, come descritto in. Creare una EventBridge regola per un' CodeCommit origine (CLI)
   - Per una fonte Amazon S3, devi creare la regola CloudWatch Events e il AWS CloudTrail trail, come descritto in. Azioni di origine di Amazon S3 e con EventBridge AWS CloudTrail

6. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline:

   > **Important**
   >
   > Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

   ```
   aws codepipeline update-pipeline --cli-input-json file://pipeline.json
   ```

   Questo comando restituisce l'intera struttura della pipeline modificata.

   > **Note**
   >
   > Il comando update-pipeline arresta la pipeline. Se è in corso di elaborazione una versione durante l'esecuzione del comando update-pipeline, tale elaborazione viene arrestata. Per eseguire tale versione nella pipeline aggiornata, avviare la pipeline manualmente.

7. Apri la CodePipeline console e scegli la pipeline che hai appena modificato.

   La pipeline mostra le modifiche. La prossima volta che si modifica la sorgente originale, la pipeline elabora tale versione attraverso la struttura modificata delle pipeline stessa.

8. Per elaborare manualmente l'ultima versione attraverso la pipeline con la struttura aggiornata, eseguire il comando start-pipeline-execution. Per ulteriori informazioni, consulta Avvio manuale di una pipeline.

Per ulteriori informazioni sulla struttura di una pipeline e sui valori previsti, consulta CodePipeline riferimento alla struttura della tubazione e AWS CodePipeline API Reference.

Visualizza le pipeline e i dettagli in CodePipeline

Puoi utilizzare la AWS CodePipeline console o il AWS CLI per visualizzare i dettagli sulle pipeline associate al tuo AWS account.

Argomenti

Visualizza le pipeline (console)

È possibile visualizzare gli aggiornamenti dello stato, delle transizioni e degli artefatti per una pipeline.

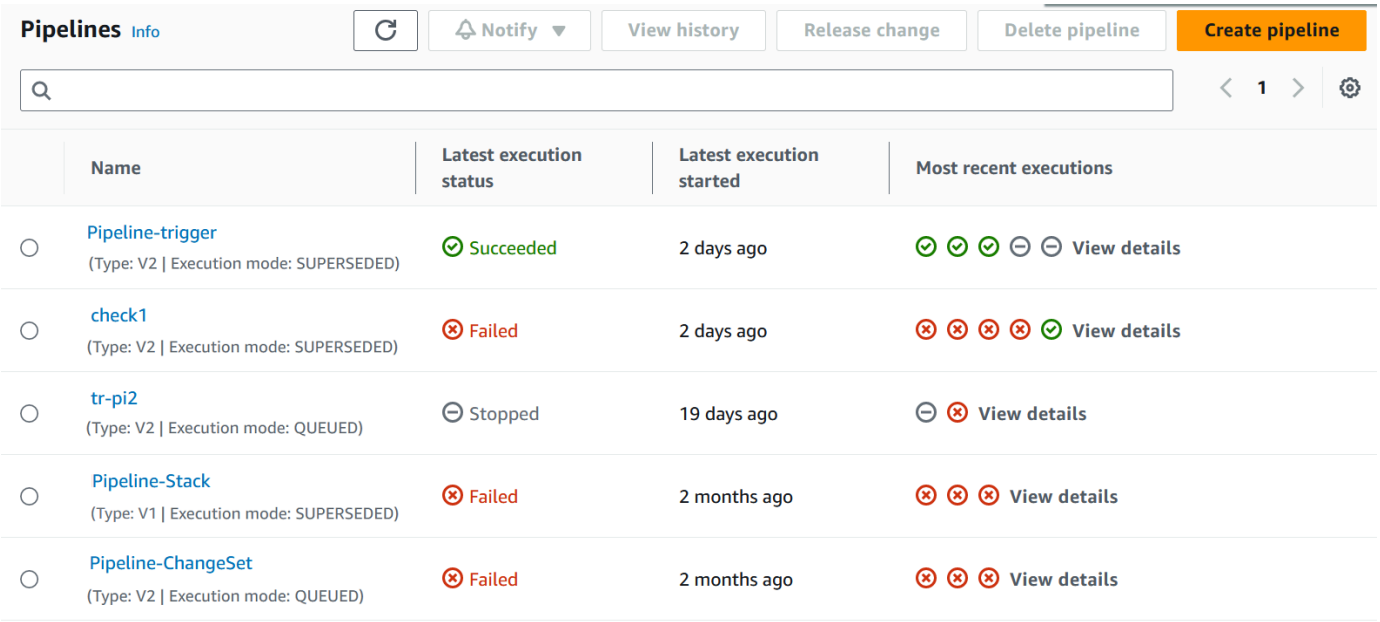
> **Note**
>
> Dopo un'ora, la visualizzazione dettagliata di una pipeline non viene più aggiornata automaticamente nel browser. Per visualizzare le informazioni correnti, aggiorna la pagina.

Per visualizzare le condutture

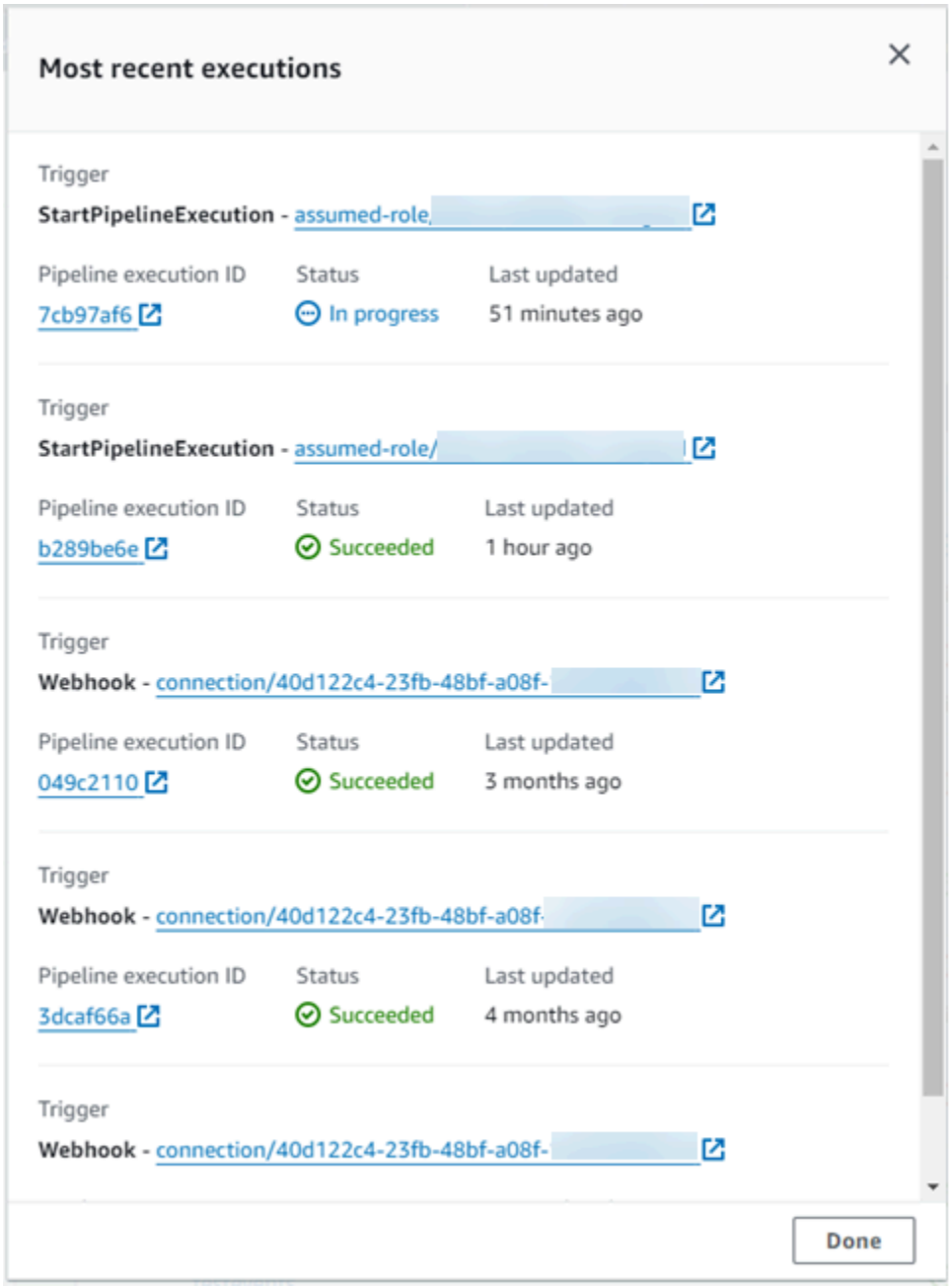
1. Accedere AWS Management Console e aprire la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Viene visualizzata la pagina Pipelines. Viene visualizzato un elenco di tutte le pipeline per quella regione.

   Vengono visualizzati il nome, il tipo, lo stato, la versione, la data di creazione e la data dell'ultima modifica di tutte le pipeline associate all' AWS account, insieme all'ora di esecuzione iniziata più di recente.
2. Viene visualizzato lo stato delle cinque esecuzioni più recenti.



Scegliete Visualizza dettagli accanto a una riga specifica per visualizzare una finestra di dialogo dei dettagli che elenca le esecuzioni più recenti.

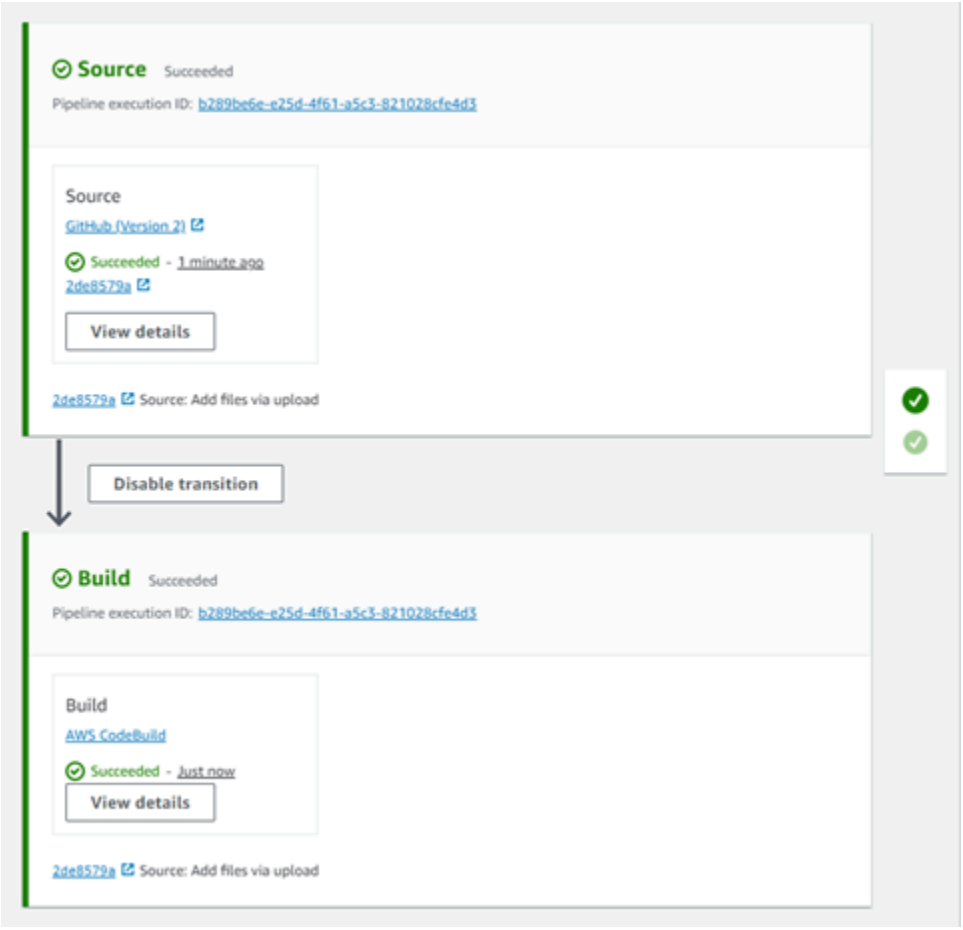


Per visualizzare i dettagli relativi alle esecuzioni più recenti per la pipeline, scegliere View history (Visualizza cronologia). Per esecuzioni precedenti, è possibile visualizzare i dettagli di revisione associati agli artefatti di origine, ad esempio ID di esecuzione, stato, orari di inizio e di fine, durata, nonché ID commit e messaggi.

> **Note**
>
> Per una pipeline in modalità di esecuzione PARALLEL, la vista principale della pipeline non mostra la struttura della pipeline o le esecuzioni in corso. Per una pipeline in

modalità di esecuzione PARALLEL, si accede alla struttura della pipeline scegliendo l'ID per l'esecuzione che si desidera visualizzare dalla pagina della cronologia di esecuzione. Scegli Cronologia nella barra di navigazione a sinistra, scegli l'ID di esecuzione per l'esecuzione parallela, quindi visualizza la pipeline nella scheda Visualizzazione.

3. Per visualizzare i dettagli di una singola pipeline, in Name (Nome), scegliere la pipeline. Viene visualizzata una vista dettagliata della pipeline, incluso lo stato di ciascuna operazione in ogni fase e lo stato delle transizioni.



La visualizzazione grafica contiene le seguenti informazioni per ciascuna fase:

- Il nome della fase.
- Ogni operazione configurata per la fase.
- Lo stato delle transizioni tra fasi (abilitato o disabilitato), come indicato dallo stato della freccia tra fasi. Una transizione abilitata è indicata da una freccia con accanto un pulsante Disable transition (Disabilita transizione). Una transizione disabilitata è indicata da una freccia con un sotto testo barrato e un pulsante Enable transition (Abilita transizione) accanto a essa.

- Una barra di colore per indicare lo stato della fase:
  - Grigio: ancora nessuna esecuzione
  - Blu: in corso
  - Verde: riuscita
  - Rosso non riuscita

La visualizzazione grafica contiene anche le seguenti informazioni relative alle operazioni in ciascuna fase:

- Il nome dell'operazione.
- Il fornitore dell'azione, ad esempio. CodeDeploy
- La data dell'ultima esecuzione dell'operazione.
- Se l'operazione è riuscita o non è riuscita.
- I collegamenti ad altri dettagli relativi all'ultima esecuzione dell'operazione, ove disponibili.
- Dettagli sulle revisioni dell'origine in corso durante l'ultima esecuzione della pipeline nella fase o, per le distribuzioni, sulle revisioni dell'origine più recenti che sono state CodeDeploy distribuite nelle istanze di destinazione.
- Un pulsante Visualizza dettagli che apre una finestra di dialogo con dettagli sull'esecuzione dell'azione, i log e la configurazione dell'azione.

> **Note**
>
> La scheda Registri è disponibile per AWS CloudFormation tutte CodeBuild le azioni eseguite nell'account della pipeline.

4. Per visualizzare i dettagli del provider dell'operazione, scegliere il provider. Ad esempio, nella pipeline dell'esempio precedente, se si sceglie CodeDeploy nelle fasi di gestione temporanea o di produzione, viene visualizzata la pagina della CodeDeploy console per il gruppo di distribuzione configurato per quella fase.
5. Per vedere lo stato di avanzamento di un'azione viene visualizzato accanto a un'azione in corso (indicata da un messaggio In corso). Se l'operazione è in corso, è possibile visualizzare l'avanzamento incrementale e le fasi o le operazioni mentre si verificano.
6. Per approvare o rifiutare operazioni che sono state configurate per l'approvazione manuale, scegliere Review (Rivedi).

7. Per riprovare operazioni in una fase che non sono state completate come previsto, scegliere Retry (Riprova).
8. Viene visualizzato lo stato dell'ultima volta in cui l'azione è stata eseguita, inclusi i risultati dell'azione (Riuscita o Non riuscita).

Visualizza i dettagli dell'azione in una pipeline (console)

Puoi visualizzare i dettagli di una pipeline, inclusi i dettagli delle azioni in ogni fase.

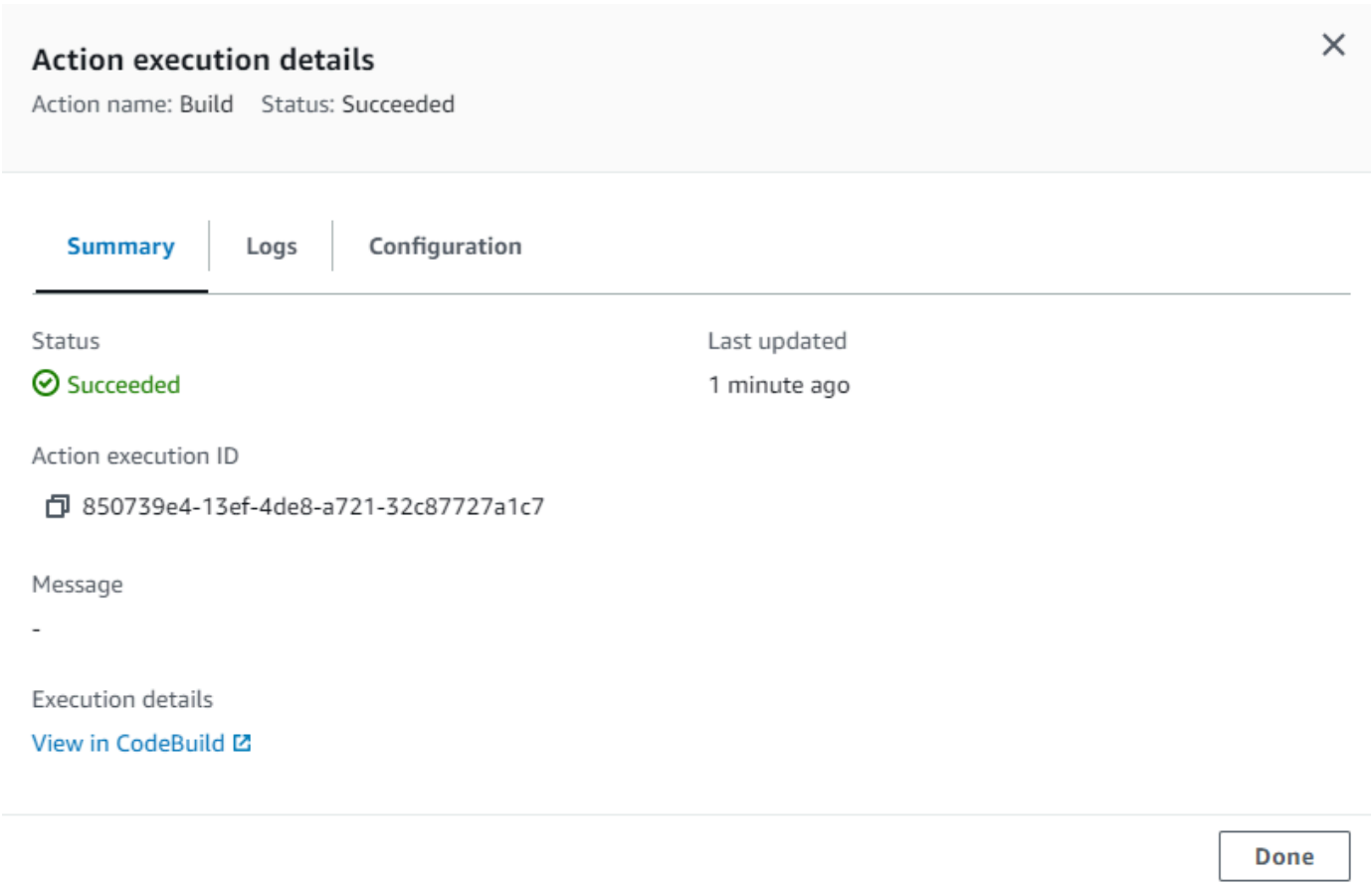
> **Note**
>
> Dopo un'ora, la visualizzazione dettagliata di una pipeline non viene più aggiornata automaticamente nel browser. Per visualizzare le informazioni correnti, aggiorna la pagina.

Per visualizzare i dettagli delle azioni in una pipeline

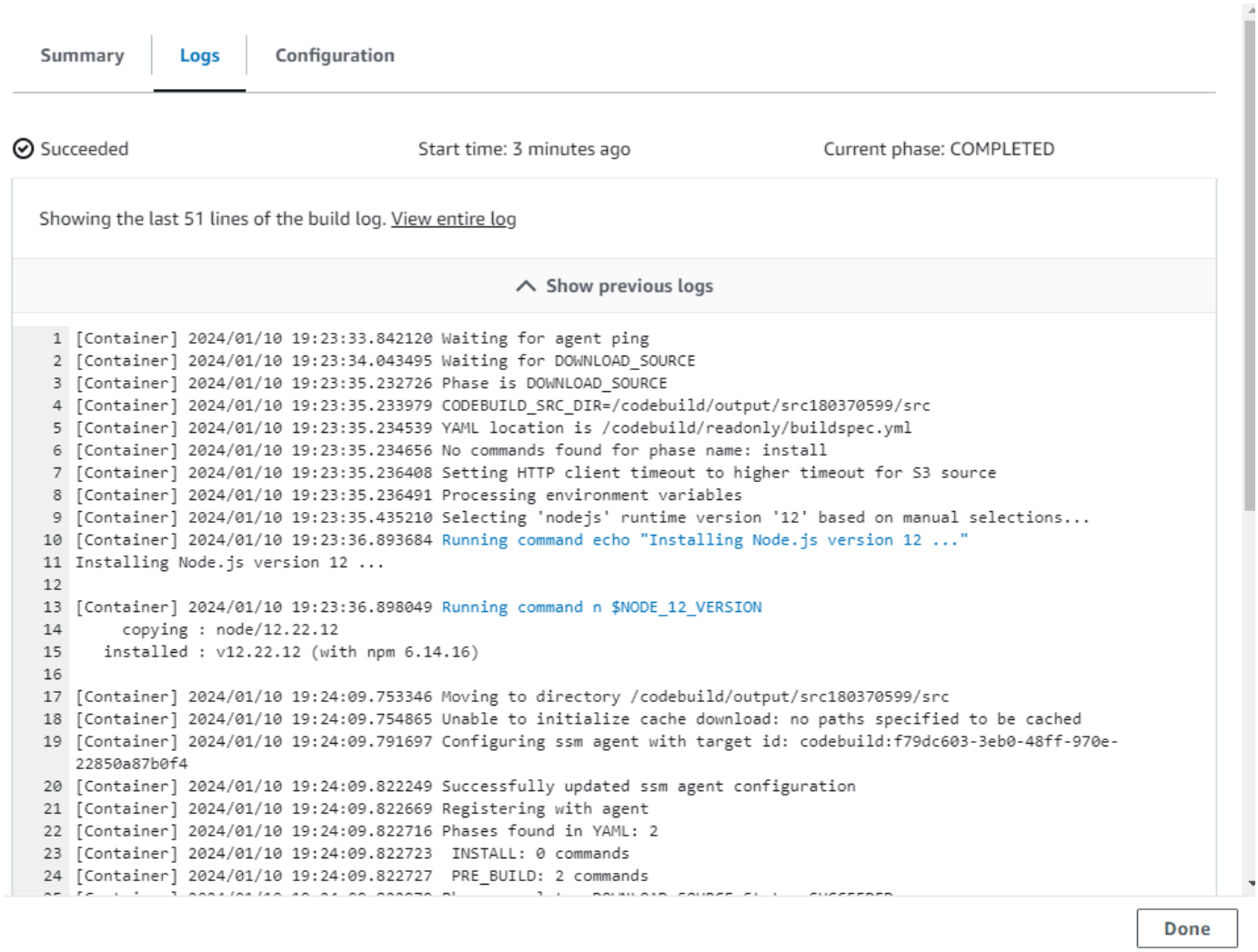
1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Viene visualizzata la pagina Pipelines.
2. Per qualsiasi azione, scegliete Visualizza dettagli per aprire una finestra di dialogo con dettagli sull'esecuzione dell'azione, i registri e la configurazione dell'azione.

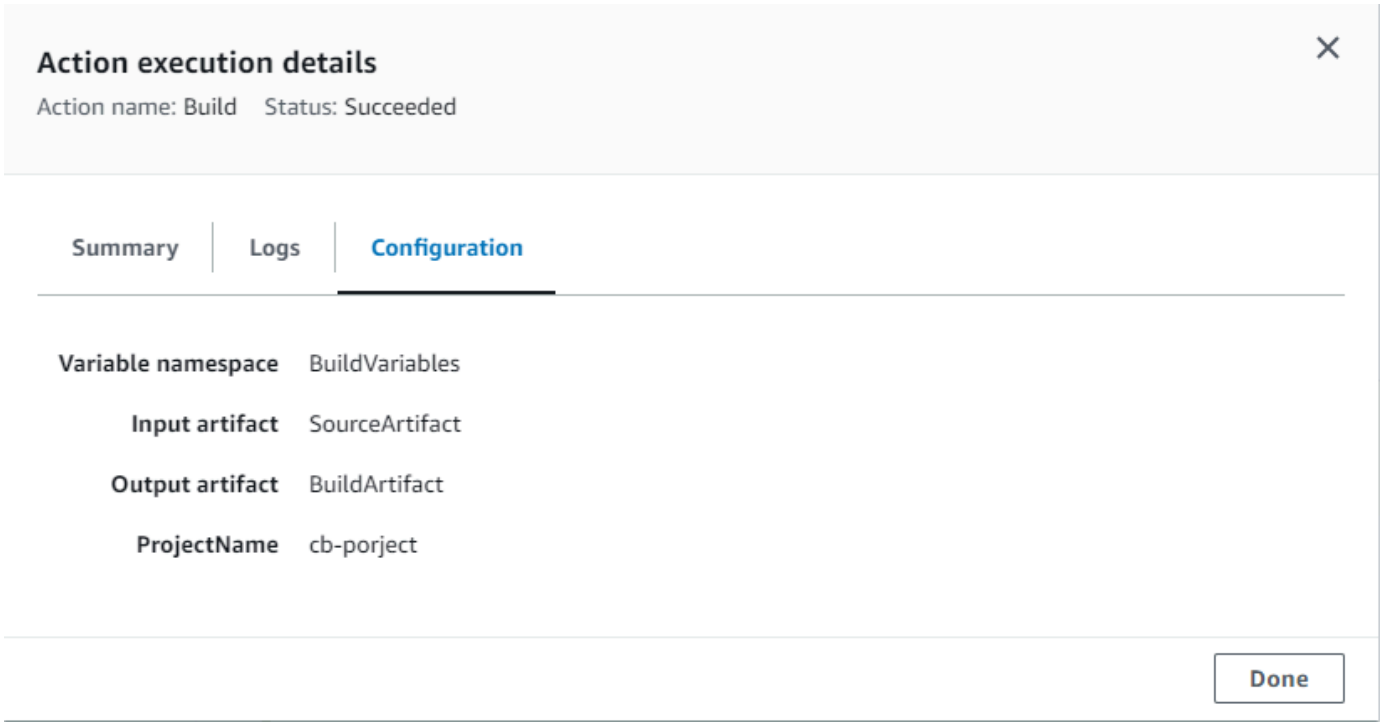
   > **Note**
   >
   > La scheda Registri è disponibile per le azioni CodeBuild e AWS CloudFormation le azioni.
3. Per visualizzare il riepilogo dell'azione relativa a un'azione in una fase di una pipeline, scegli Visualizza dettagli sull'azione, quindi scegli la scheda Riepilogo.



4. Per visualizzare i registri delle azioni relativi a un'azione con registri, scegli Visualizza dettagli sull'azione, quindi scegli la scheda Registri.



5. Per visualizzare i dettagli di configurazione di un'azione, scegli la scheda Configurazione.



Visualizza l'ARN della pipeline e l'ARN del ruolo di servizio (console)

È possibile utilizzare la console per visualizzare le impostazioni della pipeline, ad esempio l'ARN della pipeline, l'ARN del ruolo di servizio e l'archivio degli artifatti della pipeline.

1. Accedi e apri la console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home. AWS Management Console CodePipeline

   Verranno visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. Scegli il nome della pipeline, quindi scegli Impostazioni nel riquadro di navigazione a sinistra. La pagina mostra quanto segue:

   - Il nome della pipeline
   - La pipeline Amazon Resource Name (ARN)

     Il formato dell'ARN della pipeline è il seguente:

     arn:aws:codepipeline: regione: account: nome-pipeline

     ARN della pipeline di esempio:

     `arn:aws:codepipeline:us-east-2:80398EXAMPLE:MyFirstPipeline`

- L'ARN del ruolo di CodePipeline servizio per la tua pipeline
- La versione pipeline
- Il nome e la posizione dell'archivio degli artefatti per la pipeline

Visualizzazione dei dettagli e della cronologia della pipeline (CLI)

Puoi eseguire i seguenti comandi per visualizzare i dettagli relativi alla pipeline e alle esecuzioni della pipeline:

- list-pipelinescomando per visualizzare un riepilogo di tutte le pipeline associate al tuo account. AWS
- Comando get-pipeline per rivedere i dettagli di una singola pipeline.
- list-pipeline-executions per visualizzare i riepiloghi delle esecuzioni più recenti per una pipeline.
- get-pipeline-execution per visualizzare informazioni relative a un'esecuzione di una pipeline, inclusi i dettagli sugli artefatti, l'ID di esecuzione della pipeline, nonché il nome, la versione e lo stato della pipeline.
- Comando get-pipeline-state per visualizzare pipeline, fase e stato dell'operazione.
- list-action-executions per visualizzare i dettagli di esecuzione delle operazioni per una pipeline.

1. Apri un terminale (Linux, macOS o Unix) o un prompt dei comandi (Windows) e usali AWS CLI per eseguire il comando: list-pipelines

   ```
   aws codepipeline list-pipelines
   ```

   Questo comando restituisce un elenco di tutte le pipeline associate all'account AWS .

2. Per visualizzare i dettagli relativi a una pipeline, eseguire il comando get-pipeline, specificando il nome univoco della pipeline. Ad esempio, per visualizzare i dettagli su una pipeline denominata *MyFirstPipeline*, inserisci quanto segue:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline
   ```

   Questo comando restituisce la struttura della pipeline.

Eliminare una tubazione in CodePipeline

Puoi sempre modificare una pipeline per variare le sue funzionalità, ma potresti invece decidere di eliminarla. È possibile utilizzare la AWS CodePipeline console o il delete-pipeline comando in AWS CLI per eliminare una pipeline.

Argomenti

Eliminazione di una pipeline (console)

Per eliminare una pipeline

1. Accedere AWS Management Console e aprire la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi e lo stato di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegliere il nome della pipeline da eliminare.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. Nella pagina Edit (Modifica), scegliere Delete (Elimina).
5. Digitare **`delete`** nel campo per confermare e quindi scegliere Delete (Elimina).

> **Important**
>
> Questa operazione non può essere annullata.

Eliminazione di una pipeline (CLI)

Per utilizzare l'eliminazione manuale AWS CLI di una pipeline, utilizzate il comando delete-pipeline.

> **Important**
>
> L'eliminazione di una pipeline è irreversibile. Non è prevista una finestra di dialogo di conferma. Dopo l'esecuzione del comando, la pipeline viene eliminata, ma non viene

eliminata nessuna delle risorse utilizzate dalla pipeline. In questo modo è più semplice creare una nuova pipeline che utilizza tali risorse per automatizzare il rilascio del software.

Per eliminare una pipeline

1. Apri un terminale (Linux, macOS o Unix) o il prompt dei comandi (Windows) e usa AWS CLI per eseguire il delete-pipeline comando, specificando il nome della pipeline che desideri eliminare. Ad esempio, per eliminare una pipeline denominata: *MyFirstPipeline*

   ```
   aws codepipeline delete-pipeline --name MyFirstPipeline
   ```

   Questo comando non restituisce alcun risultato.

2. Eliminare tutte le risorse non più necessarie.

   > **Note**
   >
   > L'eliminazione di una pipeline non elimina le risorse utilizzate nella pipeline, come l'applicazione Elastic CodeDeploy Beanstalk che hai usato per distribuire il codice o, se hai creato la pipeline dalla console, il bucket Amazon S3 creato per archiviare gli artefatti CodePipeline delle tue pipeline. CodePipeline Assicurati di eliminare le risorse che non sono più necessarie in modo che in futuro non venga addebitato alcun costo per il loro uso. Ad esempio, quando usi la console per creare una pipeline per la prima volta, CodePipeline crea un bucket Amazon S3 per archiviare tutti gli artefatti per tutte le tue pipeline. Se hai eliminato tutte le pipeline, segui i passaggi descritti in Eliminazione di un bucket.

Crea una pipeline CodePipeline che utilizzi le risorse di un altro account AWS

Potrebbe essere necessario creare una pipeline che utilizza le risorse create o gestite da un altro account AWS . Ad esempio, potresti voler utilizzare un account per la tua pipeline e un altro per le tue CodeDeploy risorse.

> **Note**
>
> Quando si crea una pipeline con operazioni da più account, è necessario configurare le operazioni in modo che possano ancora accedere agli artefatti entro i limiti delle pipeline tra account. Le seguenti limitazioni si applicano alle operazioni tra account:
>
> - In generale, un'operazione può consumare solo un artefatto se:
>   - L'operazione è nello stesso account dell'account della pipeline OPPURE
>   - L'artefatto è stato creato nell'account della pipeline per un'operazione in un altro account OPPURE
>   - L'artefatto è stato prodotto da un'operazione precedente nello stesso account dell'azione
>
>   In altre parole, non è possibile passare un artefatto da un account a un altro se nessun account è l'account della pipeline.
> - Le operazioni tra account non sono supportate per i seguenti tipi di operazione:
>   - Operazioni di compilazione Jenkins

Per questo esempio, è necessario creare una chiave AWS Key Management Service (AWS KMS) da utilizzare, aggiungere la chiave alla pipeline e configurare le politiche e i ruoli degli account per consentire l'accesso tra account. Per una chiave AWS KMS, puoi utilizzare l'ID chiave, la chiave ARN o l'alias ARN.

> **Note**
>
> Gli alias sono riconosciuti solo nell'account che ha creato la chiave KMS. Per le operazioni tra account, puoi utilizzare solo l'ID della chiave o l'ARN della chiave per identificare la chiave. Le operazioni tra account comportano l'utilizzo del ruolo dell'altro account (AccountB), pertanto specificando l'ID chiave verrà utilizzata la chiave dell'altro account (AccountB).

In questa procedura dettagliata e nei relativi esempi, *`AccountA`* è l'account originariamente utilizzato per creare la pipeline. Ha accesso al bucket Amazon S3 utilizzato per archiviare gli artefatti della pipeline e al ruolo di servizio utilizzato da. AWS CodePipeline*`AccountB`* è l'account originariamente utilizzato per creare l' CodeDeploy applicazione, il gruppo di distribuzione e il ruolo di servizio utilizzati da. CodeDeploy

Affinché AccountA possa modificare una pipeline per utilizzare l' CodeDeploy applicazione creata da AccountB, AccountA deve:

- Richiedere l'ARN o l'ID account di *AccountB* (in questa procedura dettagliata, l'ID *AccountB* è *012ID_ACCOUNT_B*).
- *Crea o utilizza una chiave gestita AWS KMS dal cliente nella regione per la pipeline e concedi le autorizzazioni per utilizzare tale chiave per il ruolo di servizio (CodePipeline_Service_Role) e AccountB.*
- *Crea una policy per i bucket Amazon S3 che conceda all'AccountB l'accesso al bucket Amazon S3 (ad esempio, -2-1234567890). codepipeline-us-east*
- *Crea una policy che consenta all'AccountA di assumere un ruolo configurato da AccountB e allega tale policy al ruolo di servizio (_Service_Role). CodePipeline*
- Modifica la pipeline per utilizzare la chiave gestita AWS KMS dal cliente anziché la chiave predefinita.

Affinché *AccountB* possa consentire a una pipeline creata in *AccountA* di accedere alle sue risorse, *AccountB* deve:

- Richiedere l'ARN o l'ID account *AccountA* (in questa procedura dettagliata, l'ID *AccountA* è *012ID_ACCOUNT_A*).
- *Crea una policy applicata al ruolo dell'istanza Amazon EC2 configurato per CodeDeploy che consenta l'accesso al codepipeline-us-east bucket Amazon S3 (-2-1234567890).*
- *Crea una policy applicata al ruolo dell'istanza Amazon EC2 configurato per CodeDeploy che consenta l'accesso alla chiave gestita dal AWS KMS cliente utilizzata per crittografare gli artefatti della pipeline in AccounTa.*
- Configura e associa un ruolo IAM (*CrossAccount_Role*) con una politica di relazione di fiducia che consenta al ruolo di CodePipeline servizio in *AccountA* di assumere il ruolo.
- *Crea una policy che consenta l'accesso alle risorse di distribuzione richieste dalla pipeline e collegala a _Role. CrossAccount*
- *Crea una policy che consenta l'accesso al bucket Amazon S3 (codepipeline-us-east-2-1234567890) e collegalo a _Role. CrossAccount*

Argomenti

Prerequisito: creare una chiave di crittografia AWS KMS

Le chiavi gestite dal cliente sono specifiche di una regione, così come tutte le chiavi. AWS KMS È necessario creare la AWS KMS chiave gestita dal cliente nella stessa regione in cui è stata creata la pipeline (ad esempio,). `us-east-2`

Per creare una chiave gestita dal cliente in AWS KMS

1. Accedi a AWS Management Console con *`AccountA`* e apri la AWS KMS console.
2. A sinistra, scegli Chiavi gestite dal cliente.
3. Scegliere Create key (Crea chiave). In Configura chiave, lasciare selezionata l'impostazione predefinita simmetrica e scegliere Avanti.
4. *In **`Alias`**, inserisci un alias da utilizzare per questa chiave (ad esempio, `PipelineName -Key`).* Facoltativamente, fornire una descrizione per questa chiave e scegliere Fase successiva.
5. In Definisci le autorizzazioni amministrative chiave, scegli il ruolo o i ruoli che desideri ricoprire come amministratori per questa chiave, quindi scegli Avanti.
6. In Definisci le autorizzazioni di utilizzo delle chiavi, in Questo account, seleziona il nome del ruolo di servizio per la pipeline (ad esempio, _Service_Role). CodePipeline In Altri AWS account, scegli Aggiungi un altro account. AWS Digita l'ID account per *`AccountB`* per completare l'ARN, quindi scegli Successivo.
7. In Visualizza un'anteprima della policy della chiave, rivedere la policy e quindi scegliere Fine.
8. Dall'elenco di chiavi, scegliere l'alias della chiave e copiare il relativo ARN (ad esempio, ***`arn:aws:kms:us-east-2:012ID_ACCOUNT_A:key/2222222-3333333-4444-556677EXAMPLE`***). Ciò è richiesto quando si modifica la pipeline e si configurano le policy.

Fase 1: impostazione delle policy e dei ruoli dell'account

Dopo aver creato la AWS KMS chiave, devi creare e allegare politiche che consentano l'accesso da più account. Ciò richiede operazioni da *AccountA* e *AccountB*.

Argomenti

- Configurazione di policy e ruoli nell'account che crea la pipeline (AccountA)
- Configurare politiche e ruoli nell'account proprietario della AWS risorsa (AccountB)

Configurazione di policy e ruoli nell'account che crea la pipeline (***AccountA***)

Per creare una pipeline che utilizzi CodeDeploy risorse associate a un altro AWS account, *AccounTA* deve configurare le policy sia per il bucket Amazon S3 utilizzato per archiviare gli artefatti sia per il ruolo del servizio. CodePipeline

Per creare una policy per il bucket Amazon S3 che conceda l'accesso a AccountB (console)

1. Accedi a AWS Management Console con *Accounta* e apri la console Amazon S3 all'indirizzo https://console.aws.amazon.com/s3/.
2. Nell'elenco dei bucket Amazon S3, scegli il bucket Amazon S3 in cui sono archiviati gli elementi per le tue pipeline. *Questo bucket ha un nomecodepipeline-region-1234567EXAMPLE, dove region è la AWS regione in cui hai creato la pipeline e 1234567EXAMPLE è un numero casuale a dieci cifre che assicura che il nome del bucket sia univoco (ad esempio, -2-1234567890). codepipeline-us-east*
3. Nella pagina dei dettagli del bucket Amazon S3, scegli Proprietà.
4. Nel riquadro delle proprietà, espandere Permissions (Autorizzazioni), quindi scegliere Add bucket policy (Aggiungi policy di bucket).

> **Note**
>
> Se una policy è già associata al tuo bucket Amazon S3, scegli Modifica policy bucket. È possibile quindi aggiungere le dichiarazioni nell'esempio seguente alla policy esistente. Per aggiungere una nuova politica, scegli il link e segui le istruzioni nel Policy Generator. AWS Per ulteriori informazioni, consulta Panoramica delle politiche IAM.

5. Nella finestra Bucket Policy Editor (Editor policy di bucket), digitare la seguente policy. Questo consente ad *AccountB* di accedere agli artefatti pipeline e offre ad *AccountB* la possibilità di aggiungere artefatti di output se creati da un'operazione, ad esempio un'operazione di origine o di compilazione personalizzata.

   Nel seguente esempio, l'ARN per *AccountB* è *012ID_ACCOUNT_B*. *L'ARN per il bucket Amazon S3 è -2-1234567890. codepipeline-us-east* Sostituisci questi ARN con l'ARN per l'account a cui desideri consentire l'accesso e l'ARN per il bucket Amazon S3:

```
{
 "Version": "2012-10-17",
 "Id": "SSEAndSSLPolicy",
 "Statement": [
{
  "Sid": "DenyUnEncryptedObjectUploads",
  "Effect": "Deny",
  "Principal": "*",
  "Action": "s3:PutObject",
  "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*",
  "Condition": {
 "StringNotEquals": {
 "s3:x-amz-server-side-encryption": "aws:kms"
 }
   }
 },
{
  "Sid": "DenyInsecureConnections",
  "Effect": "Deny",
  "Principal": "*",
  "Action": "s3:*",
  "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*",
  "Condition": {
 "Bool": {
    "aws:SecureTransport": false
  }
 }
  },
{
  "Sid": "",
  "Effect": "Allow",
  "Principal": {
 "AWS": "arn:aws:iam::012ID_ACCOUNT_B:root"
```

```
  },
   "Action": [
            "s3:Get*",
            "s3:Put*"
        ],
   "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*"
 },
 {
   "Sid": "",
   "Effect": "Allow",
   "Principal": {
       "AWS": "arn:aws:iam::012ID_ACCOUNT_B:root"
   },
   "Action": "s3:ListBucket",
   "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890"
 }
   ]
}
```

6. Scegliere Save (Salva), quindi chiudere l'editor di policy.
7. Scegli Salva per salvare le autorizzazioni per il bucket Amazon S3.

Per creare una policy per il ruolo di servizio per CodePipeline (console)

1. Accedi a AWS Management Console with *Accounta* e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel riquadro di navigazione, seleziona Ruoli.
3. Nell'elenco dei ruoli, in Nome ruolo, scegli il nome del ruolo di servizio per CodePipeline.
4. Nella scheda Permissions (Autorizzazioni) scegliere Add inline policy (Aggiungi policy inline).
5. Scegli la scheda JSON e inserisci la seguente politica per consentire a *AccountB* di assumere il ruolo. Nel seguente esempio, *012ID_ACCOUNT_B* è l'ARN per *AccountB*.

```
{
   "Version": "2012-10-17",
   "Statement": {
       "Effect": "Allow",
       "Action": "sts:AssumeRole",
       "Resource": [
           "arn:aws:iam::012ID_ACCOUNT_B:role/*"
       ]
```

```
    }
  }
```

6. Scegli Verifica policy.
7. In Name (Nome), immetti un nome per la policy. Scegli Crea policy.

Configurare politiche e ruoli nell'account proprietario della AWS risorsa (*AccountB*)

Quando crei un'applicazione, una distribuzione e un gruppo di distribuzione in CodeDeploy, crei anche un ruolo di istanza Amazon EC2. (Questo ruolo viene creato automaticamente se utilizzi la procedura dettagliata di distribuzione, ma puoi anche crearlo manualmente.) Affinché una pipeline creata in *AccountA* CodeDeploy utilizzi le risorse create in *AccountB*, devi:

- Configura una policy per il ruolo dell'istanza che le consenta di accedere al bucket Amazon S3 in cui sono archiviati gli artefatti della pipeline.
- Creare un secondo ruolo in *AccountB* configurato per l'accesso tra account.

Questo secondo ruolo non deve solo avere accesso al bucket Amazon S3 in AccountA, ma deve anche contenere una politica che consenta l'accesso alle CodeDeploy risorse e una politica di relazione di fiducia che consenta al ruolo di servizio in AccountA di assumere il CodePipeline ruolo.

> **Note**
>
> Queste politiche sono specifiche per la configurazione CodeDeploy delle risorse da utilizzare in una pipeline creata utilizzando un account diverso. AWS Altre AWS risorse richiederanno politiche specifiche in base ai rispettivi requisiti di risorse.

Per creare una policy per il ruolo dell'istanza Amazon EC2 configurato per CodeDeploy (console)

1. Accedi a AWS Management Console con *AccountB* e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel riquadro di navigazione, seleziona Ruoli.
3. Nell'elenco dei ruoli, in Nome ruolo, scegli il nome del ruolo di servizio utilizzato come ruolo dell'istanza Amazon EC2 per l' CodeDeploy applicazione. Questo nome ruolo può variare e un

gruppo di distribuzione può utilizzare più ruoli dell'istanza. Per ulteriori informazioni, consulta Creare un profilo di istanza IAM per le istanze Amazon EC2.

4. Nella scheda Permissions (Autorizzazioni) scegliere Add inline policy (Aggiungi policy inline).
5. *Scegli la scheda **JSON** e inserisci la seguente policy per concedere l'accesso al bucket Amazon S3 utilizzato da AccounTA per archiviare gli artefatti per le pipeline (in questo esempio, -2-1234567890): codepipeline-us-east*

```
{
   "Version": "2012-10-17",
   "Statement": [
     {
       "Effect": "Allow",
       "Action": [
         "s3:Get*"
       ],
       "Resource": [
         "arn:aws:s3:::codepipeline-us-east-2-1234567890/*"
       ]
     },
     {
       "Effect": "Allow",
       "Action": [
         "s3:ListBucket"
       ],
       "Resource": [
         "arn:aws:s3:::codepipeline-us-east-2-1234567890"
       ]
     }
   ]
 }
```

6. Scegli Verifica policy.
7. In Name (Nome), immetti un nome per la policy. Scegli Crea policy.
8. *Crea una seconda policy per sapere AWS KMS **arn:aws:kms:us-east-1:012ID_ACCOUNT_A:key/2222222-3333333-4444-556677EXAMPLE** dov'è l'ARN della chiave gestita dal cliente creata in AccountA e configurata per consentire a AccountB di utilizzarla:*

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
           "kms:DescribeKey",
           "kms:GenerateDataKey*",
           "kms:Encrypt",
           "kms:ReEncrypt*",
           "kms:Decrypt"
         ],
        "Resource": [
           "arn:aws:kms:us-
east-1:012ID_ACCOUNT_A:key/2222222-3333333-4444-556677EXAMPLE"
         ]
      }
   ]
}
```

> **Important**
>
> È necessario utilizzare l'ID account di *AccountA* in questa politica come parte dell'ARN della risorsa per la AWS KMS chiave, come mostrato qui, altrimenti la politica non funzionerà.

9. Scegli Verifica policy.
10. In Name (Nome), immetti un nome per la policy. Scegli Crea policy.

Ora crea un ruolo IAM da utilizzare per l'accesso su più account e configuralo in modo che il ruolo di CodePipeline servizio in *AccountA* possa assumere il ruolo. *Questo ruolo deve contenere politiche che consentano l'accesso alle CodeDeploy risorse e al bucket Amazon S3 utilizzato per archiviare gli artefatti in Accounta.*

Per configurare il ruolo tra account diversi in IAM

1. Accedi a AWS Management Console con *AccountB* e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam.
2. Nel pannello di navigazione, seleziona Roles (Ruoli). Selezionare Create role (Crea ruolo).

3. In Seleziona tipo di entità attendibile, scegli Un altro account AWS . In Specificare gli account che possono utilizzare questo ruolo, in ID account, inserisci l'ID AWS account per l'account che creerà la pipeline in CodePipeline (*AccountA*), quindi scegli Avanti: Autorizzazioni.

   > **Important**
   >
   > Questa fase consente di creare la policy di relazione di trust tra *AccountB* e *AccountA*. *Tuttavia, ciò garantisce l'accesso di livello root all'account e CodePipeline consiglia di ridurlo al ruolo di CodePipeline servizio in AccountA.* Segui il passaggio 16 per limitare le autorizzazioni.

4. In Allega politiche di autorizzazione, scegli AmazonS3 ReadOnlyAccess, quindi scegli Avanti: tag.

   > **Note**
   >
   > Questa non è la policy che verrà utilizzata. Scegliere una policy per completare la procedura guidata.

5. Scegli Prossimo: Rivedi. *Digita un nome per questo ruolo in Nome ruolo (ad esempio, **_Role**). CrossAccount* Puoi assegnare a questo ruolo il nome che preferisci purché segua le convenzioni di denominazione in IAM. Valutare se assegnare al ruolo un nome che ne indica chiaramente lo scopo. Selezionare Crea ruolo.

6. Dall'elenco dei ruoli, scegli il ruolo che hai appena creato (ad esempio, *CrossAccount_Role) per aprire la pagina di **riepilogo** relativa a quel ruolo*.

7. Nella scheda Permissions (Autorizzazioni) scegliere Add inline policy (Aggiungi policy inline).

8. Scegli la scheda JSON e inserisci la seguente politica per consentire l'accesso alle CodeDeploy risorse:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codedeploy:CreateDeployment",
        "codedeploy:GetDeployment",
        "codedeploy:GetDeploymentConfig",
```

```
        "codedeploy:GetApplicationRevision",
        "codedeploy:RegisterApplicationRevision"
      ],
      "Resource": "*"
    }
  ]
}
```

9. Scegli Verifica policy.
10. In Name (Nome), immetti un nome per la policy. Scegli Crea policy.
11. Nella scheda Permissions (Autorizzazioni) scegliere Add inline policy (Aggiungi policy inline).
12. *Scegli la scheda **JSON** e inserisci la seguente policy per consentire a questo ruolo di recuperare gli artefatti di input e inserire gli artefatti di output nel bucket Amazon S3 in AccountA:*

```
{
   "Version": "2012-10-17",
   "Statement": [
     {
       "Effect": "Allow",
       "Action": [
         "s3:GetObject*",
         "s3:PutObject",
         "s3:PutObjectAcl"
       ],
       "Resource": [
         "arn:aws:s3:::codepipeline-us-east-2-1234567890/*"
       ]
     }
   ]
}
```

13. Scegli Verifica policy.
14. In Name (Nome), immetti un nome per la policy. Scegli Crea policy.
15. Nella scheda Autorizzazioni, trova AmazonS3 ReadOnlyAccess nell'elenco delle politiche in Nome della politica e scegli l'icona di eliminazione (X) accanto alla politica. Quando richiesto, scegliere Detach (Scollega).
16. Seleziona la scheda Relazione di fiducia, quindi scegli Modifica politica di fiducia. Scegli l'opzione Aggiungi un principale nella colonna di sinistra. *Per il **tipo Principal**, scegli **IAM Roles**, quindi fornisci l'ARN per il ruolo di CodePipeline servizio*

in AccountA. Rimuovi **arn:aws:iam::Account_A:root** dall'elenco per AWS Principal, quindi scegli Aggiorna policy.

Fase 2: modifica della pipeline

Non è possibile utilizzare la CodePipeline console per creare o modificare una pipeline che utilizza risorse associate a un altro AWS account. Tuttavia, è possibile utilizzare la console per creare la struttura generale della pipeline e quindi utilizzare la AWS CLI per modificare la pipeline e aggiungere tali risorse. In alternativa, puoi utilizzare la struttura di una pipeline esistente e aggiungere manualmente le risorse.

Per aggiungere le risorse associate a un altro AWS account ()AWS CLI

1. In un terminale (Linux, macOS o Unix) o nel prompt dei comandi (Windows), esegui il get-pipeline comando sulla pipeline a cui desideri aggiungere risorse. Copiare l'output del comando in un file JSON. Ad esempio, per una pipeline denominata MyFirstPipeline, è necessario digitare qualcosa di simile al seguente:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   ```

   L'output viene inviato al file *pipeline.json*.

2. Aprire il file JSON in qualsiasi editor di testo normale. Dopo essere `"type": "S3"` entrato nell'archivio degli artifact, aggiungi le informazioni KMS EncryptionKey, ID e type dove *codepipeline-us-east-2-1234567890* è il nome del bucket Amazon S3 utilizzato per archiviare gli artefatti per la pipeline ed è l'ARN della chiave gestita dal cliente che hai appena creato: ***arn:aws:kms:us-east-1:012ID_ACCOUNT_A:key/2222222-3333333-4444-556677EXAMPLE***

   ```
   {
     "artifactStore”: {
       "location": "codepipeline-us-east-2-1234567890",
       "type": "S3",
       "encryptionKey": {
         "id": "arn:aws:kms:us-
   east-1:012ID_ACCOUNT_A:key/2222222-3333333-4444-556677EXAMPLE",
         "type": "KMS"
       }
     },
   ```

3. *Aggiungi un'azione di distribuzione in una fase per utilizzare le CodeDeploy risorse associate a AccountB, inclusi i valori per roleArn il ruolo tra account che hai creato CrossAccount (_Role).*

   L'esempio seguente mostra JSON che aggiunge un'azione di distribuzione denominata. *ExternalDeploy Utilizza le CodeDeploy risorse create in AccountB in una fase denominata Staging.* Nel seguente esempio, l'ARN per *AccountB* è *012ID_ACCOUNT_B*:

```
,
            {
                "name": "Staging",
                "actions": [
                    {
                        "inputArtifacts": [
                            {
                                "name": "MyAppBuild"
                            }
                        ],
                        "name": "ExternalDeploy",
                        "actionTypeId": {
                            "category": "Deploy",
                            "owner": "AWS",
                            "version": "1",
                            "provider": "CodeDeploy"
                        },
                        "outputArtifacts": [],
                        "configuration": {
                            "ApplicationName": "AccountBApplicationName",
                            "DeploymentGroupName": "AccountBApplicationGroupName"
                        },
                        "runOrder": 1,
                        "roleArn":
 "arn:aws:iam::012ID_ACCOUNT_B:role/CrossAccount_Role"
                    }
                ]
            }
```

> **Note**
>
> Non si tratta del JSON per l'intera pipeline, ma solo della struttura per l'operazione in una fase.

4. Rimuovi le righe `metadata` dal file per consentire al comando update-pipeline di utilizzarlo. Rimuovere la sezione dalla struttura della pipeline nel file JSON (le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`)

   Ad esempio, rimuovere dalla struttura le seguenti righe:

   ```
   "metadata": {
     "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
     "created": "date",
     "updated": "date"
     }
   ```

   Salvare il file.

5. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline, in modo analogo al seguente:

   > **Important**
   >
   > Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

   ```
   aws codepipeline update-pipeline --cli-input-json file://pipeline.json
   ```

   Questo comando restituisce l'intera struttura della pipeline modificata.

Per testare la pipeline che utilizza risorse associate a un altro account AWS

1. In un terminale (Linux, macOS o Unix) o dal prompt dei comandi (Windows), esegui il start-pipeline-execution comando, specificando il nome della pipeline, in modo simile al seguente:

```
aws codepipeline start-pipeline-execution --name MyFirstPipeline
```

Per ulteriori informazioni, consulta Avvio manuale di una pipeline.

2. Accedi a AWS Management Console con *AccountA* e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.

3. In Name (Nome), scegliere il nome della pipeline modificata. Questa operazione apre una visualizzazione dettagliata della pipeline, che include lo stato di ciascuna operazione in ogni fase della pipeline.

4. Osservare l'avanzamento nella pipeline. Attendi un messaggio di successo sull'azione che utilizza la risorsa associata a un altro AWS account.

> **Note**
>
> Se si tenta di visualizzare i dettagli dell'operazione quando si è effettuato l'accesso con *AccountA*, verrà restituito un errore. Esci, quindi accedi con *AccountB* per visualizzare i dettagli della distribuzione. CodeDeploy

Esegui la migrazione delle pipeline di sondaggi per utilizzare il rilevamento delle modifiche basato sugli eventi

AWS CodePipeline supporta la distribuzione completa e end-to-end continua, che include l'avvio della pipeline ogni volta che si verifica una modifica del codice. Esistono due modi supportati per avviare la pipeline in caso di modifica del codice: rilevamento delle modifiche basato sugli eventi e polling. Consigliamo di utilizzare il rilevamento delle modifiche basato sugli eventi per le pipeline.

Utilizza le procedure qui incluse per migrare (aggiornare) le tue pipeline di polling al metodo di rilevamento delle modifiche basato sugli eventi per la tua pipeline.

Il metodo di rilevamento delle modifiche basato sugli eventi consigliato per le pipeline è determinato dall'origine della pipeline, ad esempio. CodeCommit In tal caso, ad esempio, la pipeline di sondaggio dovrebbe migrare al rilevamento delle modifiche basato sugli eventi con. EventBridge

Come migrare i sondaggi

Per migrare le pipeline di sondaggi, determina le tue pipeline di sondaggio e quindi determina il metodo di rilevamento delle modifiche consigliato basato sugli eventi:

- Utilizza la procedura descritta per determinare le tue pipeline di sondaggi. Visualizzazione delle pipeline di sondaggio nel proprio account
- Nella tabella, individuate il tipo di sorgente della vostra pipeline, quindi scegliete la procedura con l'implementazione che desiderate utilizzare per migrare la pipeline di polling. Ogni sezione contiene diversi metodi per la migrazione, ad esempio l'utilizzo della CLI o. AWS CloudFormation

Come migrare le pipeline nel metodo di rilevamento delle modifiche consigliato		
Fonte della pipeline	Metodo di rilevamento consiglia to basato sugli eventi	Procedure di migrazione
AWS CodeCommit	EventBridge (consigliato).	Per informazioni, consulta Migra le pipeline di polling con una fonte CodeCommit .
Amazon S3	EventBridge e bucket abilitato per le notifiche degli eventi (consigliato).	Per informazioni, consulta Esegui la migrazione delle pipeline di polling con una fonte S3 abilitata per gli eventi.
Amazon S3	EventBridge e una AWS CloudTrail pista.	Per informazioni, consulta Migra le pipeline di polling con un sorgente e un trail S3 CloudTrail .
GitHub versione 1	Connessioni (consigliate)	Per informazioni, consulta Migra le pipeline di polling per un' GitHub azione sorgente della versione 1 verso le connessioni.
GitHub versione 1	Webhook	Per informazioni, consulta Migra le pipeline di polling per un'azione sorgente della GitHub versione 1 ai webhook.

> **Important**
>
> Per gli aggiornamenti della configurazione delle azioni della pipeline applicabili, ad esempio le pipeline con un'azione della GitHub versione 1, è necessario impostare esplicitamente il `PollForSourceChanges` parametro su false all'interno della configurazione dell'azione Source per impedire il polling di una pipeline. Di conseguenza, è possibile configurare erroneamente una pipeline sia con il rilevamento delle modifiche basato sugli eventi che con il polling, ad esempio configurando una regola e omettendo anche il parametro. EventBridge `PollForSourceChanges` Questa operazione si traduce in esecuzioni di pipeline duplicate e la pipeline viene conteggiata rispetto al limite sul numero totale di pipeline di polling che per impostazione predefinita è molto più basso delle pipeline basate su eventi. Per ulteriori informazioni, consulta Quote in AWS CodePipeline.

Visualizzazione delle pipeline di sondaggio nel proprio account

Come primo passaggio, utilizza uno dei seguenti script per determinare quali pipeline del tuo account sono configurate per il polling. Queste sono le pipeline da migrare al rilevamento delle modifiche basato sugli eventi.

Visualizzazione delle pipeline di sondaggio nel tuo account (script)

Segui questi passaggi per utilizzare uno script per determinare le pipeline del tuo account che utilizzano il polling.

1. Apri una finestra di terminale, quindi esegui una delle seguenti operazioni:

   - Eseguite il comando seguente per creare un nuovo script denominato PollingPipelinesExtractor.sh.

     ```
     vi PollingPipelinesExtractor.sh
     ```

   - Per usare uno script python, esegui il comando seguente per creare un nuovo script python chiamato .py. PollingPipelinesExtractor

     ```
     vi PollingPipelinesExtractor.py
     ```

2. Copia e incolla il seguente codice nello script. PollingPipelinesExtractor Esegui una di queste operazioni:

- Copia e incolla il codice seguente nello PollingPipelinesExtractorscript.sh.

```
#!/bin/bash

set +x

POLLING_PIPELINES=()
LAST_EXECUTED_DATES=()
NEXT_TOKEN=null
HAS_NEXT_TOKEN=true
if [[ $# -eq 0 ]] ; then
    echo 'Please provide region name'
    exit 0
fi
REGION=$1


while [ "$HAS_NEXT_TOKEN" != "false" ]; do
    if [ "$NEXT_TOKEN" != "null" ];
        then
            LIST_PIPELINES_RESPONSE=$(aws codepipeline list-pipelines --region
 $REGION --next-token $NEXT_TOKEN)
        else
            LIST_PIPELINES_RESPONSE=$(aws codepipeline list-pipelines --region
 $REGION)
    fi
    LIST_PIPELINES=$(jq -r '.pipelines[].name' <<< "$LIST_PIPELINES_RESPONSE")
    NEXT_TOKEN=$(jq -r '.nextToken' <<< "$LIST_PIPELINES_RESPONSE")
    if [ "$NEXT_TOKEN" == "null" ];
        then
            HAS_NEXT_TOKEN=false
    fi

    for pipline_name in $LIST_PIPELINES
    do
        PIPELINE=$(aws codepipeline get-pipeline --name $pipline_name --region
 $REGION)
        HAS_POLLABLE_ACTIONS=$(jq '.pipeline.stages[].actions[] |
 select(.actionTypeId.category == "Source") | select(.actionTypeId.owner
 == ("ThirdParty","AWS")) | select(.actionTypeId.provider ==
 ("GitHub","S3","CodeCommit")) | select(.configuration.PollForSourceChanges ==
 ("true",null))' <<< "$PIPELINE")
        if [ ! -z "$HAS_POLLABLE_ACTIONS" ];
```

```
        then
            POLLING_PIPELINES+=("$pipline_name")
            PIPELINE_EXECUTIONS=$(aws codepipeline list-pipeline-executions --
pipeline-name $pipline_name --region $REGION)
            LAST_EXECUTION=$(jq -r '.pipelineExecutionSummaries[0]' <<<
 "$PIPELINE_EXECUTIONS")
            if [ "$LAST_EXECUTION" != "null" ];
                then
                    LAST_EXECUTED_TIMESTAMP=$(jq -r '.startTime' <<<
 "$LAST_EXECUTION")
                    LAST_EXECUTED_DATE="$(date -r ${LAST_EXECUTED_TIMESTAMP%.*})"
                else
                    LAST_EXECUTED_DATE="Not executed in last year"
            fi
            LAST_EXECUTED_DATES+=("$LAST_EXECUTED_DATE")
        fi
    done

done

fileName=$REGION-$(date +%s)
printf "| %-30s | %-30s |\n" "Polling Pipeline Name" "Last Executed Time"
printf "| %-30s | %-30s |\n" "_____________________" "___________________"
for i in "${!POLLING_PIPELINES[@]}"; do
  printf "| %-30s | %-30s |\n" "${POLLING_PIPELINES[i]}"
 "${LAST_EXECUTED_DATES[i]}"
  printf "${POLLING_PIPELINES[i]}," >> $fileName.csv
done

printf "\nSaving Polling Pipeline Names to file $fileName.csv."
```

- Copia e incolla il codice seguente nello PollingPipelinesExtractorscript.py.

```
import boto3
import sys
import time
import math

hasNextToken = True
nextToken = ""
pollablePipelines = []
lastExecutedTimes = []
if len(sys.argv) == 1:
    raise Exception("Please provide region name.")
```

```
session = boto3.Session(profile_name='default', region_name=sys.argv[1])
codepipeline = session.client('codepipeline')

def is_pollable_action(action):
    actionTypeId = action['actionTypeId']
    configuration = action['configuration']
    return actionTypeId['owner'] in {"AWS", "ThirdParty"}
 and actionTypeId['provider'] in {"GitHub", "CodeCommit",
 "S3"} and ('PollForSourceChanges' not in configuration or
 configuration['PollForSourceChanges'] == 'true')

def has_pollable_actions(pipeline):
    hasPollableAction = False
    pipelineDefinition = codepipeline.get_pipeline(name=pipeline['name'])
['pipeline']
    for action in pipelineDefinition['stages'][0]['actions']:
        hasPollableAction = is_pollable_action(action)
        if hasPollableAction:
            break
    return hasPollableAction

def get_last_executed_time(pipelineName):

 pipelineExecutions=codepipeline.list_pipeline_executions(pipelineName=pipelineName)
['pipelineExecutionSummaries']
    if pipelineExecutions:
        return pipelineExecutions[0]['startTime'].strftime("%A %m/%d/%Y, %H:%M:
%S")
    else:
        return "Not executed in last year"

while hasNextToken:
    if nextToken=="":
        list_pipelines_response = codepipeline.list_pipelines()
    else:
        list_pipelines_response =
 codepipeline.list_pipelines(nextToken=nextToken)
    if 'nextToken' in list_pipelines_response:
        nextToken = list_pipelines_response['nextToken']
    else:
        hasNextToken= False
    for pipeline in list_pipelines_response['pipelines']:
        if has_pollable_actions(pipeline):
            pollablePipelines.append(pipeline['name'])
```

```
            lastExecutedTimes.append(get_last_executed_time(pipeline['name']))

fileName="{region}-
{timeNow}.csv".format(region=sys.argv[1],timeNow=math.trunc(time.time()))
file = open(fileName, 'w')

print ("{:<30} {:<30} {:<30}".format('Polling Pipeline Name', '|','Last Executed
 Time'))
print ("{:<30} {:<30} {:<30}".format('_____________________',
 '|','___________________'))
for i in range(len(pollablePipelines)):
    print("{:<30} {:<30} {:<30}".format(pollablePipelines[i], '|',
 lastExecutedTimes[i]))
    file.write("{pipeline},".format(pipeline=pollablePipelines[i]))
file.close()
print("\nSaving Polling Pipeline Names to file
 {fileName}".format(fileName=fileName))
```

3. Per ogni regione in cui sono presenti pipeline, è necessario eseguire lo script per quella regione. Per eseguire lo script, effettuate una delle seguenti operazioni:

   - Eseguite il comando seguente per eseguire lo script denominato PollingPipelinesExtractor.sh. In questo esempio, la regione è us-west-2.

     ```
     ./PollingPipelinesExtractor.sh us-west-2
     ```

   - Per lo script python, esegui il comando seguente per eseguire lo script python denominato .py. PollingPipelinesExtractor In questo esempio, la regione è us-west-2.

     ```
     python3 PollingPipelinesExtractor.py us-west-2
     ```

   Nel seguente esempio di output dello script, la regione us-west-2 ha restituito un elenco di pipeline di polling e mostra l'ora dell'ultima esecuzione per ciascuna pipeline.

```
 % ./pollingPipelineExtractor.sh us-west-2

| Polling Pipeline Name | Last Executed Time           |
| _____________________ | ___________________          |
| myCodeBuildPipeline   | Wed Mar 8 09:35:49 PST 2023  |
| myCodeCommitPipeline  | Mon Apr 24 22:32:32 PDT 2023 |
```

```
| TestPipeline          | Not executed in last year    |

Saving list of polling pipeline names to us-west-2-1682496174.csv...%
```

Analizza l'output dello script e, per ogni pipeline nell'elenco, aggiorna la fonte di polling con il metodo di rilevamento delle modifiche basato sugli eventi consigliato.

> **Note**
>
> Le pipeline di polling sono determinate dalla configurazione delle azioni della pipeline per il parametro. `PollForSourceChanges` Se nella configurazione di origine della pipeline il `PollForSourceChanges` parametro è omesso, per impostazione CodePipeline predefinita esegue il polling del repository per verificare la presenza di modifiche all'origine. Questo comportamento è lo stesso che se fosse `PollForSourceChanges` incluso e impostato su true. Per ulteriori informazioni, consulta i parametri di configurazione per l'azione di origine della pipeline, ad esempio i parametri di configurazione dell'azione sorgente di Amazon S3 in. Azione all'origine di Amazon S3

Tieni presente che questo script genera anche un file.csv contenente l'elenco delle pipeline di polling presenti nel tuo account e salva il file.csv nella cartella di lavoro corrente.

Migra le pipeline di polling con una fonte CodeCommit

Puoi migrare la tua pipeline di polling per utilizzarla per EventBridge rilevare le modifiche nel tuo repository di CodeCommit origine o nel tuo bucket di origine Amazon S3.

CodeCommit-- Per una pipeline con una CodeCommit fonte, modifica la pipeline in modo da automatizzare il rilevamento delle modifiche. EventBridge Scegliete uno dei seguenti metodi per implementare la migrazione:

- Console: Migrazione delle pipeline di polling (o sorgente Amazon CodeCommit S3) (console)
- CLI: Migrazione delle pipeline di polling (CodeCommit origine) (CLI)
- AWS CloudFormation: Migra le pipeline di polling (CodeCommit source) (modello)AWS CloudFormation

Migrazione delle pipeline di polling (o sorgente Amazon CodeCommit S3) (console)

Puoi utilizzare la CodePipeline console per aggiornare la pipeline da utilizzare per EventBridge rilevare le modifiche nel tuo repository di CodeCommit origine o nel tuo bucket di origine Amazon S3.

> **Note**
>
> Quando usi la console per modificare una pipeline con un repository di CodeCommit origine o un bucket di origine Amazon S3, la regola e il ruolo IAM vengono creati automaticamente. Se utilizzi il AWS CLI per modificare la pipeline, devi creare tu stesso la EventBridge regola e il ruolo IAM. Per ulteriori informazioni, consulta CodeCommit azioni di origine e EventBridge.

Utilizza queste fasi per modificare una pipeline che utilizza controlli periodici. Se vuoi creare una pipeline, consulta Creare una pipeline in CodePipeline.

Per modificare la fase di origine della pipeline

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegliere il nome della pipeline da modificare. Questa operazione apre una visualizzazione dettagliata della pipeline. compreso lo stato di ciascuna delle operazioni in ciascuna fase della pipeline.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. In Edit stage (Modifica fase), scegli l'icona di modifica sull'operazione di origine.
5. Espandi le opzioni di rilevamento delle modifiche e scegli Usa CloudWatch eventi per avviare automaticamente la mia pipeline quando si verifica una modifica (scelta consigliata).

   Viene visualizzato un messaggio che mostra la EventBridge regola da creare per questa pipeline. Scegli Aggiorna.

   Se stai aggiornando una pipeline che ha una fonte Amazon S3, viene visualizzato il seguente messaggio. Scegli Aggiorna.
6. Al termine della modifica della pipeline, scegliere Save pipeline changes (Salva modifiche pipeline) per tornare alla pagina di riepilogo.

Un messaggio mostra il nome della EventBridge regola da creare per la pipeline. Seleziona Salva e continua.

7. Per testare la tua azione, rilascia una modifica utilizzando il AWS CLI comando per confermare una modifica alla fonte specificata nella fase di origine della pipeline.

Migrazione delle pipeline di polling (CodeCommit origine) (CLI)

Segui questi passaggi per modificare una pipeline che utilizza il polling (controlli periodici) per utilizzare una regola per avviare la pipeline. EventBridge Se vuoi creare una pipeline, consulta Creare una pipeline in CodePipeline.

Per creare una pipeline basata sugli eventi con CodeCommit, devi modificare il `PollForSourceChanges` parametro della pipeline e quindi creare le seguenti risorse:

- EventBridge evento
- Ruolo IAM per consentire all'evento di avviare la pipeline

Per modificare i parametri della PollForSourceChanges pipeline

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

1. Esegui il comando get-pipeline per copiare la struttura della pipeline in un file JSON. Ad esempio, per una pipeline denominata `MyFirstPipeline`, esegui il seguente comando:

```
aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
```

Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.

2. Apri il file JSON in qualsiasi editor di testo normale e modifica la fase di origine modificando il parametro `PollForSourceChanges` su `false`, come illustrato in questo esempio.

   Perché occorre apportare questa modifica? La modifica di questo parametro in `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

   ```
   "configuration": {
       "PollForSourceChanges": "false",
       "BranchName": "main",
       "RepositoryName": "MyTestRepo"
   },
   ```

3. Se stai utilizzando la struttura della pipeline recuperata tramite il comando get-pipeline, rimuovi le righe `metadata` dal file JSON. In caso contrario, il comando update-pipeline non è in grado di utilizzarlo. Rimuovi le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`.

   Ad esempio, rimuovere dalla struttura le seguenti righe:

   ```
   "metadata": {
       "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
       "created": "date",
       "updated": "date"
   },
   ```

   Salvare il file.

4. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline:

   > **Important**
   >
   > Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

   ```
   aws codepipeline update-pipeline --cli-input-json file://pipeline.json
   ```

   Questo comando restituisce l'intera struttura della pipeline modificata.

> **Note**
>
> Il comando update-pipeline arresta la pipeline. Se è in corso di elaborazione una versione durante l'esecuzione del comando update-pipeline, tale elaborazione viene arrestata. Per elaborare tale versione utilizzando la pipeline aggiornata, devi avviare manualmente la pipeline. Utilizza il comando **`start-pipeline-execution`** per avviare manualmente la pipeline.

Per creare una EventBridge regola con CodeCommit come origine dell'evento e CodePipeline come destinazione

1. Aggiungi le autorizzazioni EventBridge da utilizzare per CodePipeline richiamare la regola. Per ulteriori informazioni, consulta Utilizzo delle politiche basate sulle risorse per Amazon. EventBridge

   a. Usa l'esempio seguente per creare la politica di fiducia che consente di EventBridge assumere il ruolo di servizio. Denomina la policy di attendibilità `trustpolicyforEB.json`.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Principal": {
                   "Service": "events.amazonaws.com"
               },
               "Action": "sts:AssumeRole"
           }
       ]
   }
   ```

   b. Utilizza il seguente comando per creare il ruolo `Role-for-MyRule` e collegare la policy di attendibilità.

   ```
   aws iam create-role --role-name Role-for-MyRule --assume-role-policy-document
    file://trustpolicyforEB.json
   ```

c. Crea il JSON della policy delle autorizzazioni, come mostrato in questo esempio, per la pipeline denominata `MyFirstPipeline`. Denomina la policy delle autorizzazioni `permissionspolicyforEB.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:StartPipelineExecution"
            ],
            "Resource": [
                "arn:aws:codepipeline:us-west-2:80398EXAMPLE:MyFirstPipeline"
            ]
        }
    ]
}
```

d. Utilizza il comando seguente per collegare la policy delle autorizzazioni `CodePipeline-Permissions-Policy-for-EB` al ruolo `Role-for-MyRule`.

Perché occorre apportare questa modifica? L'aggiunta di questa politica al ruolo crea le autorizzazioni per EventBridge.

```
aws iam put-role-policy --role-name Role-for-MyRule --policy-name CodePipeline-
Permissions-Policy-For-EB --policy-document file://permissionspolicyforEB.json
```

2. Richiama il comando put-rule e includi i parametri `--name`, `--event-pattern` e `--role-arn`.

Perché occorre apportare questa modifica? Questo comando consente a AWS CloudFormation di creare l'evento.

Il seguente comando di esempio crea una regola denominata `MyCodeCommitRepoRule`.

```
aws events put-rule --name "MyCodeCommitRepoRule" --event-pattern "{\"source\":
[\"aws.codecommit\"],\"detail-type\":[\"CodeCommit Repository State Change\"],
\"resources\":[\"repository-ARN\"],\"detail\":{\"referenceType\":[\"branch\"],
\"referenceName\":[\"main\"]}}" --role-arn "arn:aws:iam::ACCOUNT_ID:role/Role-for-
MyRule"
```

3. Per aggiungere CodePipeline come destinazione, chiamate il put-targets comando e includete i seguenti parametri:

   - Il parametro `--rule` viene utilizzato con il `rule_name` che hai creato utilizzando put-rule.
   - Il parametro `--targets` viene utilizzato con l'`Id` elenco della destinazione nell'elenco delle destinazioni e l'ARN della pipeline di destinazione.

   Il comando di esempio seguente specifica che per la regola denominata `MyCodeCommitRepoRule`, la destinazione `Id` è composta dal numero uno, per indicare che in un elenco di destinazioni per la regola questa è la destinazione 1. Il comando di esempio specifica anche un esempio di ARN per la pipeline. La pipeline si avvia quando si verifica una modifica nel repository.

   ```
   aws events put-targets --rule MyCodeCommitRepoRule --targets
    Id=1,Arn=arn:aws:codepipeline:us-west-2:80398EXAMPLE:TestPipeline
   ```

Migra le pipeline di polling (CodeCommit source) (modello)AWS CloudFormation

Per creare una pipeline basata sugli eventi con AWS CodeCommit, modificate il `PollForSourceChanges` parametro della pipeline e quindi aggiungete le seguenti risorse al modello:

- EventBridge Una regola
- Un ruolo IAM per la tua EventBridge regola

Se lo utilizzi AWS CloudFormation per creare e gestire le tue pipeline, il tuo modello include contenuti come i seguenti.

> **Note**
>
> La proprietà `Configuration` nella fase di origine chiamata `PollForSourceChanges`. Se questa proprietà non è inclusa nel modello, allora `PollForSourceChanges` è impostato su `true` per impostazione predefinita.

YAML

```
Resources:
  AppPipeline:
    Type: AWS::CodePipeline::Pipeline
    Properties:
      Name: codecommit-polling-pipeline
      RoleArn:
        !GetAtt CodePipelineServiceRole.Arn
      Stages:
        -
          Name: Source
          Actions:
            -
              Name: SourceAction
              ActionTypeId:
                Category: Source
                Owner: AWS
                Version: 1
                Provider: CodeCommit
              OutputArtifacts:
                - Name: SourceOutput
              Configuration:
                BranchName: !Ref BranchName
                RepositoryName: !Ref RepositoryName
                PollForSourceChanges: true
              RunOrder: 1
```

JSON

```
"Stages": [
    {
        "Name": "Source",
  "Actions": [{
      "Name": "SourceAction",
      "ActionTypeId": {
    "Category": "Source",
    "Owner": "AWS",
    "Version": 1,
    "Provider": "CodeCommit"
      },
      "OutputArtifacts": [{
          "Name": "SourceOutput"
```

```
      }],
      "Configuration": {
          "BranchName": {
        "Ref": "BranchName"
    },
    "RepositoryName": {
        "Ref": "RepositoryName"
    },
    "PollForSourceChanges": true
            },
            "RunOrder": 1
        }]
    },
```

Per aggiornare il AWS CloudFormation modello di pipeline e creare una regola EventBridge

1. Nel modello, sotto`Resources`, utilizza la `AWS::IAM::Role` AWS CloudFormation risorsa per configurare il ruolo IAM che consente all'evento di avviare la pipeline. Questa voce crea un ruolo che utilizza due policy:

   - La prima policy consente di assumere quel ruolo.
   - La seconda policy fornisce le autorizzazioni per avviare la pipeline.

   Perché occorre apportare questa modifica? L'aggiunta della `AWS::IAM::Role` risorsa consente di AWS CloudFormation creare autorizzazioni per. EventBridge Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

   ```
   EventRole:
     Type: AWS::IAM::Role
     Properties:
       AssumeRolePolicyDocument:
         Version: 2012-10-17
         Statement:
           -
             Effect: Allow
             Principal:
               Service:
                 - events.amazonaws.com
   ```

```
            Action: sts:AssumeRole
      Path: /
      Policies:
        -
          PolicyName: eb-pipeline-execution
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              -
                Effect: Allow
                Action: codepipeline:StartPipelineExecution
                Resource: !Join [ '', [ 'arn:aws:codepipeline:', !Ref
 'AWS::Region', ':', !Ref 'AWS::AccountId', ':', !Ref AppPipeline ] ]
```

JSON

```
"EventRole": {
  "Type": "AWS::IAM::Role",
  "Properties": {
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Principal": {
            "Service": [
              "events.amazonaws.com"
            ]
          },
          "Action": "sts:AssumeRole"
        }
      ]
    },
    "Path": "/",
    "Policies": [
      {
        "PolicyName": "eb-pipeline-execution",
        "PolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Action": "codepipeline:StartPipelineExecution",
```

```
              "Resource": {
                "Fn::Join": [
                  "",
                  [
                    "arn:aws:codepipeline:",
                    {
                      "Ref": "AWS::Region"
                    },
                    ":",
                    {
                      "Ref": "AWS::AccountId"
                    },
                    ":",
                    {
                      "Ref": "AppPipeline"
                    }
                  ]

...
```

2. Nel modello, sotto`Resources`, usa la `AWS::Events::Rule` AWS CloudFormation risorsa per aggiungere una EventBridge regola. Questo modello di eventi crea un evento che monitora le modifiche push al tuo repository. Quando EventBridge rileva una modifica dello stato del repository, la regola viene `StartPipelineExecution` richiamata sulla pipeline di destinazione.

   Perché sto apportando questa modifica? L'aggiunta della `AWS::Events::Rule` risorsa AWS CloudFormation consente di creare l'evento. Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

```
  EventRule:
    Type: AWS::Events::Rule
    Properties:
      EventPattern:
        source:
          - aws.codecommit
        detail-type:
          - 'CodeCommit Repository State Change'
        resources:
```

```
          - !Join [ '', [ 'arn:aws:codecommit:', !Ref 'AWS::Region', ':', !Ref
 'AWS::AccountId', ':', !Ref RepositoryName ] ]
        detail:
          event:
            - referenceCreated
            - referenceUpdated
          referenceType:
            - branch
          referenceName:
            - main
      Targets:
        -
          Arn:
            !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region', ':', !Ref
 'AWS::AccountId', ':', !Ref AppPipeline ] ]
          RoleArn: !GetAtt EventRole.Arn
          Id: codepipeline-AppPipeline
```

JSON

```
"EventRule": {
  "Type": "AWS::Events::Rule",
  "Properties": {
    "EventPattern": {
      "source": [
        "aws.codecommit"
      ],
      "detail-type": [
        "CodeCommit Repository State Change"
      ],
      "resources": [
        {
          "Fn::Join": [
            "",
            [
              "arn:aws:codecommit:",
              {
                "Ref": "AWS::Region"
              },
              ":",
              {
                "Ref": "AWS::AccountId"
```

```
              },
              ":",
              {
                "Ref": "RepositoryName"
              }
            ]
          ]
        }
      ],
      "detail": {
        "event": [
          "referenceCreated",
          "referenceUpdated"
        ],
        "referenceType": [
          "branch"
        ],
        "referenceName": [
          "main"
        ]
      }
    },
    "Targets": [
      {
        "Arn": {
          "Fn::Join": [
            "",
            [
              "arn:aws:codepipeline:",
              {
                "Ref": "AWS::Region"
              },
              ":",
              {
                "Ref": "AWS::AccountId"
              },
              ":",
              {
                "Ref": "AppPipeline"
              }
            ]
          ]
        },
        "RoleArn": {
```

```
            "Fn::GetAtt": [
              "EventRole",
              "Arn"
            ]
          },
          "Id": "codepipeline-AppPipeline"
        }
      ]
    }
},
```

3. Salva il modello aggiornato nel computer locale e quindi apri la console AWS CloudFormation .
4. Seleziona lo stack e scegli Create Change Set for Current Stack (Crea set di modifiche per lo stack corrente).
5. Carica il modello e quindi visualizza le modifiche elencate in AWS CloudFormation. Queste sono le modifiche da apportare allo stack. Le nuove risorse dovrebbero essere visibili nell'elenco.
6. Scegliere Execute (Esegui).

Per modificare i parametri della pipeline PollForSourceChanges

> **Important**
>
> In molti casi, il parametro `PollForSourceChanges` è preimpostato su "true" al momento della creazione di una pipeline. Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

- Nel modello, modifica `PollForSourceChanges` in `false`. Se non hai incluso `PollForSourceChanges` nella definizione della pipeline, aggiungilo e impostalo su `false`.

  Perché occorre apportare questa modifica? La modifica di questo parametro in `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

YAML

```
        Name: Source
        Actions:
          -
            Name: SourceAction
            ActionTypeId:
              Category: Source
              Owner: AWS
              Version: 1
              Provider: CodeCommit
            OutputArtifacts:
              - Name: SourceOutput
            Configuration:
              BranchName: !Ref BranchName
              RepositoryName: !Ref RepositoryName
              PollForSourceChanges: false
            RunOrder: 1
```

JSON

```
{
  "Name": "Source",
  "Actions": [
    {
      "Name": "SourceAction",
      "ActionTypeId": {
        "Category": "Source",
        "Owner": "AWS",
        "Version": 1,
        "Provider": "CodeCommit"
      },
      "OutputArtifacts": [
        {
          "Name": "SourceOutput"
        }
      ],
      "Configuration": {
        "BranchName": {
          "Ref": "BranchName"
        },
        "RepositoryName": {
```

```
          "Ref": "RepositoryName"
        },
        "PollForSourceChanges": false
      },
      "RunOrder": 1
    }
  ]
},
```

Example

Quando crei queste risorse con AWS CloudFormation, la pipeline viene attivata quando i file nel tuo repository vengono creati o aggiornati. Qui è riportato il frammento di modello finale:

YAML

```
Resources:
  EventRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          -
            Effect: Allow
            Principal:
              Service:
                - events.amazonaws.com
            Action: sts:AssumeRole
      Path: /
      Policies:
        -
          PolicyName: eb-pipeline-execution
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              -
                Effect: Allow
                Action: codepipeline:StartPipelineExecution
                Resource: !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region',
 ':', !Ref 'AWS::AccountId', ':', !Ref AppPipeline ] ]
```

```
 EventRule:
   Type: AWS::Events::Rule
   Properties:
     EventPattern:
       source:
         - aws.codecommit
       detail-type:
         - 'CodeCommit Repository State Change'
       resources:
         - !Join [ '', [ 'arn:aws:codecommit:', !Ref 'AWS::Region', ':', !Ref
'AWS::AccountId', ':', !Ref RepositoryName ] ]
       detail:
         event:
           - referenceCreated
           - referenceUpdated
         referenceType:
           - branch
         referenceName:
           - main
     Targets:
       -
         Arn:
           !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region', ':', !Ref
'AWS::AccountId', ':', !Ref AppPipeline ] ]
         RoleArn: !GetAtt EventRole.Arn
         Id: codepipeline-AppPipeline
 AppPipeline:
   Type: AWS::CodePipeline::Pipeline
   Properties:
     Name: codecommit-events-pipeline
     RoleArn:
       !GetAtt CodePipelineServiceRole.Arn
     Stages:
       -
         Name: Source
         Actions:
           -
             Name: SourceAction
             ActionTypeId:
               Category: Source
               Owner: AWS
               Version: 1
               Provider: CodeCommit
             OutputArtifacts:
```

```
                - Name: SourceOutput
              Configuration:
                BranchName: !Ref BranchName
                RepositoryName: !Ref RepositoryName
                PollForSourceChanges: false
              RunOrder: 1


...
```

JSON

```
    "Resources": {

...

        "EventRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "events.amazonaws.com"
                                ]
                            },
                            "Action": "sts:AssumeRole"
                        }
                    ]
                },
                "Path": "/",
                "Policies": [
                    {
                        "PolicyName": "eb-pipeline-execution",
                        "PolicyDocument": {
                            "Version": "2012-10-17",
                            "Statement": [
                                {
                                    "Effect": "Allow",
                                    "Action": "codepipeline:StartPipelineExecution",
```

```
                                "Resource": {
                                    "Fn::Join": [
                                        "",
                                        [
                                            "arn:aws:codepipeline:",
                                            {
                                                "Ref": "AWS::Region"
                                            },
                                            ":",
                                            {
                                                "Ref": "AWS::AccountId"
                                            },
                                            ":",
                                            {
                                                "Ref": "AppPipeline"
                                            }
                                        ]
                                    ]
                                }
                            }
                        ]
                    }
                }
            ]
        }
    },
    "EventRule": {
        "Type": "AWS::Events::Rule",
        "Properties": {
            "EventPattern": {
                "source": [
                    "aws.codecommit"
                ],
                "detail-type": [
                    "CodeCommit Repository State Change"
                ],
                "resources": [
                    {
                        "Fn::Join": [
                            "",
                            [
                                "arn:aws:codecommit:",
                                {
                                    "Ref": "AWS::Region"
```

```
                                },
                                ":",
                                {
                                    "Ref": "AWS::AccountId"
                                },
                                ":",
                                {
                                    "Ref": "RepositoryName"
                                }
                            ]
                        ]
                    }
                ],
                "detail": {
                    "event": [
                        "referenceCreated",
                        "referenceUpdated"
                    ],
                    "referenceType": [
                        "branch"
                    ],
                    "referenceName": [
                        "main"
                    ]
                }
            },
            "Targets": [
                {
                    "Arn": {
                        "Fn::Join": [
                            "",
                            [
                                "arn:aws:codepipeline:",
                                {
                                    "Ref": "AWS::Region"
                                },
                                ":",
                                {
                                    "Ref": "AWS::AccountId"
                                },
                                ":",
                                {
                                    "Ref": "AppPipeline"
                                }
```

```
                            ]
                        ]
                    },
                    "RoleArn": {
                        "Fn::GetAtt": [
                            "EventRole",
                            "Arn"
                        ]
                    },
                    "Id": "codepipeline-AppPipeline"
                }
            ]
        }
    },
    "AppPipeline": {
        "Type": "AWS::CodePipeline::Pipeline",
        "Properties": {
            "Name": "codecommit-events-pipeline",
            "RoleArn": {
                "Fn::GetAtt": [
                    "CodePipelineServiceRole",
                    "Arn"
                ]
            },
            "Stages": [
                {
                    "Name": "Source",
                    "Actions": [
                        {
                            "Name": "SourceAction",
                            "ActionTypeId": {
                                "Category": "Source",
                                "Owner": "AWS",
                                "Version": 1,
                                "Provider": "CodeCommit"
                            },
                            "OutputArtifacts": [
                                {
                                    "Name": "SourceOutput"
                                }
                            ],
                            "Configuration": {
                                "BranchName": {
                                    "Ref": "BranchName"
```

```
                                },
                                "RepositoryName": {
                                    "Ref": "RepositoryName"
                                },
                                "PollForSourceChanges": false
                            },
                            "RunOrder": 1
                        }
                    ]
                },

...
```

Esegui la migrazione delle pipeline di polling con una fonte S3 abilitata per gli eventi

Per una pipeline con un'origine Amazon S3, modifica la pipeline in modo che il rilevamento delle modifiche sia automatizzato EventBridge tramite e con un bucket di origine abilitato per le notifiche degli eventi. Questo è il metodo consigliato se utilizzi la CLI o AWS CloudFormation per migrare la tua pipeline.

> **Note**
>
> Ciò include l'utilizzo di un bucket abilitato per le notifiche degli eventi, in cui non è necessario creare un percorso separato. CloudTrail Se utilizzi la console, vengono impostate automaticamente una regola e un CloudTrail percorso per gli eventi. Per questi passaggi, vediMigra le pipeline di polling con un sorgente e un trail S3 CloudTrail .

- CLI: Migra le pipeline di polling con un codice sorgente e trail CloudTrail (CLI) S3
- AWS CloudFormation: Migra le pipeline di polling con una sorgente e un trail S3 (modello) CloudTrail AWS CloudFormation

Esegui la migrazione delle pipeline di polling con un'origine S3 abilitata per gli eventi (CLI)

Segui questi passaggi per modificare una pipeline che utilizza il polling (controlli periodici) in cui utilizzare invece un evento. EventBridge Se vuoi creare una pipeline, consulta Creare una pipeline in CodePipeline.

Per creare una pipeline basata sugli eventi con Amazon S3, devi modificare il `PollForSourceChanges` parametro della pipeline e quindi creare le seguenti risorse:

- EventBridge regola dell'evento
- ruolo IAM per consentire all' EventBridge evento di avviare la pipeline

Per creare una EventBridge regola con Amazon S3 come origine dell'evento e CodePipeline come destinazione e applicare la politica delle autorizzazioni

1. Concedi le autorizzazioni EventBridge da utilizzare per CodePipeline richiamare la regola. Per ulteriori informazioni, consulta Utilizzo delle politiche basate sulle risorse per Amazon. EventBridge

   a. Usa l'esempio seguente per creare la politica di fiducia che consente di EventBridge assumere il ruolo di servizio. Denominalo `trustpolicyforEB.json`.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Principal": {
                   "Service": "events.amazonaws.com"
               },
               "Action": "sts:AssumeRole"
           }
       ]
   }
   ```

   b. Utilizza il seguente comando per creare il ruolo `Role-for-MyRule` e collegare la policy di attendibilità.

Perché occorre apportare questa modifica? L'aggiunta di questa politica di fiducia al ruolo crea le autorizzazioni per EventBridge.

```
aws iam create-role --role-name Role-for-MyRule --assume-role-policy-document
 file://trustpolicyforEB.json
```

c. Crea il JSON della policy delle autorizzazioni, come mostrato qui per la pipeline denominata `MyFirstPipeline`. Denomina la policy delle autorizzazioni `permissionspolicyforEB.json`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:StartPipelineExecution"
            ],
            "Resource": [
                "arn:aws:codepipeline:us-west-2:80398EXAMPLE:MyFirstPipeline"
            ]
        }
    ]
}
```

d. Utilizza il comando seguente per collegare la nuova policy delle autorizzazioni `CodePipeline-Permissions-Policy-for-EB` al ruolo `Role-for-MyRule` che hai creato.

```
aws iam put-role-policy --role-name Role-for-MyRule --policy-name CodePipeline-
Permissions-Policy-For-EB --policy-document file://permissionspolicyforEB.json
```

2. Richiama il comando put-rule e includi i parametri `--name`, `--event-pattern` e `--role-arn`.

   Il seguente comando di esempio crea una regola denominata `EnabledS3SourceRule`.

```
aws events put-rule --name "EnabledS3SourceRule" --event-pattern "{\"source\":
[\"aws.s3\"],\"detail-type\":[\"Object Created\"],\"detail\":{\"bucket\":{\"name\":
[\"my-bucket\"]}}}" --role-arn "arn:aws:iam::ACCOUNT_ID:role/Role-for-MyRule"
```

3. Per aggiungere CodePipeline come destinazione, chiamate il put-targets comando e includete i `--targets` parametri `--rule` and.

   Il comando seguente specifica che per la regola denominata `EnabledS3SourceRule`, la destinazione `Id` è composta dal numero uno, per indicare che in un elenco di destinazioni per la regola questa è la destinazione 1. Il comando specifica anche un esempio di ARN per la pipeline. La pipeline si avvia quando si verifica una modifica nel repository.

   ```
   aws events put-targets --rule EnabledS3SourceRule --targets Id=codepipeline-
   AppPipeline,Arn=arn:aws:codepipeline:us-west-2:80398EXAMPLE:TestPipeline
   ```

Per modificare il parametro della PollForSourceChanges pipeline

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

1. Esegui il comando get-pipeline per copiare la struttura della pipeline in un file JSON. Ad esempio, per una pipeline denominata `MyFirstPipeline`, esegui il seguente comando:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   ```

   Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.

2. Apri il file JSON in qualsiasi editor di testo normale e modifica la fase di origine modificando il parametro `PollForSourceChanges` per un bucket denominato `storage-bucketfalse` come mostrato nell'esempio seguente.

   Perché occorre apportare questa modifica? L'impostazione del parametro su `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

```
"configuration": {
    "S3Bucket": "storage-bucket",
    "PollForSourceChanges": "false",
    "S3ObjectKey": "index.zip"
},
```

3. Se stai utilizzando la struttura della pipeline recuperata tramite il comando get-pipeline, devi rimuovere le righe `metadata` dal file JSON. In caso contrario, il comando update-pipeline non è in grado di utilizzarlo. Rimuovi le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`.

   Ad esempio, rimuovere dalla struttura le seguenti righe:

   ```
   "metadata": {
       "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
       "created": "date",
       "updated": "date"
   },
   ```

   Salvare il file.

4. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline:

   > **Important**
   >
   > Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

   ```
   aws codepipeline update-pipeline --cli-input-json file://pipeline.json
   ```

   Questo comando restituisce l'intera struttura della pipeline modificata.

   > **Note**
   >
   > Il comando update-pipeline arresta la pipeline. Se è in corso di elaborazione una versione durante l'esecuzione del comando update-pipeline, tale elaborazione viene arrestata. Per elaborare tale versione utilizzando la pipeline aggiornata, devi avviare

manualmente la pipeline. Utilizza il comando start-pipeline-execution per avviare manualmente la pipeline.

Migra le pipeline di polling con una fonte S3 abilitata per gli eventi (modello)AWS CloudFormation

Questa procedura è per una pipeline in cui gli eventi sono abilitati nel bucket di origine.

Utilizza questi passaggi per modificare la tua pipeline con una fonte Amazon S3, dal polling al rilevamento delle modifiche basato sugli eventi.

Per creare una pipeline basata sugli eventi con Amazon S3, devi modificare il `PollForSourceChanges` parametro della pipeline e quindi aggiungere le seguenti risorse al modello:

- EventBridge regola e ruolo IAM per consentire a questo evento di avviare la pipeline.

Se lo utilizzi AWS CloudFormation per creare e gestire le tue pipeline, il tuo modello include contenuti come i seguenti.

> **Note**
>
> La proprietà `Configuration` nella fase di origine chiamata `PollForSourceChanges`. Se il modello non include questa proprietà, allora `PollForSourceChanges` è impostato su `true` per impostazione predefinita.

YAML

```
AppPipeline:
  Type: AWS::CodePipeline::Pipeline
  Properties:
    RoleArn: !GetAtt CodePipelineServiceRole.Arn
    Stages:
      -
        Name: Source
        Actions:
          -
```

```
            Name: SourceAction
            ActionTypeId:
              Category: Source
              Owner: AWS
              Version: 1
              Provider: S3
            OutputArtifacts:
              -
                Name: SourceOutput
            Configuration:
              S3Bucket: !Ref SourceBucket
              S3ObjectKey: !Ref S3SourceObjectKey
              PollForSourceChanges: true
            RunOrder: 1


...
```

JSON

```
        "AppPipeline": {
            "Type": "AWS::CodePipeline::Pipeline",
            "Properties": {
                "RoleArn": {
                    "Fn::GetAtt": ["CodePipelineServiceRole", "Arn"]
                },
                "Stages": [
                    {
                        "Name": "Source",
                        "Actions": [
                            {
                                "Name": "SourceAction",
                                "ActionTypeId": {
                                    "Category": "Source",
                                    "Owner": "AWS",
                                    "Version": 1,
                                    "Provider": "S3"
                                },
                                "OutputArtifacts": [
                                    {
                                        "Name": "SourceOutput"
                                    }
                                ],
```

```
                                "Configuration": {
                                    "S3Bucket": {
                                        "Ref": "SourceBucket"
                                    },
                                    "S3ObjectKey": {
                                        "Ref": "SourceObjectKey"
                                    },
                                    "PollForSourceChanges": true
                                },
                                "RunOrder": 1
                            }
                        ]
                    },


...
```

Per creare una EventBridge regola con Amazon S3 come origine dell'evento e CodePipeline come destinazione e applicare la politica delle autorizzazioni

1. Nel modello, sotto`Resources`, utilizza la `AWS::IAM::Role` AWS CloudFormation risorsa per configurare il ruolo IAM che consente all'evento di avviare la pipeline. Questa voce crea un ruolo che utilizza due policy:

   - La prima policy consente di assumere quel ruolo.
   - La seconda policy fornisce le autorizzazioni per avviare la pipeline.

   Perché occorre apportare questa modifica? L'aggiunta di `AWS::IAM::Role` risorse consente AWS CloudFormation di creare autorizzazioni per. EventBridge Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

   ```
   EventRole:
     Type: AWS::IAM::Role
     Properties:
       AssumeRolePolicyDocument:
         Version: 2012-10-17
         Statement:
           -
   ```

```
            Effect: Allow
            Principal:
              Service:
                - events.amazonaws.com
            Action: sts:AssumeRole
      Path: /
      Policies:
        -
          PolicyName: eb-pipeline-execution
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              -
                Effect: Allow
                Action: codepipeline:StartPipelineExecution
                Resource: !Join [ '', [ 'arn:aws:codepipeline:', !Ref
 'AWS::Region', ':', !Ref 'AWS::AccountId', ':', !Ref AppPipeline ] ]


...
```

JSON

```
  "EventRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Effect": "Allow",
            "Principal": {
              "Service": [
                "events.amazonaws.com"
              ]
            },
            "Action": "sts:AssumeRole"
          }
        ]
      },
      "Path": "/",
      "Policies": [
        {
```

```
          "PolicyName": "eb-pipeline-execution",
          "PolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
              {
                "Effect": "Allow",
                "Action": "codepipeline:StartPipelineExecution",
                "Resource": {
                  "Fn::Join": [
                    "",
                    [
                      "arn:aws:codepipeline:",
                      {
                        "Ref": "AWS::Region"
                      },
                      ":",
                      {
                        "Ref": "AWS::AccountId"
                      },
                      ":",
                      {
                        "Ref": "AppPipeline"
                      }
                    ]
                  ]

...
```

2. Usa la `AWS::Events::Rule` AWS CloudFormation risorsa per aggiungere una EventBridge regola. Questo modello di eventi crea un evento che monitora la creazione o l'eliminazione di oggetti nel bucket di origine Amazon S3. Inoltre, include una destinazione della pipeline. Quando viene creato un oggetto, questa regola invoca `StartPipelineExecution` la pipeline di destinazione.

   Perché occorre apportare questa modifica? L'aggiunta della `AWS::Events::Rule` risorsa consente di AWS CloudFormation creare l'evento. Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

```
  EventRule:
    Type: AWS::Events::Rule
```

```
    Properties:
      EventBusName: default
      EventPattern:
        source:
          - aws.s3
        detail-type:
          - Object Created
        detail:
          bucket:
            name:
              - !Ref SourceBucket
      Name: EnabledS3SourceRule
      State: ENABLED
      Targets:
        -
          Arn:
            !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region', ':', !Ref
 'AWS::AccountId', ':', !Ref AppPipeline ] ]
          RoleArn: !GetAtt EventRole.Arn
          Id: codepipeline-AppPipeline


...
```

JSON

```
  "EventRule": {
    "Type": "AWS::Events::Rule",
    "Properties": {
  "EventBusName": "default",
  "EventPattern": {
      "source": [
   "aws.s3"
      ],
      "detail-type": [
    "Object Created"
      ],
      "detail": {
    "bucket": {
        "name": [
      "s3-pipeline-source-fra-bucket"
        ]
```

```
        }
            }
  },
  "Name": "EnabledS3SourceRule",
        "State": "ENABLED",
        "Targets": [
        {
          "Arn": {
            "Fn::Join": [
              "",
              [
                "arn:aws:codepipeline:",
                {
                  "Ref": "AWS::Region"
                },
                ":",
                {
                  "Ref": "AWS::AccountId"
                },
                ":",
                {
                  "Ref": "AppPipeline"
                }
              ]
            ]
          },
          "RoleArn": {
            "Fn::GetAtt": [
              "EventRole",
              "Arn"
            ]
          },
          "Id": "codepipeline-AppPipeline"
        }
      ]
    }
  }
},

...
```

3. Salvare il modello aggiornato sul computer locale e aprire la console AWS CloudFormation .

4. Seleziona lo stack e scegli Create Change Set for Current Stack (Crea set di modifiche per lo stack corrente).
5. Caricare il modello aggiornato e quindi visualizzare le modifiche elencate in AWS CloudFormation. Queste sono le modifiche che verranno apportate allo stack. Le nuove risorse dovrebbero essere visibili nell'elenco.
6. Scegliere Execute (Esegui).

Per modificare i parametri della pipeline PollForSourceChanges

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

- Nel modello, modifica `PollForSourceChanges` in `false`. Se non hai incluso `PollForSourceChanges` nella definizione della pipeline, aggiungilo e impostalo su `false`.

  Perché occorre apportare questa modifica? La modifica di `PollForSourceChanges` in `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

  YAML

```
        Name: Source
        Actions:
          -
            Name: SourceAction
            ActionTypeId:
              Category: Source
              Owner: AWS
              Version: 1
              Provider: S3
            OutputArtifacts:
              - Name: SourceOutput
```

```
            Configuration:
              S3Bucket: !Ref SourceBucket
              S3ObjectKey: !Ref SourceObjectKey
              PollForSourceChanges: false
            RunOrder: 1
```

JSON

```
{
   "Name": "SourceAction",
   "ActionTypeId": {
     "Category": "Source",
     "Owner": "AWS",
     "Version": 1,
     "Provider": "S3"
   },
   "OutputArtifacts": [
     {
       "Name": "SourceOutput"
     }
   ],
   "Configuration": {
     "S3Bucket": {
       "Ref": "SourceBucket"
     },
     "S3ObjectKey": {
       "Ref": "SourceObjectKey"
     },
     "PollForSourceChanges": false
   },
   "RunOrder": 1
 }
```

Example

Quando si utilizzano AWS CloudFormation per creare queste risorse, la pipeline viene attivata quando i file nel repository vengono creati o aggiornati.

> **Note**
>
> Non è finita qui. Sebbene la pipeline sia stata creata, è necessario creare un secondo AWS CloudFormation modello per la pipeline Amazon S3. Se non crei il secondo modello, la pipeline non avrà la funzionalità di rilevamento delle modifiche.

YAML

```
Parameters:
  SourceObjectKey:
    Description: 'S3 source artifact'
    Type: String
    Default: SampleApp_Linux.zip
  ApplicationName:
    Description: 'CodeDeploy application name'
    Type: String
    Default: DemoApplication
  BetaFleet:
    Description: 'Fleet configured in CodeDeploy'
    Type: String
    Default: DemoFleet

Resources:
  SourceBucket:
    Type: AWS::S3::Bucket
    Properties:
      NotificationConfiguration:
        EventBridgeConfiguration:
          EventBridgeEnabled: true
      VersioningConfiguration:
        Status: Enabled
  CodePipelineArtifactStoreBucket:
    Type: AWS::S3::Bucket
  CodePipelineArtifactStoreBucketPolicy:
    Type: AWS::S3::BucketPolicy
    Properties:
      Bucket: !Ref CodePipelineArtifactStoreBucket
      PolicyDocument:
        Version: 2012-10-17
        Statement:
          -
```

```
            Sid: DenyUnEncryptedObjectUploads
            Effect: Deny
            Principal: '*'
            Action: s3:PutObject
            Resource: !Join [ '', [ !GetAtt CodePipelineArtifactStoreBucket.Arn, '/
*' ] ]
            Condition:
              StringNotEquals:
                s3:x-amz-server-side-encryption: aws:kms
          -
            Sid: DenyInsecureConnections
            Effect: Deny
            Principal: '*'
            Action: s3:*
            Resource: !Sub ${CodePipelineArtifactStoreBucket.Arn}/*
            Condition:
              Bool:
                aws:SecureTransport: false
  CodePipelineServiceRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          -
            Effect: Allow
            Principal:
              Service:
                - codepipeline.amazonaws.com
            Action: sts:AssumeRole
      Path: /
      Policies:
        -
          PolicyName: AWS-CodePipeline-Service-3
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              -
                Effect: Allow
                Action:
                  - codecommit:CancelUploadArchive
                  - codecommit:GetBranch
                  - codecommit:GetCommit
                  - codecommit:GetUploadArchiveStatus
```

```
      - codecommit:UploadArchive
    Resource: 'resource_ARN'
  -
    Effect: Allow
    Action:
      - codedeploy:CreateDeployment
      - codedeploy:GetApplicationRevision
      - codedeploy:GetDeployment
      - codedeploy:GetDeploymentConfig
      - codedeploy:RegisterApplicationRevision
    Resource: 'resource_ARN'
  -
    Effect: Allow
    Action:
      - codebuild:BatchGetBuilds
      - codebuild:StartBuild
    Resource: 'resource_ARN'
  -
    Effect: Allow
    Action:
      - devicefarm:ListProjects
      - devicefarm:ListDevicePools
      - devicefarm:GetRun
      - devicefarm:GetUpload
      - devicefarm:CreateUpload
      - devicefarm:ScheduleRun
    Resource: 'resource_ARN'
  -
    Effect: Allow
    Action:
      - lambda:InvokeFunction
      - lambda:ListFunctions
    Resource: 'resource_ARN'
  -
    Effect: Allow
    Action:
      - iam:PassRole
    Resource: 'resource_ARN'
  -
    Effect: Allow
    Action:
      - elasticbeanstalk:*
      - ec2:*
      - elasticloadbalancing:*
```

```
                - autoscaling:*
                - cloudwatch:*
                - s3:*
                - sns:*
                - cloudformation:*
                - rds:*
                - sqs:*
                - ecs:*
              Resource: 'resource_ARN'
AppPipeline:
  Type: AWS::CodePipeline::Pipeline
  Properties:
    Name: s3-events-pipeline
    RoleArn:
      !GetAtt CodePipelineServiceRole.Arn
    Stages:
      -
        Name: Source
        Actions:
          -
            Name: SourceAction
            ActionTypeId:
              Category: Source
              Owner: AWS
              Version: 1
              Provider: S3
            OutputArtifacts:
              - Name: SourceOutput
            Configuration:
              S3Bucket: !Ref SourceBucket
              S3ObjectKey: !Ref SourceObjectKey
              PollForSourceChanges: false
            RunOrder: 1
      -
        Name: Beta
        Actions:
          -
            Name: BetaAction
            InputArtifacts:
              - Name: SourceOutput
            ActionTypeId:
              Category: Deploy
              Owner: AWS
              Version: 1
```

```
              Provider: CodeDeploy
            Configuration:
              ApplicationName: !Ref ApplicationName
              DeploymentGroupName: !Ref BetaFleet
            RunOrder: 1
    ArtifactStore:
      Type: S3
      Location: !Ref CodePipelineArtifactStoreBucket
 EventRole:
   Type: AWS::IAM::Role
   Properties:
     AssumeRolePolicyDocument:
       Version: 2012-10-17
       Statement:
         -
           Effect: Allow
           Principal:
             Service:
               - events.amazonaws.com
           Action: sts:AssumeRole
     Path: /
     Policies:
       -
         PolicyName: eb-pipeline-execution
         PolicyDocument:
           Version: 2012-10-17
           Statement:
             -
               Effect: Allow
               Action: codepipeline:StartPipelineExecution
               Resource: !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region',
':', !Ref 'AWS::AccountId', ':', !Ref AppPipeline ] ]
 EventRule:
   Type: AWS::Events::Rule
   Properties:
     EventBusName: default
     EventPattern:
       source:
         - aws.s3
       detail-type:
         - Object Created
       detail:
         bucket:
           name:
```

```
              - !Ref SourceBucket
      Name: EnabledS3SourceRule
      State: ENABLED
      Targets:
        -
          Arn:
            !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region', ':', !Ref
 'AWS::AccountId', ':', !Ref AppPipeline ] ]
          RoleArn: !GetAtt EventRole.Arn
          Id: codepipeline-AppPipeline
```

JSON

```
{
    "Parameters": {
        "SourceObjectKey": {
            "Description": "S3 source artifact",
            "Type": "String",
            "Default": "SampleApp_Linux.zip"
        },
        "ApplicationName": {
            "Description": "CodeDeploy application name",
            "Type": "String",
            "Default": "DemoApplication"
        },
        "BetaFleet": {
            "Description": "Fleet configured in CodeDeploy",
            "Type": "String",
            "Default": "DemoFleet"
        }
    },
    "Resources": {
        "SourceBucket": {
            "Type": "AWS::S3::Bucket",
              "Properties": {
                "NotificationConfiguration": {
                    "EventBridgeConfiguration": {
                        "EventBridgeEnabled": true
                    }
                },
                "VersioningConfiguration": {
                    "Status": "Enabled"
```

```
                }
            }
        },
        "CodePipelineArtifactStoreBucket": {
            "Type": "AWS::S3::Bucket"
        },
        "CodePipelineArtifactStoreBucketPolicy": {
            "Type": "AWS::S3::BucketPolicy",
            "Properties": {
                "Bucket": {
                    "Ref": "CodePipelineArtifactStoreBucket"
                },
                "PolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Sid": "DenyUnEncryptedObjectUploads",
                            "Effect": "Deny",
                            "Principal": "*",
                            "Action": "s3:PutObject",
                            "Resource": {
                                "Fn::Join": [
                                    "",
                                    [
                                        {
                                            "Fn::GetAtt": [
                                                "CodePipelineArtifactStoreBucket",
                                                "Arn"
                                            ]
                                        },
                                        "/*"
                                    ]
                                ]
                            },
                            "Condition": {
                                "StringNotEquals": {
                                    "s3:x-amz-server-side-encryption": "aws:kms"
                                }
                            }
                        },
                        {
                            "Sid": "DenyInsecureConnections",
                            "Effect": "Deny",
                            "Principal": "*",
```

```
                        "Action": "s3:*",
                        "Resource": {
                            "Fn::Join": [
                                "",
                                [
                                    {
                                        "Fn::GetAtt": [
                                            "CodePipelineArtifactStoreBucket",
                                            "Arn"
                                        ]
                                    },
                                    "/*"
                                ]
                            ]
                        },
                        "Condition": {
                            "Bool": {
                                "aws:SecureTransport": false
                            }
                        }
                    }
                ]
            }
        }
    },
    "CodePipelineServiceRole": {
        "Type": "AWS::IAM::Role",
        "Properties": {
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {
                            "Service": [
                                "codepipeline.amazonaws.com"
                            ]
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "Path": "/",
            "Policies": [
```

```
                {
                    "PolicyName": "AWS-CodePipeline-Service-3",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "codecommit:CancelUploadArchive",
                                    "codecommit:GetBranch",
                                    "codecommit:GetCommit",
                                    "codecommit:GetUploadArchiveStatus",
                                    "codecommit:UploadArchive"
                                ],
                                "Resource": "resource_ARN"
                            },
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "codedeploy:CreateDeployment",
                                    "codedeploy:GetApplicationRevision",
                                    "codedeploy:GetDeployment",
                                    "codedeploy:GetDeploymentConfig",
                                    "codedeploy:RegisterApplicationRevision"
                                ],
                                "Resource": "resource_ARN"
                            },
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "codebuild:BatchGetBuilds",
                                    "codebuild:StartBuild"
                                ],
                                "Resource": "resource_ARN"
                            },
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "devicefarm:ListProjects",
                                    "devicefarm:ListDevicePools",
                                    "devicefarm:GetRun",
                                    "devicefarm:GetUpload",
                                    "devicefarm:CreateUpload",
                                    "devicefarm:ScheduleRun"
```

```
                                ],
                                "Resource": "resource_ARN"
                            },
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "lambda:InvokeFunction",
                                    "lambda:ListFunctions"
                                ],
                                "Resource": "resource_ARN"
                            },
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "iam:PassRole"
                                ],
                                "Resource": "resource_ARN"
                            },
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "elasticbeanstalk:*",
                                    "ec2:*",
                                    "elasticloadbalancing:*",
                                    "autoscaling:*",
                                    "cloudwatch:*",
                                    "s3:*",
                                    "sns:*",
                                    "cloudformation:*",
                                    "rds:*",
                                    "sqs:*",
                                    "ecs:*"
                                ],
                                "Resource": "resource_ARN"
                            }
                        ]
                    }
                }
            ]
        }
    },
    "AppPipeline": {
        "Type": "AWS::CodePipeline::Pipeline",
        "Properties": {
```

```
                "Name": "s3-events-pipeline",
                "RoleArn": {
                    "Fn::GetAtt": [
                        "CodePipelineServiceRole",
                        "Arn"
                    ]
                },
                "Stages": [
                    {
                        "Name": "Source",
                        "Actions": [
                            {
                                "Name": "SourceAction",
                                "ActionTypeId": {
                                    "Category": "Source",
                                    "Owner": "AWS",
                                    "Version": 1,
                                    "Provider": "S3"
                                },
                                "OutputArtifacts": [
                                    {
                                        "Name": "SourceOutput"
                                    }
                                ],
                                "Configuration": {
                                    "S3Bucket": {
                                        "Ref": "SourceBucket"
                                    },
                                    "S3ObjectKey": {
                                        "Ref": "SourceObjectKey"
                                    },
                                    "PollForSourceChanges": false
                                },
                                "RunOrder": 1
                            }
                        ]
                    },
                    {
                        "Name": "Beta",
                        "Actions": [
                            {
                                "Name": "BetaAction",
                                "InputArtifacts": [
                                    {
```

```
                                    "Name": "SourceOutput"
                                }
                            ],
                            "ActionTypeId": {
                                "Category": "Deploy",
                                "Owner": "AWS",
                                "Version": 1,
                                "Provider": "CodeDeploy"
                            },
                            "Configuration": {
                                "ApplicationName": {
                                    "Ref": "ApplicationName"
                                },
                                "DeploymentGroupName": {
                                    "Ref": "BetaFleet"
                                }
                            },
                            "RunOrder": 1
                        }
                    ]
                }
            ],
            "ArtifactStore": {
                "Type": "S3",
                "Location": {
                    "Ref": "CodePipelineArtifactStoreBucket"
                }
            }
        }
    },
    "EventRole": {
        "Type": "AWS::IAM::Role",
        "Properties": {
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {
                            "Service": [
                                "events.amazonaws.com"
                            ]
                        },
                        "Action": "sts:AssumeRole"
```

```
                    }
                ]
            },
            "Path": "/",
            "Policies": [
                {
                    "PolicyName": "eb-pipeline-execution",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": "codepipeline:StartPipelineExecution",
                                "Resource": {
                                    "Fn::Join": [
                                        "",
                                        [
                                            "arn:aws:codepipeline:",
                                            {
                                                "Ref": "AWS::Region"
                                            },
                                            ":",
                                            {
                                                "Ref": "AWS::AccountId"
                                            },
                                            ":",
                                            {
                                                "Ref": "AppPipeline"
                                            }
                                        ]
                                    ]
                                }
                            }
                        ]
                    }
                }
            ]
        }
    },
    "EventRule": {
        "Type": "AWS::Events::Rule",

        "Properties": {
            "EventBusName": "default",
```

```
          "EventPattern": {
              "source": [
            "aws.s3"
          ],
                "detail-type": [
        "Object Created"
          ],
          "detail": {
        "bucket": {
                        "name": [
                            {
                            "Ref": "SourceBucket"
                            }
                        ]
                    }
                }
    },
    "Name": "EnabledS3SourceRule",
          "State": "ENABLED",
                  "Targets": [
                      {
                          "Arn": {
                              "Fn::Join": [
                                  "",
                                  [
                                      "arn:aws:codepipeline:",
                                      {
                                          "Ref": "AWS::Region"
                                      },
                                      ":",
                                      {
                                          "Ref": "AWS::AccountId"
                                      },
                                      ":",
                                      {
                                          "Ref": "AppPipeline"
                                      }
                                  ]
                              ]
                          },
                          "RoleArn": {
                              "Fn::GetAtt": [
                                  "EventRole",
                                  "Arn"
```

```
                            ]
                        },
                        "Id": "codepipeline-AppPipeline"
                    }
                ]
            }
        }
    }
    }
}
```

Migra le pipeline di polling con un sorgente e un trail S3 CloudTrail

Per una pipeline con una fonte Amazon S3, modifica la pipeline in modo da automatizzare il rilevamento delle modifiche. EventBridge Scegli tra i seguenti metodi per implementare la migrazione:

- Console: Migrazione delle pipeline di polling (o sorgente Amazon CodeCommit S3) (console)
- CLI: Migra le pipeline di polling con un codice sorgente e trail CloudTrail (CLI) S3
- AWS CloudFormation: Migra le pipeline di polling con una sorgente e un trail S3 (modello) CloudTrail AWS CloudFormation

Migra le pipeline di polling con un codice sorgente e trail CloudTrail (CLI) S3

Segui questi passaggi per modificare una pipeline che utilizza il polling (controlli periodici) per utilizzare invece un evento. EventBridge Se vuoi creare una pipeline, consulta Creare una pipeline in CodePipeline.

Per creare una pipeline basata sugli eventi con Amazon S3, devi modificare il `PollForSourceChanges` parametro della pipeline e quindi creare le seguenti risorse:

- AWS CloudTrail policy trail, bucket e bucket che Amazon S3 può utilizzare per registrare gli eventi.
- EventBridge evento
- Ruolo IAM per consentire all' EventBridge evento di avviare la tua pipeline

Per creare un AWS CloudTrail percorso e abilitare la registrazione

Per utilizzare il AWS CLI per creare una traccia, chiamate il create-trail comando, specificando:

- Il nome del trail.
- Il bucket nel quale hai già applicato le policy del bucket per AWS CloudTrail.

Per ulteriori informazioni, vedere Creazione di un percorso con l'interfaccia a riga di AWS comando.

1. Chiama il comando create-trail e includi i parametri `--name` e `--s3-bucket-name`.

   Perché occorre apportare questa modifica? Questo crea il CloudTrail percorso richiesto per il tuo bucket di origine S3.

   Il comando seguente utilizza `--name` e `--s3-bucket-name` per creare un trail denominato `my-trail` e un bucket denominato `myBucket`.

   ```
   aws cloudtrail create-trail --name my-trail --s3-bucket-name myBucket
   ```

2. Chiama il comando start-logging e includi il parametro `--name`.

   Perché sto apportando questa modifica? Questo comando avvia la CloudTrail registrazione per il bucket di origine e invia gli eventi a. EventBridge

   Esempio:

   Il comando seguente utilizza `--name` per avviare la registrazione su un trail denominato `my-trail`.

   ```
   aws cloudtrail start-logging --name my-trail
   ```

3. Chiama il comando put-event-selectors e includi i parametri `--trail-name` e `--event-selectors`. Usa i selettori di eventi per specificare che desideri che il tuo trail registri gli eventi di dati per il tuo bucket di origine e invii gli eventi alla regola. EventBridge

   Perché sto apportando questa modifica? Questo comando filtra gli eventi.

   Esempio:

   Il comando seguente utilizza `--trail-name` e `--event-selectors` per specificare gli eventi dati per un bucket e un prefisso di origine denominati `myBucket/myFolder`.

   ```
   aws cloudtrail put-event-selectors --trail-name my-trail --event-selectors
    '[{ "ReadWriteType": "WriteOnly", "IncludeManagementEvents":false,
   ```

```
 "DataResources": [{ "Type": "AWS::S3::Object", "Values": ["arn:aws:s3:::myBucket/
myFolder/file.zip"] }] }]'
```

Per creare una EventBridge regola con Amazon S3 come origine dell'evento e CodePipeline come destinazione e applicare la politica delle autorizzazioni

1. Concedi le autorizzazioni EventBridge da utilizzare per CodePipeline richiamare la regola. Per ulteriori informazioni, consulta Utilizzo delle politiche basate sulle risorse per Amazon. EventBridge

   a. Usa l'esempio seguente per creare la politica di fiducia che consente di EventBridge assumere il ruolo di servizio. Denominalo `trustpolicyforEB.json`.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Principal": {
                   "Service": "events.amazonaws.com"
               },
               "Action": "sts:AssumeRole"
           }
       ]
   }
   ```

   b. Utilizza il seguente comando per creare il ruolo `Role-for-MyRule` e collegare la policy di attendibilità.

   Perché occorre apportare questa modifica? L'aggiunta di questa politica di fiducia al ruolo crea le autorizzazioni per EventBridge.

   ```
   aws iam create-role --role-name Role-for-MyRule --assume-role-policy-document
    file://trustpolicyforEB.json
   ```

   c. Crea il JSON della policy delle autorizzazioni, come mostrato qui per la pipeline denominata `MyFirstPipeline`. Denomina la policy delle autorizzazioni `permissionspolicyforEB.json`.

   ```
   {
   ```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:StartPipelineExecution"
            ],
            "Resource": [
                "arn:aws:codepipeline:us-west-2:80398EXAMPLE:MyFirstPipeline"
            ]
        }
    ]
}
```

d. Utilizza il comando seguente per collegare la nuova policy delle autorizzazioni `CodePipeline-Permissions-Policy-for-EB` al ruolo `Role-for-MyRule` che hai creato.

```
aws iam put-role-policy --role-name Role-for-MyRule --policy-name CodePipeline-
Permissions-Policy-For-EB --policy-document file://permissionspolicyforEB.json
```

2. Richiama il comando put-rule e includi i parametri `--name`, `--event-pattern` e `--role-arn`.

   Il seguente comando di esempio crea una regola denominata `MyS3SourceRule`.

```
aws events put-rule --name "MyS3SourceRule" --event-pattern "{\"source\":
[\"aws.s3\"],\"detail-type\":[\"AWS API Call via CloudTrail\"],\"detail\":
{\"eventSource\":[\"s3.amazonaws.com\"],\"eventName\":[\"CopyObject\",\"PutObject
\",\"CompleteMultipartUpload\"],\"requestParameters\":{\"bucketName\":[\"my-bucket
\"],\"key\":[\"my-key\"]}}}
 --role-arn "arn:aws:iam::ACCOUNT_ID:role/Role-for-MyRule"
```

3. Per aggiungere CodePipeline come destinazione, chiamate il put-targets comando e includete i `--targets` parametri `--rule` and.

   Il comando seguente specifica che per la regola denominata `MyS3SourceRule`, la destinazione `Id` è composta dal numero uno, per indicare che in un elenco di destinazioni per la regola questa è la destinazione 1. Il comando specifica anche un esempio di ARN per la pipeline. La pipeline si avvia quando si verifica una modifica nel repository.

```
aws events put-targets --rule MyS3SourceRule --targets
 Id=1,Arn=arn:aws:codepipeline:us-west-2:80398EXAMPLE:TestPipeline
```

Per modificare il parametro della PollForSourceChanges pipeline

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

1. Esegui il comando get-pipeline per copiare la struttura della pipeline in un file JSON. Ad esempio, per una pipeline denominata `MyFirstPipeline`, esegui il seguente comando:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   ```

   Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.

2. Apri il file JSON in qualsiasi editor di testo normale e modifica la fase di origine modificando il parametro `PollForSourceChanges` per un bucket denominato `storage-bucketfalse` come mostrato nell'esempio seguente.

   Perché occorre apportare questa modifica? L'impostazione del parametro su `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

   ```
   "configuration": {
       "S3Bucket": "storage-bucket",
       "PollForSourceChanges": "false",
       "S3ObjectKey": "index.zip"
   },
   ```

3. Se stai utilizzando la struttura della pipeline recuperata tramite il comando get-pipeline, devi rimuovere le righe `metadata` dal file JSON. In caso contrario, il comando update-pipeline non è in grado di utilizzarlo. Rimuovi le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`.

   Ad esempio, rimuovere dalla struttura le seguenti righe:

   ```
   "metadata": {
       "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
       "created": "date",
       "updated": "date"
   },
   ```

   Salvare il file.

4. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline:

   > **Important**
   >
   > Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

   ```
   aws codepipeline update-pipeline --cli-input-json file://pipeline.json
   ```

   Questo comando restituisce l'intera struttura della pipeline modificata.

   > **Note**
   >
   > Il comando update-pipeline arresta la pipeline. Se è in corso di elaborazione una versione durante l'esecuzione del comando update-pipeline, tale elaborazione viene arrestata. Per elaborare tale versione utilizzando la pipeline aggiornata, devi avviare manualmente la pipeline. Utilizza il comando start-pipeline-execution per avviare manualmente la pipeline.

Migra le pipeline di polling con una sorgente e un trail S3 (modello) CloudTrail AWS CloudFormation

Utilizza questi passaggi per modificare la tua pipeline con una fonte Amazon S3, dal polling al rilevamento delle modifiche basato sugli eventi.

Per creare una pipeline basata sugli eventi con Amazon S3, devi modificare il `PollForSourceChanges` parametro della pipeline e quindi aggiungere le seguenti risorse al modello:

- EventBridge richiede la registrazione di tutti gli eventi di Amazon S3. È necessario creare una policy AWS CloudTrail trail, bucket e bucket che Amazon S3 possa utilizzare per registrare gli eventi che si verificano. Per ulteriori informazioni, consulta Registrazione degli eventi relativi ai dati per i sentieri e Registrazione degli eventi di gestione dei percorsi.
- EventBridge regola e ruolo IAM per consentire a questo evento di avviare la nostra pipeline.

Se lo utilizzi AWS CloudFormation per creare e gestire le tue pipeline, il tuo modello include contenuti come i seguenti.

> **Note**
>
> La proprietà `Configuration` nella fase di origine chiamata `PollForSourceChanges`. Se il modello non include questa proprietà, allora `PollForSourceChanges` è impostato su `true` per impostazione predefinita.

YAML

```
AppPipeline:
  Type: AWS::CodePipeline::Pipeline
  Properties:
    RoleArn: !GetAtt CodePipelineServiceRole.Arn
    Stages:
      -
        Name: Source
        Actions:
          -
            Name: SourceAction
            ActionTypeId:
```

```
                Category: Source
                Owner: AWS
                Version: 1
                Provider: S3
              OutputArtifacts:
                -
                  Name: SourceOutput
              Configuration:
                S3Bucket: !Ref SourceBucket
                S3ObjectKey: !Ref S3SourceObjectKey
                PollForSourceChanges: true
              RunOrder: 1


...
```

JSON

```
        "AppPipeline": {
            "Type": "AWS::CodePipeline::Pipeline",
            "Properties": {
                "RoleArn": {
                    "Fn::GetAtt": ["CodePipelineServiceRole", "Arn"]
                },
                "Stages": [
                    {
                        "Name": "Source",
                        "Actions": [
                            {
                                "Name": "SourceAction",
                                "ActionTypeId": {
                                    "Category": "Source",
                                    "Owner": "AWS",
                                    "Version": 1,
                                    "Provider": "S3"
                                },
                                "OutputArtifacts": [
                                    {
                                        "Name": "SourceOutput"
                                    }
                                ],
                                "Configuration": {
                                    "S3Bucket": {
```

```
                                    "Ref": "SourceBucket"
                                },
                                "S3ObjectKey": {
                                    "Ref": "SourceObjectKey"
                                },
                                "PollForSourceChanges": true
                            },
                            "RunOrder": 1
                        }
                    ]
                },


...
```

Per creare una EventBridge regola con Amazon S3 come origine dell'evento e CodePipeline come destinazione e applicare la politica delle autorizzazioni

1. Nel modello, sotto`Resources`, utilizza la `AWS::IAM::Role` AWS CloudFormation risorsa per configurare il ruolo IAM che consente all'evento di avviare la pipeline. Questa voce crea un ruolo che utilizza due policy:

   - La prima policy consente di assumere quel ruolo.
   - La seconda policy fornisce le autorizzazioni per avviare la pipeline.

   Perché occorre apportare questa modifica? L'aggiunta di `AWS::IAM::Role` risorse consente AWS CloudFormation di creare autorizzazioni per. EventBridge Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

   ```
   EventRole:
     Type: AWS::IAM::Role
     Properties:
       AssumeRolePolicyDocument:
         Version: 2012-10-17
         Statement:
           -
             Effect: Allow
             Principal:
   ```

```
              Service:
                - events.amazonaws.com
            Action: sts:AssumeRole
      Path: /
      Policies:
        -
          PolicyName: eb-pipeline-execution
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              -
                Effect: Allow
                Action: codepipeline:StartPipelineExecution
                Resource: !Join [ '', [ 'arn:aws:codepipeline:', !Ref
 'AWS::Region', ':', !Ref 'AWS::AccountId', ':', !Ref AppPipeline ] ]


...
```

JSON

```
 "EventRole": {
   "Type": "AWS::IAM::Role",
   "Properties": {
     "AssumeRolePolicyDocument": {
       "Version": "2012-10-17",
       "Statement": [
         {
           "Effect": "Allow",
           "Principal": {
             "Service": [
               "events.amazonaws.com"
             ]
           },
           "Action": "sts:AssumeRole"
         }
       ]
     },
     "Path": "/",
     "Policies": [
       {
         "PolicyName": "eb-pipeline-execution",
         "PolicyDocument": {
```

```
            "Version": "2012-10-17",
            "Statement": [
              {
                "Effect": "Allow",
                "Action": "codepipeline:StartPipelineExecution",
                "Resource": {
                  "Fn::Join": [
                    "",
                    [
                      "arn:aws:codepipeline:",
                      {
                        "Ref": "AWS::Region"
                      },
                      ":",
                      {
                        "Ref": "AWS::AccountId"
                      },
                      ":",
                      {
                        "Ref": "AppPipeline"
                      }
                    ]
                  ]

...
```

2. Usa la `AWS::Events::Rule` AWS CloudFormation risorsa per aggiungere una EventBridge regola. Questo modello di eventi crea un evento che monitora `CopyObject` `PutObject` e `CompleteMultipartUpload` sul tuo bucket di origine Amazon S3. Inoltre, include una destinazione della pipeline. Quando si verifica `CopyObject`, `PutObject` o `CompleteMultipartUpload`, questa regola richiama `StartPipelineExecution` sulla pipeline di destinazione.

   Perché occorre apportare questa modifica? L'aggiunta della `AWS::Events::Rule` risorsa consente di AWS CloudFormation creare l'evento. Questa risorsa viene aggiunta al tuo AWS CloudFormation stack.

   YAML

```
  EventRule:
    Type: AWS::Events::Rule
    Properties:
```

```
      EventPattern:
        source:
          - aws.s3
        detail-type:
          - 'AWS API Call via CloudTrail'
        detail:
          eventSource:
            - s3.amazonaws.com
          eventName:
            - CopyObject
            - PutObject
            - CompleteMultipartUpload
          requestParameters:
            bucketName:
              - !Ref SourceBucket
            key:
              - !Ref SourceObjectKey
      Targets:
        -
          Arn:
            !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region', ':', !Ref
 'AWS::AccountId', ':', !Ref AppPipeline ] ]
          RoleArn: !GetAtt EventRole.Arn
          Id: codepipeline-AppPipeline


...
```

JSON

```
 "EventRule": {
   "Type": "AWS::Events::Rule",
   "Properties": {
     "EventPattern": {
       "source": [
         "aws.s3"
       ],
       "detail-type": [
         "AWS API Call via CloudTrail"
       ],
       "detail": {
         "eventSource": [
```

```
            "s3.amazonaws.com"
          ],
          "eventName": [
            "CopyObject",
            "PutObject",
            "CompleteMultipartUpload"
          ],
          "requestParameters": {
            "bucketName": [
              {
                "Ref": "SourceBucket"
              }
            ],
            "key": [
              {
                "Ref": "SourceObjectKey"
              }
            ]
          }
        }
      },
      "Targets": [
        {
          "Arn": {
            "Fn::Join": [
              "",
              [
                "arn:aws:codepipeline:",
                {
                  "Ref": "AWS::Region"
                },
                ":",
                {
                  "Ref": "AWS::AccountId"
                },
                ":",
                {
                  "Ref": "AppPipeline"
                }
              ]
            ]
          },
          "RoleArn": {
            "Fn::GetAtt": [
```

```
              "EventRole",
              "Arn"
            ]
          },
          "Id": "codepipeline-AppPipeline"
        }
      ]
    }
  }
},

...
```

3. Aggiungere questo snippet di codice al primo modello per consentire la funzionalità tra stack:

   YAML

   ```
   Outputs:
     SourceBucketARN:
       Description: "S3 bucket ARN that Cloudtrail will use"
       Value: !GetAtt SourceBucket.Arn
       Export:
         Name: SourceBucketARN
   ```

   JSON

   ```
     "Outputs" : {
       "SourceBucketARN" : {
         "Description" : "S3 bucket ARN that Cloudtrail will use",
         "Value" : { "Fn::GetAtt": ["SourceBucket", "Arn"] },
         "Export" : {
           "Name" : "SourceBucketARN"
         }
       }

   ...
   ```

4. Salva il modello aggiornato sul computer locale e apri la AWS CloudFormation console.
5. Seleziona lo stack e scegli Create Change Set for Current Stack (Crea set di modifiche per lo stack corrente).

6. Caricare il modello aggiornato e quindi visualizzare le modifiche elencate in AWS CloudFormation. Queste sono le modifiche che verranno apportate allo stack. Le nuove risorse dovrebbero essere visibili nell'elenco.
7. Scegliere Execute (Esegui).

Per modificare i parametri della PollForSourceChanges pipeline

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

- Nel modello, modifica `PollForSourceChanges` in `false`. Se non hai incluso `PollForSourceChanges` nella definizione della pipeline, aggiungilo e impostalo su `false`.

  Perché occorre apportare questa modifica? La modifica di `PollForSourceChanges` in `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

  YAML

```
Name: Source
Actions:
  -
    Name: SourceAction
    ActionTypeId:
      Category: Source
      Owner: AWS
      Version: 1
      Provider: S3
    OutputArtifacts:
      - Name: SourceOutput
    Configuration:
      S3Bucket: !Ref SourceBucket
```

```
                S3ObjectKey: !Ref SourceObjectKey
                PollForSourceChanges: false
              RunOrder: 1
```

JSON

```
{
   "Name": "SourceAction",
   "ActionTypeId": {
     "Category": "Source",
     "Owner": "AWS",
     "Version": 1,
     "Provider": "S3"
   },
   "OutputArtifacts": [
     {
       "Name": "SourceOutput"
     }
   ],
   "Configuration": {
     "S3Bucket": {
       "Ref": "SourceBucket"
     },
     "S3ObjectKey": {
       "Ref": "SourceObjectKey"
     },
     "PollForSourceChanges": false
   },
   "RunOrder": 1
 }
```

Per creare un secondo modello per le risorse della tua pipeline Amazon S3 CloudTrail

- In un modello separato, sotto`Resources`, utilizza le `AWS::CloudTrail::Trail` AWS CloudFormation risorse `AWS::S3::BucketAWS::S3::BucketPolicy`, e per fornire una definizione e un percorso semplici per il bucket. CloudTrail

  Perché sto apportando questa modifica? Dato l'attuale limite di cinque percorsi per account, il CloudTrail percorso deve essere creato e gestito separatamente. (Vedi Limiti in AWS CloudTrail.) Tuttavia, puoi includere molti bucket Amazon S3 in un singolo trail, in modo da poter

creare il trail una sola volta e poi aggiungere bucket Amazon S3 per altre pipeline, se necessario. Incolla quanto segue nel secondo file di modello di esempio.

YAML

```
###########################################################################################
# Prerequisites:
#   - S3 SourceBucket and SourceObjectKey must exist
###########################################################################################

Parameters:
  SourceObjectKey:
    Description: 'S3 source artifact'
    Type: String
    Default: SampleApp_Linux.zip

Resources:
  AWSCloudTrailBucketPolicy:
    Type: AWS::S3::BucketPolicy
    Properties:
      Bucket: !Ref AWSCloudTrailBucket
      PolicyDocument:
        Version: 2012-10-17
        Statement:
          -
            Sid: AWSCloudTrailAclCheck
            Effect: Allow
            Principal:
              Service:
                - cloudtrail.amazonaws.com
            Action: s3:GetBucketAcl
            Resource: !GetAtt AWSCloudTrailBucket.Arn
          -
            Sid: AWSCloudTrailWrite
            Effect: Allow
            Principal:
              Service:
                - cloudtrail.amazonaws.com
            Action: s3:PutObject
            Resource: !Join [ '', [ !GetAtt AWSCloudTrailBucket.Arn, '/
AWSLogs/', !Ref 'AWS::AccountId', '/*' ] ]
            Condition:
              StringEquals:
```

```
                s3:x-amz-acl: bucket-owner-full-control
  AWSCloudTrailBucket:
    Type: AWS::S3::Bucket
    DeletionPolicy: Retain
  AwsCloudTrail:
    DependsOn:
      - AWSCloudTrailBucketPolicy
    Type: AWS::CloudTrail::Trail
    Properties:
      S3BucketName: !Ref AWSCloudTrailBucket
      EventSelectors:
        -
          DataResources:
            -
              Type: AWS::S3::Object
              Values:
                - !Join [ '', [ !ImportValue SourceBucketARN, '/', !Ref
 SourceObjectKey ] ]
          ReadWriteType: WriteOnly
          IncludeManagementEvents: false
      IncludeGlobalServiceEvents: true
      IsLogging: true
      IsMultiRegionTrail: true


...
```

JSON

```
{
  "Parameters": {
    "SourceObjectKey": {
      "Description": "S3 source artifact",
      "Type": "String",
      "Default": "SampleApp_Linux.zip"
    }
  },
  "Resources": {
    "AWSCloudTrailBucket": {
      "Type": "AWS::S3::Bucket",
        "DeletionPolicy": "Retain"
    },
    "AWSCloudTrailBucketPolicy": {
```

```
      "Type": "AWS::S3::BucketPolicy",
      "Properties": {
        "Bucket": {
          "Ref": "AWSCloudTrailBucket"
        },
        "PolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "AWSCloudTrailAclCheck",
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "cloudtrail.amazonaws.com"
                ]
              },
              "Action": "s3:GetBucketAcl",
              "Resource": {
                "Fn::GetAtt": [
                  "AWSCloudTrailBucket",
                  "Arn"
                ]
              }
            },
            {
              "Sid": "AWSCloudTrailWrite",
              "Effect": "Allow",
              "Principal": {
                "Service": [
                  "cloudtrail.amazonaws.com"
                ]
              },
              "Action": "s3:PutObject",
              "Resource": {
                "Fn::Join": [
                  "",
                  [
                    {
                      "Fn::GetAtt": [
                        "AWSCloudTrailBucket",
                        "Arn"
                      ]
                    },
                    "/AWSLogs/",
```

```
                {
                  "Ref": "AWS::AccountId"
                },
                "/*"
              ]
            ]
          },
          "Condition": {
            "StringEquals": {
              "s3:x-amz-acl": "bucket-owner-full-control"
            }
          }
        }
      ]
    }
  }
},
"AwsCloudTrail": {
  "DependsOn": [
    "AWSCloudTrailBucketPolicy"
  ],
  "Type": "AWS::CloudTrail::Trail",
  "Properties": {
    "S3BucketName": {
      "Ref": "AWSCloudTrailBucket"
    },
    "EventSelectors": [
      {
        "DataResources": [
          {
            "Type": "AWS::S3::Object",
            "Values": [
              {
                "Fn::Join": [
                  "",
                  [
                    {
                      "Fn::ImportValue": "SourceBucketARN"
                    },
                    "/",
                    {
                      "Ref": "SourceObjectKey"
                    }
                  ]
```

```
                    ]
                  }
                ]
              }
            ],
            "ReadWriteType": "WriteOnly",
            "IncludeManagementEvents": false
          }
        ],
        "IncludeGlobalServiceEvents": true,
        "IsLogging": true,
        "IsMultiRegionTrail": true
      }
    }
  }
}

...
```

Example

Quando utilizzi AWS CloudFormation per creare queste risorse, la tua pipeline viene attivata quando i file nel tuo repository vengono creati o aggiornati.

> **Note**
>
> Non è finita qui. Sebbene la pipeline sia stata creata, è necessario creare un secondo AWS CloudFormation modello per la pipeline Amazon S3. Se non crei il secondo modello, la pipeline non avrà la funzionalità di rilevamento delle modifiche.

YAML

```
Resources:
  SourceBucket:
    Type: AWS::S3::Bucket
    Properties:
      VersioningConfiguration:
        Status: Enabled
  CodePipelineArtifactStoreBucket:
    Type: AWS::S3::Bucket
```

```
  CodePipelineArtifactStoreBucketPolicy:
    Type: AWS::S3::BucketPolicy
    Properties:
      Bucket: !Ref CodePipelineArtifactStoreBucket
      PolicyDocument:
        Version: 2012-10-17
        Statement:
          -
            Sid: DenyUnEncryptedObjectUploads
            Effect: Deny
            Principal: '*'
            Action: s3:PutObject
            Resource: !Join [ '', [ !GetAtt CodePipelineArtifactStoreBucket.Arn, '/
*' ] ]
            Condition:
              StringNotEquals:
                s3:x-amz-server-side-encryption: aws:kms
          -
            Sid: DenyInsecureConnections
            Effect: Deny
            Principal: '*'
            Action: s3:*
            Resource: !Join [ '', [ !GetAtt CodePipelineArtifactStoreBucket.Arn, '/
*' ] ]
            Condition:
              Bool:
                aws:SecureTransport: false
  CodePipelineServiceRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          -
            Effect: Allow
            Principal:
              Service:
                - codepipeline.amazonaws.com
            Action: sts:AssumeRole
      Path: /
      Policies:
        -
          PolicyName: AWS-CodePipeline-Service-3
          PolicyDocument:
```

```
            Version: 2012-10-17
            Statement:
              -
                Effect: Allow
                Action:
                  - codecommit:CancelUploadArchive
                  - codecommit:GetBranch
                  - codecommit:GetCommit
                  - codecommit:GetUploadArchiveStatus
                  - codecommit:UploadArchive
                Resource: 'resource_ARN'
              -
                Effect: Allow
                Action:
                  - codedeploy:CreateDeployment
                  - codedeploy:GetApplicationRevision
                  - codedeploy:GetDeployment
                  - codedeploy:GetDeploymentConfig
                  - codedeploy:RegisterApplicationRevision
                Resource: 'resource_ARN'
              -
                Effect: Allow
                Action:
                  - codebuild:BatchGetBuilds
                  - codebuild:StartBuild
                Resource: 'resource_ARN'
              -
                Effect: Allow
                Action:
                  - devicefarm:ListProjects
                  - devicefarm:ListDevicePools
                  - devicefarm:GetRun
                  - devicefarm:GetUpload
                  - devicefarm:CreateUpload
                  - devicefarm:ScheduleRun
                Resource: 'resource_ARN'
              -
                Effect: Allow
                Action:
                  - lambda:InvokeFunction
                  - lambda:ListFunctions
                Resource: 'resource_ARN'
              -
                Effect: Allow
```

```
              Action:
                - iam:PassRole
              Resource: 'resource_ARN'
            -
              Effect: Allow
              Action:
                - elasticbeanstalk:*
                - ec2:*
                - elasticloadbalancing:*
                - autoscaling:*
                - cloudwatch:*
                - s3:*
                - sns:*
                - cloudformation:*
                - rds:*
                - sqs:*
                - ecs:*
              Resource: 'resource_ARN'
  AppPipeline:
    Type: AWS::CodePipeline::Pipeline
    Properties:
      Name: s3-events-pipeline
      RoleArn:
        !GetAtt CodePipelineServiceRole.Arn
      Stages:
        -
          Name: Source
          Actions:
            -
              Name: SourceAction
              ActionTypeId:
                Category: Source
                Owner: AWS
                Version: 1
                Provider: S3
              OutputArtifacts:
                - Name: SourceOutput
              Configuration:
                S3Bucket: !Ref SourceBucket
                S3ObjectKey: !Ref SourceObjectKey
                PollForSourceChanges: false
              RunOrder: 1
        -
          Name: Beta
```

```
          Actions:
            -
              Name: BetaAction
              InputArtifacts:
                - Name: SourceOutput
              ActionTypeId:
                Category: Deploy
                Owner: AWS
                Version: 1
                Provider: CodeDeploy
              Configuration:
                ApplicationName: !Ref ApplicationName
                DeploymentGroupName: !Ref BetaFleet
              RunOrder: 1
      ArtifactStore:
        Type: S3
        Location: !Ref CodePipelineArtifactStoreBucket
  EventRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          -
            Effect: Allow
            Principal:
              Service:
                - events.amazonaws.com
            Action: sts:AssumeRole
      Path: /
      Policies:
        -
          PolicyName: eb-pipeline-execution
          PolicyDocument:
            Version: 2012-10-17
            Statement:
              -
                Effect: Allow
                Action: codepipeline:StartPipelineExecution
                Resource: !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region',
':', !Ref 'AWS::AccountId', ':', !Ref AppPipeline ] ]
  EventRule:
    Type: AWS::Events::Rule
    Properties:
```

```
      EventPattern:
        source:
          - aws.s3
        detail-type:
          - 'AWS API Call via CloudTrail'
        detail:
          eventSource:
            - s3.amazonaws.com
          eventName:
            - PutObject
            - CompleteMultipartUpload
          resources:
            ARN:
              - !Join [ '', [ !GetAtt SourceBucket.Arn, '/', !Ref
 SourceObjectKey ] ]
      Targets:
        -
          Arn:
            !Join [ '', [ 'arn:aws:codepipeline:', !Ref 'AWS::Region', ':', !Ref
 'AWS::AccountId', ':', !Ref AppPipeline ] ]
          RoleArn: !GetAtt EventRole.Arn
          Id: codepipeline-AppPipeline

Outputs:
  SourceBucketARN:
    Description: "S3 bucket ARN that Cloudtrail will use"
    Value: !GetAtt SourceBucket.Arn
    Export:
      Name: SourceBucketARN
```

JSON

```
    "Resources": {
        "SourceBucket": {
            "Type": "AWS::S3::Bucket",
            "Properties": {
                "VersioningConfiguration": {
                    "Status": "Enabled"
                }
            }
        },
        "CodePipelineArtifactStoreBucket": {
            "Type": "AWS::S3::Bucket"
```

```
        },
        "CodePipelineArtifactStoreBucketPolicy": {
            "Type": "AWS::S3::BucketPolicy",
            "Properties": {
                "Bucket": {
                    "Ref": "CodePipelineArtifactStoreBucket"
                },
                "PolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Sid": "DenyUnEncryptedObjectUploads",
                            "Effect": "Deny",
                            "Principal": "*",
                            "Action": "s3:PutObject",
                            "Resource": {
                                "Fn::Join": [
                                    "",
                                    [
                                        {
                                            "Fn::GetAtt": [
                                                "CodePipelineArtifactStoreBucket",
                                                "Arn"
                                            ]
                                        },
                                        "/*"
                                    ]
                                ]
                            },
                            "Condition": {
                                "StringNotEquals": {
                                    "s3:x-amz-server-side-encryption": "aws:kms"
                                }
                            }
                        },
                        {
                            "Sid": "DenyInsecureConnections",
                            "Effect": "Deny",
                            "Principal": "*",
                            "Action": "s3:*",
                            "Resource": {
                                "Fn::Join": [
                                    "",
                                    [
```

```
                                    {
                                        "Fn::GetAtt": [
                                            "CodePipelineArtifactStoreBucket",
                                            "Arn"
                                        ]
                                    },
                                    "/*"
                                ]
                            ]
                        },
                        "Condition": {
                            "Bool": {
                                "aws:SecureTransport": false
                            }
                        }
                    }
                ]
            }
        }
    },
    "CodePipelineServiceRole": {
        "Type": "AWS::IAM::Role",
        "Properties": {
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {
                            "Service": [
                                "codepipeline.amazonaws.com"
                            ]
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "Path": "/",
            "Policies": [
                {
                    "PolicyName": "AWS-CodePipeline-Service-3",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
```

```
                {
                    "Effect": "Allow",
                    "Action": [
                        "codecommit:CancelUploadArchive",
                        "codecommit:GetBranch",
                        "codecommit:GetCommit",
                        "codecommit:GetUploadArchiveStatus",
                        "codecommit:UploadArchive"
                    ],
                    "Resource": "resource_ARN"
                },
                {
                    "Effect": "Allow",
                    "Action": [
                        "codedeploy:CreateDeployment",
                        "codedeploy:GetApplicationRevision",
                        "codedeploy:GetDeployment",
                        "codedeploy:GetDeploymentConfig",
                        "codedeploy:RegisterApplicationRevision"
                    ],
                    "Resource": "resource_ARN"
                },
                {
                    "Effect": "Allow",
                    "Action": [
                        "codebuild:BatchGetBuilds",
                        "codebuild:StartBuild"
                    ],
                    "Resource": "resource_ARN"
                },
                {
                    "Effect": "Allow",
                    "Action": [
                        "devicefarm:ListProjects",
                        "devicefarm:ListDevicePools",
                        "devicefarm:GetRun",
                        "devicefarm:GetUpload",
                        "devicefarm:CreateUpload",
                        "devicefarm:ScheduleRun"
                    ],
                    "Resource": "resource_ARN"
                },
                {
                    "Effect": "Allow",
```

```
                                "Action": [
                                    "lambda:InvokeFunction",
                                    "lambda:ListFunctions"
                                ],
                                "Resource": "resource_ARN"
                            },
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "iam:PassRole"
                                ],
                                "Resource": "resource_ARN"
                            },
                            {
                                "Effect": "Allow",
                                "Action": [
                                    "elasticbeanstalk:*",
                                    "ec2:*",
                                    "elasticloadbalancing:*",
                                    "autoscaling:*",
                                    "cloudwatch:*",
                                    "s3:*",
                                    "sns:*",
                                    "cloudformation:*",
                                    "rds:*",
                                    "sqs:*",
                                    "ecs:*"
                                ],
                                "Resource": "resource_ARN"
                            }
                        ]
                    }
                }
            ]
        }
    },
    "AppPipeline": {
        "Type": "AWS::CodePipeline::Pipeline",
        "Properties": {
            "Name": "s3-events-pipeline",
            "RoleArn": {
                "Fn::GetAtt": [
                    "CodePipelineServiceRole",
                    "Arn"
```

```
            ]
        },
        "Stages": [
            {
                "Name": "Source",
                "Actions": [
                    {
                        "Name": "SourceAction",
                        "ActionTypeId": {
                            "Category": "Source",
                            "Owner": "AWS",
                            "Version": 1,
                            "Provider": "S3"
                        },
                        "OutputArtifacts": [
                            {
                                "Name": "SourceOutput"
                            }
                        ],
                        "Configuration": {
                            "S3Bucket": {
                                "Ref": "SourceBucket"
                            },
                            "S3ObjectKey": {
                                "Ref": "SourceObjectKey"
                            },
                            "PollForSourceChanges": false
                        },
                        "RunOrder": 1
                    }
                ]
            },
            {
                "Name": "Beta",
                "Actions": [
                    {
                        "Name": "BetaAction",
                        "InputArtifacts": [
                            {
                                "Name": "SourceOutput"
                            }
                        ],
                        "ActionTypeId": {
                            "Category": "Deploy",
```

```
                                "Owner": "AWS",
                                "Version": 1,
                                "Provider": "CodeDeploy"
                            },
                            "Configuration": {
                                "ApplicationName": {
                                    "Ref": "ApplicationName"
                                },
                                "DeploymentGroupName": {
                                    "Ref": "BetaFleet"
                                }
                            },
                            "RunOrder": 1
                        }
                    ]
                }
            ],
            "ArtifactStore": {
                "Type": "S3",
                "Location": {
                    "Ref": "CodePipelineArtifactStoreBucket"
                }
            }
        }
    },
    "EventRole": {
        "Type": "AWS::IAM::Role",
        "Properties": {
            "AssumeRolePolicyDocument": {
                "Version": "2012-10-17",
                "Statement": [
                    {
                        "Effect": "Allow",
                        "Principal": {
                            "Service": [
                                "events.amazonaws.com"
                            ]
                        },
                        "Action": "sts:AssumeRole"
                    }
                ]
            },
            "Path": "/",
            "Policies": [
```

```
                {
                    "PolicyName": "eb-pipeline-execution",
                    "PolicyDocument": {
                        "Version": "2012-10-17",
                        "Statement": [
                            {
                                "Effect": "Allow",
                                "Action": "codepipeline:StartPipelineExecution",
                                "Resource": {
                                    "Fn::Join": [
                                        "",
                                        [
                                            "arn:aws:codepipeline:",
                                            {
                                                "Ref": "AWS::Region"
                                            },
                                            ":",
                                            {
                                                "Ref": "AWS::AccountId"
                                            },
                                            ":",
                                            {
                                                "Ref": "AppPipeline"
                                            }
                                        ]
                                    ]
                                }
                            }
                        ]
                    }
                }
            ]
        }
    },
    "EventRule": {
        "Type": "AWS::Events::Rule",
        "Properties": {
            "EventPattern": {
                "source": [
                    "aws.s3"
                ],
                "detail-type": [
                    "AWS API Call via CloudTrail"
                ],
```

```
                    "detail": {
                        "eventSource": [
                            "s3.amazonaws.com"
                        ],
                        "eventName": [
                            "PutObject",
                            "CompleteMultipartUpload"
                        ],
                        "resources": {
                            "ARN": [
                                {
                                    "Fn::Join": [
                                        "",
                                        [
                                            {
                                                "Fn::GetAtt": [
                                                    "SourceBucket",
                                                    "Arn"
                                                ]
                                            },
                                            "/",
                                            {
                                                "Ref": "SourceObjectKey"
                                            }
                                        ]
                                    ]
                                }
                            ]
                        }
                    }
                },
                "Targets": [
                    {
                        "Arn": {
                            "Fn::Join": [
                                "",
                                [
                                    "arn:aws:codepipeline:",
                                    {
                                        "Ref": "AWS::Region"
                                    },
                                    ":",
                                    {
                                        "Ref": "AWS::AccountId"
```

```
                                    },
                                    ":",
                                    {
                                        "Ref": "AppPipeline"
                                    }
                                ]
                            ]
                        },
                        "RoleArn": {
                            "Fn::GetAtt": [
                                "EventRole",
                                "Arn"
                            ]
                        },
                        "Id": "codepipeline-AppPipeline"
                    }
                ]
            }
        }
    },
    "Outputs" : {
        "SourceBucketARN" : {
            "Description" : "S3 bucket ARN that Cloudtrail will use",
            "Value" : { "Fn::GetAtt": ["SourceBucket", "Arn"] },
            "Export" : {
                "Name" : "SourceBucketARN"
            }
        }
    }
}


...
```

Migra le pipeline di polling per un' GitHub azione sorgente della versione 1 verso le connessioni

Puoi migrare un'azione di origine della GitHub versione 1 per utilizzare le connessioni per il tuo repository esterno. Questo è il metodo di rilevamento delle modifiche consigliato per le pipeline con un'azione di origine della GitHub versione 1.

Per una pipeline con un'azione di origine della GitHub versione 1, consigliamo di modificare la pipeline per utilizzare un'azione della GitHub versione 2 in modo da automatizzare il rilevamento delle modifiche. AWS CodeConnections Per ulteriori informazioni sull'utilizzo delle connessioni, vedere. GitHub connessioni

Creare una connessione a GitHub (console)

È possibile utilizzare la console per creare una connessione a GitHub.

Passaggio 1: Sostituisci l' GitHub azione della versione 1

Utilizza la pagina di modifica della pipeline per sostituire l'azione della versione 1 con GitHub un'azione della versione 2 GitHub .

Per sostituire l'azione della versione 1 GitHub

1. Accedi alla CodePipeline console.
2. Scegli la tua pipeline e scegli Modifica. Scegli Modifica fase nella fase di origine. Viene visualizzato un messaggio che consiglia di aggiornare l'azione.
3. Nel provider Action, scegli GitHub (Versione 2).
4. Esegui una di queste operazioni:

   - In Connessione, se non hai già creato una connessione con il tuo provider, scegli Connetti a GitHub. Procedi al Passaggio 2: Crea una connessione a GitHub.
   - In Connessione, se hai già creato una connessione al tuo provider, scegli la connessione. Procedi al passaggio 3: Salva l'azione di origine per la tua connessione.

Passaggio 2: Creare una connessione a GitHub

Dopo aver scelto di creare la connessione, viene GitHub visualizzata la pagina Connetti a.

Per creare una connessione a GitHub

1. Nelle impostazioni di GitHub connessione, il nome della connessione viene visualizzato in Nome connessione.

   In GitHub App, scegli l'installazione di un'app o scegli Installa una nuova app per crearne una.

> **Note**
>
> È sufficiente installare una sola app per tutte le connessioni a un provider specifico. Se hai già installato l' GitHub app, sceglila e salta questo passaggio.

2. Se GitHub viene visualizzata la pagina di autorizzazione, accedi con le tue credenziali e scegli di continuare.
3. Nella pagina di installazione dell'app, un messaggio indica che l' AWS CodeStar app sta tentando di connettersi al tuo GitHub account.

> **Note**
>
> L'app viene installata una sola volta per ogni GitHub account. Se hai già installato l'app, puoi scegliere Configure (Configura) per passare a una pagina di modifica per l'installazione dell'app oppure è possibile utilizzare il pulsante Indietro per tornare alla console.

4. Nella AWS CodeStar pagina di installazione, scegli Installa.
5. Nella GitHub pagina Connect to, viene visualizzato l'ID di connessione per la nuova installazione. Scegli Connetti.

Passaggio 3: Salva l'azione GitHub sorgente

Completa gli aggiornamenti nella pagina Modifica azione per salvare la nuova azione sorgente.

Per salvare l'azione GitHub sorgente

1. In Repository, inserisci il nome del tuo repository di terze parti. In Branch, inserisci il ramo in cui desideri che la pipeline rilevi le modifiche all'origine.

> **Note**
>
> In Repository, digita `owner-name/repository-name` come mostrato in questo esempio:
>
> ```
> my-account/my-repository
> ```

2. In Formato di output degli artefatti, scegliete il formato per gli artefatti.

   - Per memorizzare gli artefatti di output derivanti dall' GitHub azione utilizzando il metodo predefinito, scegliete predefinito. CodePipeline L'azione accede ai file dal GitHub repository e archivia gli artefatti in un file ZIP nel Pipeline Artifact Store.
   - Per archiviare un file JSON contenente un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git, scegliere Full clone (Clone completo). Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild

     Se scegli questa opzione, dovrai aggiornare le autorizzazioni per il tuo ruolo di CodeBuild Project Service come mostrato in. Aggiungi le autorizzazioni per le connessioni a Bitbucket, Enterprise Server o.com CodeBuild GitClone GitHub GitHub GitLab Per un tutorial che mostra come usare l'opzione Full clone, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline
3. In Output Artifacts, puoi mantenere il nome dell'artefatto di output per questa azione, ad esempio. `SourceArtifact` Scegli Fine per chiudere la pagina Modifica azione.
4. Scegliete Fine per chiudere la pagina di modifica dello stage. Scegliete Salva per chiudere la pagina di modifica della pipeline.

Creare una connessione a GitHub (CLI)

È possibile utilizzare AWS Command Line Interface (AWS CLI) per creare una connessione a GitHub.

Per farlo, utilizzare il comando create-connection.

> **Important**
>
> Per impostazione predefinita, una connessione creata tramite AWS CLI o AWS CloudFormation è in `PENDING` stato. Dopo aver creato una connessione con la CLI o AWS CloudFormation, utilizza la console per modificare la connessione e definirne lo stato. `AVAILABLE`

Per creare una connessione a GitHub

1. Apri un terminale (Linux, macOS o Unix) o prompt dei comandi (Windows). Utilizzate il AWS CLI per eseguire il create-connection comando, specificando `--provider-type` e `--`

`connection-name` per la connessione. In questo esempio, il nome del provider di terze parti è `GitHub` e il nome della connessione specificato è `MyConnection`.

```
aws codeconnections create-connection --provider-type GitHub --connection-name
 MyConnection
```

In caso di esito positivo, questo comando restituisce informazioni dell'ARN della connessione simili alle seguenti.

```
{
    "ConnectionArn": "arn:aws:codeconnections:us-west-2:account_id:connection/
aEXAMPLE-8aad-4d5d-8878-dfcab0bc441f"
}
```

2. Utilizzare la console per completare la connessione.

Migra le pipeline di polling per un'azione sorgente della GitHub versione 1 ai webhook

Puoi migrare la tua pipeline per utilizzare i webhook per rilevare le modifiche nel tuo repository di origine. GitHub Questa migrazione ai webhook riguarda solo l'azione della versione 1. GitHub

- Console: Migrazione delle pipeline di polling ai webhook (azioni di origine della GitHub versione 1) (console)
- CLI: Migrazione delle pipeline di polling ai webhook (azioni sorgente GitHub versione 1) (CLI)
- AWS CloudFormation: Pipeline di aggiornamento per gli eventi push (azioni di origine della GitHub versione 1) (modello)AWS CloudFormation

Migrazione delle pipeline di polling ai webhook (azioni di origine della GitHub versione 1) (console)

È possibile utilizzare la CodePipeline console per aggiornare la pipeline e utilizzare i webhook per rilevare le modifiche nell'archivio di origine. CodeCommit

Segui questi passaggi per modificare una pipeline che utilizza invece il polling (controlli periodici). EventBridge Se vuoi creare una pipeline, consulta Creare una pipeline in CodePipeline.

Quando utilizzi la console, il parametro `PollForSourceChanges` per la pipeline viene modificato per te. Il GitHub webhook viene creato e registrato automaticamente.

Per modificare la fase di origine della pipeline

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegliere il nome della pipeline da modificare. Questa operazione apre una visualizzazione dettagliata della pipeline. compreso lo stato di ciascuna delle operazioni in ciascuna fase della pipeline.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. In Edit stage (Modifica fase), scegli l'icona di modifica sull'operazione di origine.
5. Espandi le opzioni di rilevamento delle modifiche e scegli Usa Amazon CloudWatch Events per avviare automaticamente la mia pipeline quando si verifica una modifica (consigliato).

   Viene visualizzato un messaggio che avvisa che CodePipeline crea un webhook GitHub per rilevare le modifiche all'origine: AWS CodePipeline creerà un webhook per te. È possibile effettuare l'opt-out nelle opzioni riportate di seguito. Scegli Aggiorna. Oltre al webhook, CodePipeline crea quanto segue:

   - Un segreto, generato casualmente e utilizzato per autorizzare la connessione a. GitHub
   - L'URL del webhook, generato utilizzando l'endpoint pubblico della regione.

   CodePipeline registra il webhook con. GitHub Questo consente di effettuare la sottoscrizione dell'URL per ricevere eventi del repository.
6. Al termine della modifica della pipeline, scegliere Save pipeline changes (Salva modifiche pipeline) per tornare alla pagina di riepilogo.

   Viene visualizzato un messaggio con il nome del webhook da creare per la pipeline. Seleziona Salva e continua.
7. Per testare la tua azione, rilascia una modifica utilizzando il AWS CLI comando per confermare una modifica alla fonte specificata nella fase di origine della pipeline.

Migrazione delle pipeline di polling ai webhook (azioni sorgente GitHub versione 1) (CLI)

Segui queste fasi per modificare una pipeline che utilizza polling (controlli periodici) in modo da usare invece un webhook. Se vuoi creare una pipeline, consulta Creare una pipeline in CodePipeline.

Per creare una pipeline basata su eventi, dovrai modificare il parametro `PollForSourceChanges` della pipeline e quindi creare le seguenti risorse manualmente:

- GitHub webhook e parametri di autorizzazione

Per creare e registrare il webhook

> **Note**
>
> Quando si utilizza la CLI o AWS CloudFormation si crea una pipeline e si aggiunge un webhook, è necessario disabilitare i controlli periodici. Per disabilitare i controlli periodici, devi aggiungere esplicitamente il parametro `PollForSourceChanges` e impostarlo su "false", come descritto nella procedura finale di seguito. Altrimenti, l'impostazione predefinita per una CLI o una AWS CloudFormation pipeline è che il `PollForSourceChanges` valore predefinito è true e non viene visualizzato nell'output della struttura della pipeline. Per ulteriori informazioni sui valori predefiniti, vedere. PollForSourceChanges Impostazioni predefinite per il parametro PollForSourceChanges

1. In un editor di testo, creare e salvare un file JSON per il webhook da creare. Utilizza questo file di esempio per un webhook denominato `my-webhook`:

```
{
    "webhook": {
        "name": "my-webhook",
  "targetPipeline": "pipeline_name",
  "targetAction": "source_action_name",
  "filters": [{
      "jsonPath": "$.ref",
      "matchEquals": "refs/heads/{Branch}"
  }],
  "authentication": "GITHUB_HMAC",
  "authenticationConfiguration": {
```

```
      "SecretToken": "secret"
  }
    }
}
```

2. Chiama il comando put-webhook e includi i parametri `--cli-input` e `--region`.

   Il comando di esempio seguente crea un webhook con il file JSON `webhook_json`.

```
aws codepipeline put-webhook --cli-input-json file://webhook_json.json --region
 "eu-central-1"
```

3. Nell'output mostrato in questo esempio, vengono restituiti l'URL e l'ARN per un webhook denominato `my-webhook`.

```
{
    "webhook": {
        "url": "https://webhooks.domain.com/
trigger111111111EXAMPLE1111111111111111",
        "definition": {
            "authenticationConfiguration": {
                "SecretToken": "secret"
            },
            "name": "my-webhook",
            "authentication": "GITHUB_HMAC",
            "targetPipeline": "pipeline_name",
            "targetAction": "Source",
            "filters": [
                {
                    "jsonPath": "$.ref",
                    "matchEquals": "refs/heads/{Branch}"
                }
            ]
        },
        "arn": "arn:aws:codepipeline:eu-central-1:ACCOUNT_ID:webhook:my-webhook"
    },
    "tags": [{
      "key": "Project",
      "value": "ProjectA"
    }]
}
```

In questo esempio vengono aggiunte tag al webhook, inclusa la chiave di tag `Project` e il valore `ProjectA` al webhook. Per ulteriori informazioni sull'etichettatura delle risorse, vedere. CodePipeline Assegnazione di tag alle risorse

4. Chiama il comando register-webhook-with-third-party e includi il parametro `--webhook-name`.

   Il comando di esempio seguente registra un webhook denominato `my-webhook`.

   ```
   aws codepipeline register-webhook-with-third-party --webhook-name my-webhook
   ```

Per modificare il parametro della pipeline PollForSourceChanges

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

1. Esegui il comando get-pipeline per copiare la struttura della pipeline in un file JSON. Ad esempio, per una pipeline denominata `MyFirstPipeline`, digitare il comando seguente:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   ```

   Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.

2. Apri il file JSON in qualsiasi editor di testo normale e modifica la fase di origine modificando o aggiungendo il parametro `PollForSourceChanges`. In questo esempio, per un repository denominato `UserGitHubRepo`, il parametro è impostato su `false`.

   Perché sto apportando questa modifica? La modifica di questo parametro disattiva i controlli periodici, quindi è possibile utilizzare solo il rilevamento delle modifiche basato sugli eventi.

   ```
   "configuration": {
   ```

```
    "Owner": "name",
    "Repo": "UserGitHubRepo",
    "PollForSourceChanges": "false",
    "Branch": "main",
    "OAuthToken": "****"
},
```

3. Se utilizzi la struttura della pipeline recuperata utilizzando il comando get-pipeline, devi modificare la struttura del file JSON rimuovendo le righe `metadata` dal file. In caso contrario, il comando update-pipeline non è in grado di utilizzarlo. Rimuovi la sezione `"metadata"` dalla struttura della pipeline nel file JSON, inclusi `{ }` e i campi `"created"`, `"pipelineARN"` e `"updated"`.

   Ad esempio, rimuovere dalla struttura le seguenti righe:

```
"metadata": {
    "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
    "created": "date",
    "updated": "date"
},
```

   Salvare il file.

4. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline, in modo analogo al seguente:

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

```
aws codepipeline update-pipeline --cli-input-json file://pipeline.json
```

   Questo comando restituisce l'intera struttura della pipeline modificata.

> **Note**
>
> Il comando update-pipeline arresta la pipeline. Se è in corso di elaborazione una versione durante l'esecuzione del comando update-pipeline, tale elaborazione viene

arrestata. Per elaborare tale versione utilizzando la pipeline aggiornata, devi avviare manualmente la pipeline. Utilizza il comando start-pipeline-execution per avviare manualmente la pipeline.

Pipeline di aggiornamento per gli eventi push (azioni di origine della GitHub versione 1) (modello)AWS CloudFormation

Segui questi passaggi per aggiornare la pipeline (con una GitHub fonte) dai controlli periodici (polling) al rilevamento delle modifiche basato sugli eventi tramite webhook.

Per creare una pipeline basata sugli eventi con AWS CodeCommit, devi modificare il `PollForSourceChanges` parametro della pipeline e quindi aggiungere una risorsa webhook al tuo modello. GitHub

Se lo utilizzi AWS CloudFormation per creare e gestire le tue pipeline, il tuo modello ha contenuti come i seguenti.

> **Note**
>
> Nota la proprietà di configurazione `PollForSourceChanges` nella fase di origine. Se il modello non include questa proprietà, allora `PollForSourceChanges` è impostato su `true` per impostazione predefinita.

YAML

```
Resources:
  AppPipeline:
    Type: AWS::CodePipeline::Pipeline
    Properties:
      Name: github-polling-pipeline
      RoleArn:
        !GetAtt CodePipelineServiceRole.Arn
      Stages:
        -
          Name: Source
          Actions:
            -
              Name: SourceAction
```

```
              ActionTypeId:
                Category: Source
                Owner: ThirdParty
                Version: 1
                Provider: GitHub
              OutputArtifacts:
                - Name: SourceOutput
              Configuration:
                Owner: !Ref GitHubOwner
                Repo: !Ref RepositoryName
                Branch: !Ref BranchName
                OAuthToken:
 {{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}
 PollForSourceChanges: true
              RunOrder: 1


...
```

JSON

```
        "AppPipeline": {
            "Type": "AWS::CodePipeline::Pipeline",
            "Properties": {
                "Name": "github-polling-pipeline",
                "RoleArn": {
                    "Fn::GetAtt": [
                        "CodePipelineServiceRole",
                        "Arn"
                    ]
                },
                "Stages": [
                    {
                        "Name": "Source",
                        "Actions": [
                            {
                                "Name": "SourceAction",
                                "ActionTypeId": {
                                    "Category": "Source",
                                    "Owner": "ThirdParty",
                                    "Version": 1,
                                    "Provider": "GitHub"
                                },
```

```
                                "OutputArtifacts": [
                                    {
                                        "Name": "SourceOutput"
                                    }
                                ],
                                "Configuration": {
                                    "Owner": {
                                        "Ref": "GitHubOwner"
                                    },
                                    "Repo": {
                                        "Ref": "RepositoryName"
                                    },
                                    "Branch": {
                                        "Ref": "BranchName"
                                    },
                                    "OAuthToken":
 "{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}",
                                    "PollForSourceChanges": true
                                },
                                "RunOrder": 1
                            }
                        ]
                    },


...
```

Per aggiungere parametri e creare un webhook nel modello

Ti consigliamo vivamente di AWS Secrets Manager utilizzarlo per archiviare le tue credenziali. Se si utilizza Secrets Manager, è necessario avere già configurato e archiviato i parametri segreti in Secrets Manager. Questo esempio utilizza riferimenti dinamici a Secrets Manager per GitHub le credenziali del webhook. Per ulteriori informazioni, consulta Utilizzo di riferimenti dinamici per specificare valori di modello.

> **Important**
>
> Quando si passano parametri segreti, non immettere il valore direttamente nel modello. Il valore viene reso come testo in chiaro ed è quindi leggibile. Per motivi di sicurezza, non utilizzate testo semplice nel AWS CloudFormation modello per memorizzare le credenziali.

Quando si utilizza la CLI o AWS CloudFormation si crea una pipeline e si aggiunge un webhook, è necessario disabilitare i controlli periodici.

> **Note**
>
> Per disabilitare i controlli periodici, devi aggiungere esplicitamente il parametro `PollForSourceChanges` e impostarlo su "false", come descritto nella procedura finale di seguito. Altrimenti, l'impostazione predefinita per una CLI o una AWS CloudFormation pipeline è che il `PollForSourceChanges` valore predefinito è true e non viene visualizzato nell'output della struttura della pipeline. Per ulteriori informazioni sui valori predefiniti, vedere. PollForSourceChanges Impostazioni predefinite per il parametro PollForSourceChanges

1. Nel modello, in `Resources`, aggiungi i parametri:

   YAML

   ```
   Parameters:
           GitHubOwner:
             Type: String

   ...
   ```

   JSON

   ```
   {
    "Parameters": {
     "BranchName": {
      "Description": "GitHub branch name",
      "Type": "String",
      "Default": "main"
     },
     "GitHubOwner": {
      "Type": "String"
     },

   ...
   ```

2. Usa la `AWS::CodePipeline::Webhook` AWS CloudFormation risorsa per aggiungere un webhook.

> **Note**
>
> Il `TargetAction` specificato deve corrispondere alla proprietà `Name` dell'operazione di origine definita nella pipeline.

Se `RegisterWithThirdParty` è impostato su`true`, assicurati che l'utente associato a `0AuthToken` possa impostare gli ambiti richiesti. GitHub Il token e il webhook richiedono i seguenti ambiti: GitHub

- `repo` - utilizzato per il controllo completo per leggere ed estrarre artefatti da repository pubblici e privati in una pipeline.
- `admin:repo_hook` - utilizzato per il controllo completo di hook di repository.

Altrimenti, GitHub restituisce un 404. Per ulteriori informazioni sulla restituzione di un 404, consulta https://help.github.com/articles/about-webhooks.

YAML

```
 AppPipelineWebhook:
   Type: AWS::CodePipeline::Webhook
   Properties:
     Authentication: GITHUB_HMAC
     AuthenticationConfiguration:
       SecretToken:
{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}
     Filters:
       -
         JsonPath: "$.ref"
         MatchEquals: refs/heads/{Branch}
     TargetPipeline: !Ref AppPipeline
     TargetAction: SourceAction
     Name: AppPipelineWebhook
     TargetPipelineVersion: !GetAtt AppPipeline.Version
     RegisterWithThirdParty: true


...
```

JSON

```
"AppPipelineWebhook": {
    "Type": "AWS::CodePipeline::Webhook",
    "Properties": {
        "Authentication": "GITHUB_HMAC",
        "AuthenticationConfiguration": {
            "SecretToken":
 "{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}"
        },
        "Filters": [{
            "JsonPath": "$.ref",
            "MatchEquals": "refs/heads/{Branch}"
        }],
        "TargetPipeline": {
            "Ref": "AppPipeline"
        },
        "TargetAction": "SourceAction",
        "Name": "AppPipelineWebhook",
        "TargetPipelineVersion": {
            "Fn::GetAtt": [
              "AppPipeline",
              "Version"
            ]
        },
        "RegisterWithThirdParty": true
    }
},

...
```

3. Salva il modello aggiornato sul computer locale, quindi apri la AWS CloudFormation console.
4. Seleziona lo stack e scegli Create Change Set for Current Stack (Crea set di modifiche per lo stack corrente).
5. Carica il modello e quindi visualizza le modifiche elencate in AWS CloudFormation. Queste sono le modifiche da apportare allo stack. Le nuove risorse dovrebbero essere visibili nell'elenco.
6. Scegliere Execute (Esegui).

Per modificare i parametri della PollForSourceChanges pipeline

> **Important**
>
> Quando crei una pipeline con questo metodo, il parametro `PollForSourceChanges` è preimpostato su "true" se non viene impostato esplicitamente su "false". Quando aggiungi il rilevamento delle modifiche basato su eventi, devi aggiungere il parametro all'output e impostarlo su "false" per disabilitare il polling. In caso contrario, la pipeline si avvia due volte per una singola modifica dell'origine. Per informazioni dettagliate, vedi Impostazioni predefinite per il parametro PollForSourceChanges .

- Nel modello, modifica `PollForSourceChanges` in `false`. Se non hai incluso `PollForSourceChanges` nella definizione della pipeline, aggiungilo e impostalo su "false".

  Perché sto apportando questa modifica? La modifica di questo parametro in `false` disattiva i controlli periodici, in modo che sia possibile utilizzare solo il rilevamento delle modifiche basato su eventi.

  YAML

```
          Name: Source
          Actions:
            -
              Name: SourceAction
              ActionTypeId:
                Category: Source
                Owner: ThirdParty
                Version: 1
                Provider: GitHub
              OutputArtifacts:
                - Name: SourceOutput
              Configuration:
                Owner: !Ref GitHubOwner
                Repo: !Ref RepositoryName
                Branch: !Ref BranchName
                OAuthToken:
{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}
                PollForSourceChanges: false
              RunOrder: 1
```

JSON

```
{
   "Name": "Source",
   "Actions": [{
 "Name": "SourceAction",
 "ActionTypeId": {
     "Category": "Source",
     "Owner": "ThirdParty",
     "Version": 1,
     "Provider": "GitHub"
},
"OutputArtifacts": [{
    "Name": "SourceOutput"
}],
"Configuration": {
    "Owner": {
  "Ref": "GitHubOwner"
    },
    "Repo": {
  "Ref": "RepositoryName"
    },
    "Branch": {
        "Ref": "BranchName"
    },
    "OAuthToken":
"{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}",
    PollForSourceChanges: false
},
"RunOrder": 1
   }]
```

Example

Quando si creano queste risorse con AWS CloudFormation, il webhook definito viene creato nel GitHub repository specificato. La pipeline si attiva alla conferma.

YAML

```
Parameters:
  GitHubOwner:
```

```
    Type: String

Resources:
  AppPipelineWebhook:
    Type: AWS::CodePipeline::Webhook
    Properties:
      Authentication: GITHUB_HMAC
      AuthenticationConfiguration:
        SecretToken: {{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}
      Filters:
        -
          JsonPath: "$.ref"
          MatchEquals: refs/heads/{Branch}
      TargetPipeline: !Ref AppPipeline
      TargetAction: SourceAction
      Name: AppPipelineWebhook
      TargetPipelineVersion: !GetAtt AppPipeline.Version
      RegisterWithThirdParty: true
  AppPipeline:
    Type: AWS::CodePipeline::Pipeline
    Properties:
      Name: github-events-pipeline
      RoleArn:
        !GetAtt CodePipelineServiceRole.Arn
      Stages:
        -
          Name: Source
          Actions:
            -
              Name: SourceAction
              ActionTypeId:
                Category: Source
                Owner: ThirdParty
                Version: 1
                Provider: GitHub
              OutputArtifacts:
                - Name: SourceOutput
              Configuration:
                Owner: !Ref GitHubOwner
                Repo: !Ref RepositoryName
                Branch: !Ref BranchName
                OAuthToken:
 {{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}
                PollForSourceChanges: false
```

```
              RunOrder: 1

...
```

JSON

```
{
    "Parameters": {
        "BranchName": {
            "Description": "GitHub branch name",
            "Type": "String",
            "Default": "main"
        },
        "RepositoryName": {
            "Description": "GitHub repository name",
            "Type": "String",
            "Default": "test"
        },
        "GitHubOwner": {
            "Type": "String"
        },
        "ApplicationName": {
            "Description": "CodeDeploy application name",
            "Type": "String",
            "Default": "DemoApplication"
        },
        "BetaFleet": {
            "Description": "Fleet configured in CodeDeploy",
            "Type": "String",
            "Default": "DemoFleet"
        }
    },
    "Resources": {

...

        },
        "AppPipelineWebhook": {
            "Type": "AWS::CodePipeline::Webhook",
            "Properties": {
                "Authentication": "GITHUB_HMAC",
                "AuthenticationConfiguration": {
                    "SecretToken": {
```

```
"{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}"
                    }
                },
                "Filters": [
                    {
                        "JsonPath": "$.ref",
                        "MatchEquals": "refs/heads/{Branch}"
                    }
                ],
                "TargetPipeline": {
                    "Ref": "AppPipeline"
                },
                "TargetAction": "SourceAction",
                "Name": "AppPipelineWebhook",
                "TargetPipelineVersion": {
                    "Fn::GetAtt": [
                        "AppPipeline",
                        "Version"
                    ]
                },
                "RegisterWithThirdParty": true
            }
        },
        "AppPipeline": {
            "Type": "AWS::CodePipeline::Pipeline",
            "Properties": {
                "Name": "github-events-pipeline",
                "RoleArn": {
                    "Fn::GetAtt": [
                        "CodePipelineServiceRole",
                        "Arn"
                    ]
                },
                "Stages": [
                    {
                        "Name": "Source",
                        "Actions": [
                            {
                                "Name": "SourceAction",
                                "ActionTypeId": {
                                    "Category": "Source",
                                    "Owner": "ThirdParty",
                                    "Version": 1,
```

```
                                    "Provider": "GitHub"
                                },
                                "OutputArtifacts": [
                                    {
                                        "Name": "SourceOutput"
                                    }
                                ],
                                "Configuration": {
                                    "Owner": {
                                        "Ref": "GitHubOwner"
                                    },
                                    "Repo": {
                                        "Ref": "RepositoryName"
                                    },
                                    "Branch": {
                                        "Ref": "BranchName"
                                    },
                                    "OAuthToken":
 "{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}",
                                    "PollForSourceChanges": false
                                },
                                "RunOrder": 1

...
```

Creare il ruolo CodePipeline di servizio

Quando crei una pipeline, devi creare un ruolo del servizio o utilizzare un ruolo del servizio esistente.

È possibile utilizzare la CodePipeline console o AWS CLI creare un ruolo CodePipeline di servizio. Un ruolo del servizio è obbligatorio per creare una pipeline e la pipeline è sempre associata a quel ruolo del servizio.

Prima di creare una pipeline con la AWS CLI, è necessario creare CodePipeline un ruolo di servizio per la pipeline. Per un AWS CloudFormation modello di esempio con il ruolo di servizio e la policy specificati, consulta i tutorial in. Tutorial: crea una pipeline che utilizza le variabili delle azioni di AWS CloudFormation distribuzione

Il ruolo di servizio non è un ruolo AWS gestito, ma viene creato inizialmente per la creazione della pipeline, quindi, man mano che vengono aggiunte nuove autorizzazioni alla politica del ruolo di servizio, potrebbe essere necessario aggiornare il ruolo di servizio per la pipeline. Una volta che la

pipeline viene creata con un ruolo del servizio, non è possibile applicare un altro ruolo del servizio a quella pipeline. Collega la policy consigliata al ruolo del servizio.

Per ulteriori informazioni sul ruolo del servizio, consulta Gestisci il ruolo CodePipeline del servizio.

Crea il ruolo di CodePipeline servizio (console)

Quando si utilizza la console per creare una pipeline, si crea il ruolo di CodePipeline servizio con la procedura guidata per la creazione della pipeline.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Scegli Create pipeline (Crea pipeline) e completa la pagina Step 1: Choose pipeline settings (Fase 1: scegliere le impostazioni della pipeline) nella procedura guidata di creazione della pipeline.

   > **Note**
   >
   > Non è possibile modificare il nome di una pipeline dopo che è stata creata. Per informazioni su altre limitazioni, consulta Quote in AWS CodePipeline.

2. In Ruolo di servizio, scegli Nuovo ruolo di servizio CodePipeline per consentire la creazione di un nuovo ruolo di servizio in IAM.
3. Completa la creazione della pipeline. Il ruolo del servizio pipeline è disponibile per la visualizzazione nell'elenco dei ruoli IAM e puoi visualizzare l'ARN del ruolo di servizio associato a una pipeline eseguendo il comando con `get-pipeline` la CLI. AWS

Creare il ruolo CodePipeline di servizio (CLI)

Prima di creare una pipeline con la AWS CLI AWS CloudFormation oppure, è necessario creare CodePipeline un ruolo di servizio per la pipeline e allegare la policy del ruolo di servizio e la policy di trust. Per utilizzare la CLI per creare il tuo ruolo di servizio, utilizza i passaggi seguenti per creare prima una policy di fiducia JSON e la policy di ruolo JSON come file separati nella directory in cui eseguirai i comandi CLI.

> **Note**
>
> Ti consigliamo di consentire solo agli utenti amministrativi di creare qualsiasi ruolo di servizio. Una persona con autorizzazioni per creare un ruolo e allegare qualsiasi policy può eseguire l'escalation delle proprie autorizzazioni. Invece, crea una policy che consenta a questa persona di creare solo i ruoli di cui hanno bisogno o lascia che un amministratore crei il ruolo di servizio per suo conto.

1. In una finestra di terminale, inserisci il seguente comando per creare un file denominato`TrustPolicy.json`, in cui incollerai la politica del ruolo JSON. Questo esempio utilizza VIM.

   ```
   vim TrustPolicy.json
   ```

2. Incolla il seguente codice JSON nel file.

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Principal": {
                   "Service": "codepipeline.amazonaws.com"
               },
               "Action": "sts:AssumeRole"
           }
       ]
   }
   ```

   Per salvare e uscire dal file, inserisci il seguente comando VIM:

   ```
   :wq
   ```

3. In una finestra di terminale, inserisci il seguente comando per creare un file denominato`RolePolicy.json`, in cui incollerai la policy del ruolo JSON. Questo esempio utilizza VIM.

   ```
   vim RolePolicy.json
   ```

4. Incolla il seguente codice JSON nel file. Assicurati di limitare il più possibile le autorizzazioni aggiungendo l'Amazon Resource Name (ARN) per la tua pipeline nel campo della policy statement. `Resource`

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:CancelUploadArchive",
        "codecommit:GetBranch",
        "codecommit:GetCommit",
        "codecommit:GetUploadArchiveStatus",
        "codecommit:UploadArchive"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "codedeploy:CreateDeployment",
        "codedeploy:GetApplicationRevision",
        "codedeploy:GetDeployment",
        "codedeploy:GetDeploymentConfig",
        "codedeploy:RegisterApplicationRevision"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "codebuild:BatchGetBuilds",
        "codebuild:StartBuild"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "devicefarm:ListProjects",
        "devicefarm:ListDevicePools",
        "devicefarm:GetRun",
```

```
        "devicefarm:GetUpload",
        "devicefarm:CreateUpload",
        "devicefarm:ScheduleRun"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "lambda:InvokeFunction",
        "lambda:ListFunctions"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "elasticbeanstalk:*",
        "ec2:*",
        "elasticloadbalancing:*",
        "autoscaling:*",
        "cloudwatch:*",
        "s3:*",
        "sns:*",
        "cloudformation:*",
        "rds:*",
        "sqs:*",
        "ecs:*"
      ],
      "Resource": "resource_ARN"
    }
  ]
}
```

Per salvare e uscire dal file, inserisci il seguente comando VIM:

```
:wq
```

5. Immettete il seguente comando per creare il ruolo e allegare la policy di trust role. Il formato del nome della policy è solitamente lo stesso del nome del ruolo. Questo esempio utilizza il nome del ruolo `MyRole` e `TrustPolicy` la policy creati come file separato.

```
aws iam create-role --role-name MyRole --assume-role-policy-document file://
TrustPolicy.json
```

6. Immettete il comando seguente per creare la politica del ruolo e allegarla al ruolo. Il formato del nome della policy è solitamente lo stesso del nome del ruolo. Questo esempio utilizza il nome del ruolo `MyRole` e `MyRole` la policy creati come file separato.

```
aws iam put-role-policy --role-name MyRole --policy-name RolePolicy --policy-
document file://RolePolicy.json
```

7. Per visualizzare il nome del ruolo e la politica di attendibilità creati, immettete il seguente comando per il ruolo denominato`MyRole`:

```
aws iam get-role --role-name MyRole
```

8. Usa il ruolo di servizio ARN quando crei la tua pipeline con la CLI o AWS . AWS CloudFormation

Contrassegna una pipeline in CodePipeline

I tag sono coppie chiave-valore associate alle risorse. AWS Puoi applicare tag alle tue pipeline in. CodePipeline Per informazioni sull'etichettatura CodePipeline delle risorse, sui casi d'uso, sui vincoli di chiave e valore dei tag e sui tipi di risorse supportati, consulta. Assegnazione di tag alle risorse

È possibile utilizzare l'interfaccia a riga di comando per specificare i tag al momento della creazione di una pipeline. È possibile utilizzare la console o l'interfaccia a riga di comando per aggiungere o rimuovere i tag e aggiornare i valori di tag in una pipeline. Puoi aggiungere fino a 50 tag per ciascuna pipeline.

Argomenti

- Tagging di pipeline (console)
- Tagging di pipeline (CLI)

Tagging di pipeline (console)

È possibile utilizzare la console o l'interfaccia a riga di comando per aggiungere tag alle risorse. Le pipeline sono l'unica CodePipeline risorsa che può essere gestita con la console o la CLI.

Argomenti

Aggiunta di tag a una pipeline (console)

È possibile utilizzare la console per aggiungere i tag a una pipeline esistente.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Pipelines (Pipeline), scegliere la pipeline in cui aggiungere i tag.
3. Nel riquadro di navigazione scegliere Impostazioni.
4. In Pipeline tags (Tag della pipeline), scegliere Modifica.
5. Nei campi Key (Chiave) e Value (Valore), immettere una coppia di chiavi per ogni set di tag che si desidera aggiungere. Il campo Value (Valore) è facoltativo. Ad esempio, in Key (Chiave), immettere **`Project`**. In Valore, immetti **`ProjectA`**.
6. (Facoltativo) Scegliere Add tag (Aggiungi tag) per aggiungere ulteriori righe e inserire più tag.
7. Scegli Invia. I tag sono elencati nelle impostazioni della pipeline.

Visualizzazione di tag per una pipeline (console)

È possibile utilizzare la console per elencare i tag per le pipeline esistenti.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Pipelines scegliere la pipeline in cui si desidera visualizzare i tag.
3. Nel riquadro di navigazione scegliere Impostazioni.

4. In Pipeline tags (Tag della pipeline), visualizzare i tag per la pipeline nelle colonne Key (Chiave) e Value (Valore).

Modifica di tag per una pipeline (console)

È possibile utilizzare la console per modificare i tag che sono stati aggiunti alle pipeline.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Pipelines, scegliere la pipeline in cui si desidera aggiornare i tag.
3. Nel riquadro di navigazione scegliere Impostazioni.
4. In Pipeline tags (Tag della pipeline), scegliere Modifica.
5. Nei campi Key (Chiave) e Value (Valore), aggiornare i valori di ogni campo in base alle esigenze. Ad esempio, per la chiave **`Project`**, in Value (Valore), modificare **`ProjectA`** in **`ProjectB`**.
6. Scegli Invia.

Rimozione di tag da una pipeline (console)

È possibile utilizzare la console per eliminare i tag dalle pipeline. Quando rimuovi i tag dalla risorsa associata, questi vengono eliminati.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Pipelines, scegliere la pipeline in cui si desidera rimuovere i tag.
3. Nel riquadro di navigazione scegliere Impostazioni.
4. In Pipeline tags (Tag della pipeline), scegliere Modifica.
5. Accanto alla chiave e al valori di ciascun tag che desideri eliminare, scegli Remove tag (Rimuovi tag).
6. Scegli Invia.

Tagging di pipeline (CLI)

È possibile utilizzare l'interfaccia a riga di comando per aggiungere tag alle risorse. È necessario utilizzare la console per gestire i tag nelle pipeline.

Argomenti

Aggiunta di tag a una pipeline (CLI)

Puoi usare la console o AWS CLI etichettare le pipeline.

Per aggiungere un tag a una pipeline al momento della creazione, consulta Creare una pipeline in CodePipeline.

In queste fasi, si assume che sia già installata una versione recente della AWS CLI o che sia aggiornata alla versione corrente. Per ulteriori informazioni, consulta l'argomento relativo all'installazione di AWS Command Line Interface.

Al terminale o alla riga di comando, esegui il comando tag-resource, specificando l'ARN (Amazon Resource Name) della pipeline in cui aggiungere i tag e la chiave e il valore del tag che desideri aggiungere. È possibile aggiungere più di un tag a una pipeline. *Ad esempio, per etichettare una pipeline denominata MyPipelinecon due tag, una chiave di tag denominata DeploymentEnvironmentcon il valore del tag Test e una chiave di tag denominata IscontainerBasedcon il valore del tag true:*

```
aws codepipeline tag-resource --resource-arn arn:aws:codepipeline:us-west-2:account-id:MyPipeline --tags key=Project,value=ProjectA key=IscontainerBased,value=true
```

In caso di successo, questo comando non restituisce alcun risultato.

Visualizzazione di tag per una pipeline (CLI)

Segui questi passaggi per utilizzare AWS CLI per visualizzare i AWS tag per una pipeline. Se non sono stati aggiunti tag, l'elenco restituito è vuoto.

Dal terminale o dalla riga di comando, esegui il comando list-tags-for-resource. Ad esempio, per visualizzare un elenco di chiavi e valori di tag per una pipeline denominata *MyPipeline*con il valore `arn:aws:codepipeline:`*`us-west-2`*`:`*`account-id`*`:`*`MyPipeline`* ARN:

```
aws codepipeline list-tags-for-resource --resource-arn arn:aws:codepipeline:us-west-2:account-id:MyPipeline
```

Se il comando viene eseguito correttamente, restituisce informazioni simili alle seguenti:

```
{
    "tags": {
        "Project": "ProjectA",
        "IscontainerBased": "true"
    }
}
```

Modifica di tag per una pipeline (CLI)

Segui questi passaggi per utilizzare per modificare un tag per una pipeline. AWS CLI È possibile modificare il valore di una chiave esistente o aggiungere un'altra chiave. È anche possibile rimuovere i tag da una pipeline, come illustrato nella sezione successiva.

Al terminale o nella riga di comando, esegui il comando tag-resource specificando l'ARN (Amazon Resource Name) della pipeline in cui desideri aggiornare un tag e specifica la chiave e il valore di tag:

```
aws codepipeline tag-resource --resource-arn arn:aws:codepipeline:us-west-2:account-id:MyPipeline --tags key=Project,value=ProjectA
```

In caso di successo, questo comando non restituisce alcun risultato.

Rimozione di tag da una pipeline (CLI)

Segui questi passaggi per utilizzare AWS CLI per rimuovere un tag da una pipeline. Quando rimuovi i tag dalla risorsa associata, questi vengono eliminati.

> **Note**
>
> Se si elimina una pipeline, tutte le associazioni di tag vengono rimosse dalla pipeline eliminata. Non è necessario rimuovere i tag prima di eliminare una pipeline.

Al terminale o nella riga di comando, esegui il comando untag-resource specificando l'ARN della pipeline in cui desideri rimuovere i tag e la chiave del tag che desideri rimuovere. Ad esempio,

per rimuovere più tag su una pipeline denominata *MyPipeline*con le chiavi dei tag *Project* e: *IscontainerBased*

```
aws codepipeline untag-resource --resource-arn arn:aws:codepipeline:us-west-2:account-id:MyPipeline --tag-keys Project IscontainerBased
```

In caso di successo, questo comando non restituisce alcun risultato. Per verificare i tag associati alla pipeline, esegui il comando list-tags-for-resource.

Creazione di una regola di notifica

È possibile utilizzare le regole di notifica per notificare agli utenti modifiche importanti, ad esempio quando una pipeline avvia l'esecuzione. Le regole di notifica specificano sia gli eventi che l'argomento Amazon SNS utilizzato per inviare le notifiche. Per ulteriori informazioni, vedere Cosa sono le notifiche?

È possibile utilizzare la console o creare regole di notifica AWS CLI per. AWS CodePipeline

Per creare una regola di notifica (console)

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. Scegliere Pipeline, quindi scegliere una pipeline in cui si desidera aggiungere le notifiche.
3. Nella pagina delle risorse, scegliere Notify (Notifica), quindi selezionare Create notification rule (Crea regola di notifica). È inoltre possibile accedere alla pagina Impostazioni per la pipeline e scegliere Crea regola di notifica.
4. In Notification name (Nome notifica), immettere un nome per la regola.
5. In Tipo di dettaglio, scegli Basic se desideri che nella notifica siano EventBridge incluse solo le informazioni fornite ad Amazon. Scegli Completo se desideri includere le informazioni fornite ad Amazon EventBridge e le informazioni che potrebbero essere fornite da CodePipeline o dal gestore delle notifiche.

   Per ulteriori informazioni, vedere Informazioni sul contenuto delle notifiche e sulla sicurezza.
6. In Events that trigger notifications (Eventi che attivano le notifiche), selezionare gli eventi per i quali si desidera inviare notifiche. Per ulteriori informazioni, consulta l'argomento relativo agli eventi per le regole di notifica sulle pipeline.

7. In Targets (Destinazioni), procedere in uno dei seguenti modi:

   - Se è già stata configurata una risorsa da utilizzare con le notifiche, in Choose target type (Scegli tipo di destinazione), scegliere AWS Chatbot (Slack) o SNS topic (Argomento SNS). In Scegli destinazione, scegli il nome del client (per un client Slack configurato in AWS Chatbot) o l'Amazon Resource Name (ARN) dell'argomento Amazon SNS (per gli argomenti Amazon SNS già configurati con la politica richiesta per le notifiche).
   - Se non è stata configurata una risorsa da utilizzare con le notifiche, scegliere Create target (Crea destinazione), e quindi scegliere SNS topic (Argomento SNS). Fornire un nome per l'argomento dopo codestar-notifications-, quindi scegliere Create (Crea).

   > **Note**
   >
   > - Se si crea l'argomento Amazon SNS come parte della creazione della regola di notifica, viene applicata la policy che consente alla funzionalità di notifica di pubblicare eventi nell'argomento. L'utilizzo di un argomento creato per le regole di notifica consente di iscrivere solo gli utenti che si desidera ricevano le notifiche relative a questa risorsa.
   > - Non puoi creare un AWS Chatbot client come parte della creazione di una regola di notifica. Se scegli AWS Chatbot (Slack), vedrai un pulsante che ti indirizza alla configurazione di un client in. AWS Chatbot Scegliendo questa opzione si apre la console. AWS Chatbot Per ulteriori informazioni, consulta Configurare le integrazioni tra notifiche e AWS Chatbot.
   > - Se desideri utilizzare un argomento Amazon SNS esistente come obiettivo, devi aggiungere la politica richiesta per AWS CodeStar le notifiche oltre a qualsiasi altra politica che potrebbe esistere per quell'argomento. Per ulteriori informazioni, consulta l'argomento relativo alla configurazione degli argomenti Amazon SNS per le notifiche e le informazioni sulla sicurezza e sui contenuti delle notifiche.

8. Per completare la creazione della regola, scegliere Invia.
9. È necessario iscrivere gli utenti all'argomento Amazon SNS relativo alla regola prima che possano ricevere notifiche. Per ulteriori informazioni, consulta Sottoscrivere gli utenti agli argomenti di Amazon SNS relativi agli obiettivi. Puoi anche configurare l'integrazione tra le notifiche e inviare notifiche AWS Chatbot alle chat room di Amazon Chime o ai canali Slack. Per ulteriori informazioni, consulta Configurare l'integrazione tra notifiche e. AWS Chatbot

Per creare una regola di notifica (AWS CLI)

1. Da un terminale o dal prompt dei comandi, eseguire il comando create-notification rule per generare lo skeleton JSON:

```
aws codestar-notifications create-notification-rule --generate-cli-skeleton
 > rule.json
```

È possibile assegnare al file un nome qualsiasi. In questo esempio, il file è denominato *rule.json*.

2. Aprire il file JSON in un editor di testo normale e modificarlo per includere la risorsa, i tipi di evento e la destinazione desiderata per la regola. *L'esempio seguente mostra una regola di notifica denominata* ***MyNotificationRule*** *per una pipeline denominata MyDemoPipelinein un AWS account con l'ID 123456789012.* Le notifiche vengono inviate con il tipo di dettaglio completo a un argomento di Amazon SNS denominato *codestar-notifications, quando iniziano le esecuzioni della pipeline: MyNotificationTopic*

```
{
    "Name": "MyNotificationRule",
    "EventTypeIds": [
        "codepipeline-pipeline-pipeline-execution-started"
    ],
    "Resource": "arn:aws:codebuild:us-east-2:123456789012:MyDemoPipeline",
    "Targets": [
        {
            "TargetType": "SNS",
            "TargetAddress": "arn:aws:sns:us-east-2:123456789012:codestar-
notifications-MyNotificationTopic"
        }
    ],
    "Status": "ENABLED",
    "DetailType": "FULL"
}
```

Salvare il file.

3. Utilizzando il file appena modificato, dal terminale o dalla riga di comando, eseguire nuovamente il comando create-notification-rule per creare la regola di notifica:

```
aws codestar-notifications create-notification-rule --cli-input-json
 file://rule.json
```

4. In caso di esito positivo, il comando restituisce l'ARN della regola di notifica, simile al seguente:

```
{
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
}
```

Utilizzo dei trigger in CodePipeline

I trigger consentono di configurare la pipeline in modo che inizi in base a un particolare tipo di evento o a un tipo di evento filtrato, ad esempio quando viene rilevata una modifica su un particolare branch o pull request. I trigger sono configurabili per le azioni di origine con connessioni che utilizzano l'`CodeStarSourceConnection`azione in CodePipeline, ad esempio Bitbucket e GitHub. GitLab

Le azioni di origine, come CodeCommit e S3, utilizzano il rilevamento delle modifiche, come descritto in questa sezione sull'avvio delle pipeline.

Puoi aggiungere un trigger alla tua pipeline e configurarlo per filtrare in base a eventi particolari

I trigger vengono specificati utilizzando la console o la CLI.

Filtra i trigger nelle richieste push o pull di codice

Puoi configurare i filtri per i trigger della pipeline per avviare le esecuzioni della pipeline per diversi eventi Git, come tag o branch push, modifiche a percorsi di file specifici, una richiesta pull aperta in un ramo specifico e così via. Puoi usare la AWS CodePipeline console o i update-pipeline comandi create-pipeline and in AWS CLI per configurare i filtri dei trigger.

È possibile specificare filtri per i seguenti tipi di trigger:

- Push

  Un push trigger avvia una pipeline quando una modifica viene inviata al repository di origine. L'esecuzione utilizzerà il commit dal ramo verso cui stai inviando il push (ovvero il ramo di destinazione). Puoi filtrare i trigger push su rami, percorsi di file o tag Git.

- Richiesta pull

  Un trigger di pull request avvia una pipeline quando una pull request viene aperta, aggiornata o chiusa nel repository di origine. L'esecuzione utilizzerà il commit dal ramo di origine da cui stai estraendo (ovvero il ramo di origine). Puoi filtrare i trigger delle pull request sui rami e sui percorsi dei file.

> **Note**
>
> I tipi di eventi supportati per le richieste pull vengono aperti, aggiornati o chiusi (uniti). Tutti gli altri eventi di pull request vengono ignorati.

La definizione della pipeline consente di combinare diversi filtri all'interno della stessa configurazione push trigger. Per informazioni dettagliate sulla definizione della pipeline, vedere. Filtraggio dei trigger nella pipeline JSON (CLI) Le combinazioni valide sono:

- tags
- rami
- rami + percorsi di file

I tipi di filtro vengono specificati come segue:

- Nessun filtro

  Questa configurazione di trigger avvia la pipeline su qualsiasi push al ramo predefinito specificato come parte della configurazione dell'azione.

- Specificare il filtro

  Aggiungi un filtro che avvia la pipeline su un filtro specifico, ad esempio sui nomi delle filiali per un push di codice, e recupera il commit esatto. Ciò configura anche la pipeline in modo che non si avvii automaticamente in caso di modifica.

- Non rileva le modifiche

  Ciò non aggiunge un trigger e la pipeline non si avvia automaticamente in caso di modifica.

La tabella seguente fornisce opzioni di filtro valide per ogni tipo di evento. La tabella mostra anche quali configurazioni di attivazione sono predefinite su true o false per il rilevamento automatico delle modifiche nella configurazione dell'azione.

Configurazione dei trigger	Tipo di evento	Opzioni di filtro	Rileva le modifiche
Aggiungi un trigger, nessun filtro	nessuno	nessuno	true
Aggiungi un trigger: filtra in base al codice push	evento push	Tag Git, rami, percorsi di file	false
Aggiungi un trigger: filtro per le richieste pull	richieste pull	rami, percorsi di file	false
Nessun trigger: non rileva	nessuno	nessuno	false

> **Note**
>
> Questo tipo di trigger utilizza il rilevamento automatico delle modifiche (come tipo di `Webhook` trigger). I source action provider che utilizzano questo tipo di trigger sono connessioni configurate per code push (Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab autogestite).

Per il filtraggio, sono supportati i modelli di espressioni regolari in formato glob, come descritto in. Lavorare con i modelli a globo nella sintassi

> **Note**
>
> In alcuni casi, per le pipeline con trigger filtrati sui percorsi dei file, la pipeline potrebbe non avviarsi quando viene creato per la prima volta un ramo con un filtro per il percorso dei file. Per ulteriori informazioni, consulta Le pipeline con connessioni che utilizzano il filtraggio dei trigger in base ai percorsi dei file potrebbero non iniziare alla creazione del ramo.

Argomenti

Considerazioni sui filtri di attivazione

Le seguenti considerazioni si applicano all'utilizzo dei trigger.

- Per un trigger con filtri per rami e percorsi di file, quando si preme il ramo per la prima volta, la pipeline non verrà eseguita poiché non è possibile accedere all'elenco dei file modificati per il ramo appena creato.
- L'unione di una richiesta pull potrebbe innescare due esecuzioni di pipeline, nei casi in cui le configurazioni dei trigger push (branch filter) e pull request (branch filter) si intersecano.

Esempi di filtri trigger

Per una configurazione Git con filtri per i tipi di eventi push e pull request, i filtri specificati potrebbero entrare in conflitto tra loro. I seguenti sono esempi di combinazioni di filtri valide per gli eventi di richieste push e pull.

Quando i clienti combinano filtri all'interno di un singolo oggetto di configurazione, questi filtri utilizzeranno un'operazione AND, il che significa che solo una corrispondenza completa avvierà la pipeline. L'esempio seguente mostra la configurazione Git:

```
{
  "filePaths": {
    "includes": ["common/**/*.js"]
  },
  "branches": {
    "includes": ["feature/**"]
  }
}
```

Con la configurazione Git precedente, questo esempio mostra un evento che avvierà l'esecuzione della pipeline perché l'operazione AND ha esito positivo.

```
{
  "ref": "refs/heads/feature/triggers",
  ...
  "commits": [
    {
      ...
      "modified": [
        "common/app.js"
      ]
      ...
    }
  ]
}
```

Questo esempio mostra un evento che non avvia l'esecuzione della pipeline perché il ramo è in grado di filtrare, ma il percorso del file no.

```
{
   "ref": "refs/heads/feature/triggers",
  ...
  "commits": [
    {
      ...
      "modified": [
        "src/Main.java"
      ]
      ...
    }
  ]
}
```

Allo stesso tempo, gli oggetti di configurazione trigger all'interno dell'array push utilizzano un'operazione OR. Ciò consente di configurare più trigger per avviare l'esecuzione per la stessa pipeline. Per un elenco delle definizioni dei campi nella struttura JSON, vedi. Filtraggio dei trigger nella pipeline JSON (CLI)

Filtraggio in base agli eventi push (console)

Puoi utilizzare la console per aggiungere filtri per gli eventi push e includere o escludere rami o percorsi di file.

Filtraggio in base agli eventi push (console)

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi e lo stato di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegliere il nome della pipeline da modificare. Altrimenti, utilizzate questi passaggi nella procedura guidata di creazione della pipeline.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. Nella pagina Modifica, scegliete l'azione sorgente che desiderate modificare. Scegli Modifica trigger. Scegli Specificare filtro.
5. In Tipo di evento, scegli Push tra le seguenti opzioni.

   - Scegli Push per avviare la pipeline quando una modifica viene inviata al tuo repository di origine. Questa opzione consente ai campi di specificare filtri per rami e percorsi di file o tag Git.
   - Scegli Pull request per avviare la pipeline quando una pull request viene aperta, aggiornata o chiusa nel tuo repository di origine. Questa opzione consente ai campi di specificare filtri per i rami di destinazione e i percorsi dei file.
6. In Tipo di filtro, scegliete una delle seguenti opzioni.

   - Scegliete Branch per specificare i rami del repository di origine monitorati dal trigger per sapere quando avviare l'esecuzione di un flusso di lavoro. In Include, inserite gli schemi per i nomi dei rami in formato glob che desiderate specificare per la configurazione del trigger per avviare la pipeline in base alle modifiche nei rami specificati. In Exclude, inserite i modelli regex per i nomi dei rami in formato glob che desiderate specificare affinché la configurazione del trigger ignori e non avvii la pipeline in base alle modifiche nei rami specificati. Per ulteriori informazioni, consulta Lavorare con i modelli a globo nella sintassi.

> **Note**
>
> Se l'inclusione e l'esclusione hanno entrambe lo stesso schema, l'impostazione predefinita prevede l'esclusione del pattern.

È possibile utilizzare modelli regex in formato glob per definire i nomi dei rami. Ad esempio, `main.*` da utilizzare per abbinare tutti i rami che iniziano con. `main.*` Per ulteriori informazioni, consulta Lavorare con i modelli a globo nella sintassi.

Per attivare un pulsante, specifica i rami verso cui stai effettuando il push, ovvero i rami di destinazione. Per attivare una pull request, specifica i rami di destinazione verso cui stai aprendo la pull request.

- (Facoltativo) In Percorsi dei file, specifica i percorsi dei file per il trigger. Immettete i nomi nelle caselle Includi ed Escludi, a seconda dei casi.

  È possibile utilizzare modelli regex in formato glob per definire i nomi dei percorsi dei file. Ad esempio, utilizzare per `prod.*` abbinare tutti i percorsi dei file che iniziano con. `prod.*` Per ulteriori informazioni, consulta Lavorare con i modelli a globo nella sintassi.
- Scegli Tag per configurare la configurazione del trigger della pipeline per iniziare con i tag Git. In Includi, inserisci i modelli per i nomi dei tag in formato glob che desideri specificare per la configurazione del trigger per avviare la pipeline al rilascio del tag o dei tag specificati. In Exclude, inserite i pattern regex per i nomi dei tag in formato glob che desiderate specificare affinché la configurazione del trigger ignori e non avvii la pipeline al rilascio del tag o dei tag specificati. Se include ed esclude hanno entrambi lo stesso schema di tag, l'impostazione predefinita prevede l'esclusione del modello di tag.

Filtraggio in base alle richieste pull (console)

È possibile utilizzare la console per aggiungere filtri per le richieste pull con eventi specifici e includere o escludere rami o percorsi di file.

Filtraggio in base alle richieste pull (console)

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

Vengono visualizzati i nomi e lo stato di tutte le pipeline associate al tuo AWS account.

2. In Name (Nome), scegliere il nome della pipeline da modificare. Altrimenti, utilizzate questi passaggi nella procedura guidata di creazione della pipeline.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. Nella pagina Modifica, scegliete l'azione sorgente che desiderate modificare. Scegli Modifica trigger. Scegli Specificare filtro.
5. In Tipo di evento, scegli Pull request tra le seguenti opzioni.

   - Scegli Push per avviare la pipeline quando una modifica viene inviata al tuo repository di origine. Questa opzione consente ai campi di specificare filtri per rami e percorsi di file o tag Git.
   - Scegli Pull request per avviare la pipeline quando una pull request viene aperta, aggiornata o chiusa rispetto ai rami di destinazione specificati. La selezione di questa opzione consente ai campi di specificare filtri per rami e percorsi di file.

     Facoltativamente, puoi specificare i seguenti eventi di pull request da filtrare:

     - Viene creata la pull request
     - Viene apportata una nuova revisione alla pull request
     - La pull request è chiusa
6. In Tipo di filtro, scegli una delle seguenti opzioni.

   - Scegliete Branch per specificare i rami del repository di origine monitorati dal trigger per sapere quando avviare l'esecuzione di un flusso di lavoro. In Include, inserite gli schemi per i nomi dei rami in formato glob che desiderate specificare per la configurazione del trigger per avviare la pipeline in base alle modifiche nei rami specificati. In Exclude, inserite i modelli regex per i nomi dei rami in formato glob che desiderate specificare affinché la configurazione del trigger ignori e non avvii la pipeline in base alle modifiche nei rami specificati. Per ulteriori informazioni, consulta Lavorare con i modelli a globo nella sintassi.

     > **Note**
     >
     > Se l'inclusione e l'esclusione hanno entrambe lo stesso schema, l'impostazione predefinita prevede l'esclusione del pattern.

È possibile utilizzare modelli regex in formato glob per definire i nomi dei rami. Ad esempio, `main.*` da utilizzare per abbinare tutti i rami che iniziano con. `main.*` Per ulteriori informazioni, consulta Lavorare con i modelli a globo nella sintassi.

Per attivare un pulsante, specifica i rami verso cui stai effettuando il push, ovvero i rami di destinazione. Per attivare una pull request, specifica i rami di destinazione verso cui stai aprendo la pull request.

- (Facoltativo) In Percorsi di file, specifica i nomi dei percorsi dei file per il trigger. Immettete i nomi nelle caselle Includi ed Escludi, a seconda dei casi.

  È possibile utilizzare modelli regex in formato glob per definire i nomi dei percorsi dei file. Ad esempio, utilizzare per `prod.*` abbinare tutti i percorsi dei file che iniziano con. `prod.*` Per ulteriori informazioni, consulta Lavorare con i modelli a globo nella sintassi.

Filtraggio dei trigger nella pipeline JSON (CLI)

Puoi aggiornare la pipeline JSON per aggiungere filtri per i trigger.

Per utilizzare il AWS CLI per creare o aggiornare la pipeline, usa il comando o. `create-pipeline` `update-pipeline`

La seguente struttura JSON di esempio fornisce un riferimento per le definizioni di campo riportate di seguito. `create-pipeline`

```
{
    "pipeline": {
        "name": "MyServicePipeline",
        "triggers": [
            {
                "provider": "Connection",
                "gitConfiguration": {
                    "sourceActionName": "ApplicationSource",
                    "push": [
                        {
                            "filePaths": {
                                "includes": [
                                    "projectA/**",
                                    "common/**/*.js"
                                ],
```

```
                        "excludes": [
                            "**/README.md",
                            "**/LICENSE",
                            "**/CONTRIBUTING.md"
                        ]
                    },
                    "branches": {
                        "includes": [
                            "feature/**",
                            "release/**"
                        ],
                        "excludes": [
                            "mainline"
                        ]
                    },
                    "tags": {
                        "includes": [
                            "release-v0", "release-v1"
                        ],
                        "excludes": [
                            "release-v2"
                        ]
                    }
                }
            ],
            "pullRequest": [
                {
                    "events": [
                        "CLOSED"
                    ],
                    "branches": {
                        "includes": [
                            "feature/**",
                            "release/**"
                        ],
                        "excludes": [
                            "mainline"
                        ]
                    },
                    "filePaths": {
                        "includes": [
                            "projectA/**",
                            "common/**/*.js"
                        ],
```

```
                                "excludes": [
                                    "**/README.md",
                                    "**/LICENSE",
                                    "**/CONTRIBUTING.md"
                                ]
                            }
                        }
                    ]
                }
            }
        ],
        "stages": [
            {
                "name": "Source",
                "actions": [
                    {
                        "name": "ApplicationSource",
                        "configuration": {
                            "BranchName": "mainline",
                            "ConnectionArn": "arn:aws:codestar-connections:eu-
central-1:111122223333:connection/fe9ff2e8-ee25-40c9-829e-65f8EXAMPLE",
                            "FullRepositoryId": "monorepo-example",
                            "OutputArtifactFormat": "CODE_ZIP"
                        }
                    }
                ]
            }
        ]
    }
}
```

I campi nella struttura JSON sono definiti come segue:

- `sourceActionName`: il nome dell'azione di origine della pipeline con la configurazione Git.
- `push`: eventi push con filtro. Questi eventi utilizzano un'operazione OR tra diversi filtri push e un'operazione AND all'interno dei filtri.
  - `branches`: I rami su cui filtrare. Le filiali utilizzano un'operazione AND tra le inclusioni e le esclusioni.
    - `includes`: modelli in base ai quali filtrare i rami che verranno inclusi. Include l'uso di un'operazione OR.

    - `excludes`: modelli in base ai quali filtrare i rami che verranno esclusi. Esclude l'uso di un'operazione OR.
  - `filePaths`: i nomi dei percorsi dei file in base ai quali filtrare.
    - `includes`: modelli in base ai quali filtrare i percorsi dei file che verranno inclusi. Include l'uso di un'operazione OR.
    - `excludes`: modelli in base ai quali filtrare i percorsi di file che verranno esclusi. Esclude l'utilizzo di un'operazione OR.
  - `tags`: i nomi dei tag in base ai quali filtrare.
    - `includes`: modelli in base ai quali filtrare i tag che verranno inclusi. Include l'uso di un'operazione OR.
    - `excludes`: modelli in base ai quali filtrare i tag che verranno esclusi. Esclude l'uso di un'operazione OR.
- `pullRequest`: eventi di pull request con filtraggio sugli eventi di pull request e sui filtri di pull request.
  - `events`: Filtra sugli eventi di pull request aperti, aggiornati o chiusi come specificato.
  - `branches`: I rami su cui filtrare. Le filiali utilizzano un'operazione AND tra le inclusioni e le esclusioni.
    - `includes`: modelli in base ai quali filtrare i rami che verranno inclusi. Include l'uso di un'operazione OR.
    - `excludes`: modelli in base ai quali filtrare i rami che verranno esclusi. Esclude l'uso di un'operazione OR.
  - `filePaths`: i nomi dei percorsi dei file in base ai quali filtrare.
    - `includes`: modelli in base ai quali filtrare i percorsi dei file che verranno inclusi. Include l'uso di un'operazione OR.
    - `excludes`: modelli in base ai quali filtrare i percorsi di file che verranno esclusi. Esclude l'utilizzo di un'operazione OR.

Attiva il filtraggio nei modelli AWS CloudFormation

Puoi aggiornare la risorsa della pipeline in AWS CloudFormation per aggiungere il filtro dei trigger.

Il seguente frammento di modello di esempio fornisce un riferimento YAML per le definizioni dei campi dei trigger. Per un elenco delle definizioni dei campi, vedere. Filtraggio dei trigger nella pipeline JSON (CLI)

```
pipeline:
  name: MyServicePipeline
  executionMode: PARALLEL
  triggers:
    - provider: CodeConnection
      gitConfiguration:
        sourceActionName: ApplicationSource
        push:
          - filePaths:
              includes:
                - projectA/**
                - common/**/*.js
              excludes:
                - '**/README.md'
                - '**/LICENSE'
                - '**/CONTRIBUTING.md'
            branches:
              includes:
                - feature/**
                - release/**
              excludes:
                - mainline
          - tags:
              includes:
                - release-v0
                - release-v1
              excludes:
                - release-v2
        pullRequest:
          - events:
              - CLOSED
            branches:
              includes:
                - feature/**
                - release/**
              excludes:
                - mainline
            filePaths:
              includes:
                - projectA/**
                - common/**/*.js
              excludes:
                - '**/README.md'
```

```
                - '**/LICENSE'
                - '**/CONTRIBUTING.md'
  stages:
    - name: Source
      actions:
        - name: ApplicationSource
          configuration:
            BranchName: mainline
            ConnectionArn: arn:aws:codestar-connections:eu-
central-1:111122223333:connection/fe9ff2e8-ee25-40c9-829e-65f85EXAMPLE
            FullRepositoryId: monorepo-example
            OutputArtifactFormat: CODE_ZIP
```

Gestisci le esecuzioni in CodePipeline

Per analizzare l'avanzamento della pipeline, puoi visualizzare i log degli errori, visualizzare la cronologia di esecuzione della pipeline e delle azioni e riprovare le fasi o le azioni non riuscite.

Argomenti

Visualizza le esecuzioni in CodePipeline

È possibile utilizzare la AWS CodePipeline console o il AWS CLI per visualizzare lo stato di esecuzione, visualizzare la cronologia di esecuzione e riprovare le fasi o le azioni non riuscite.

Argomenti

Visualizzazione della cronologia delle esecuzioni della pipeline (console)

Puoi utilizzare la CodePipeline console per visualizzare un elenco di tutte le pipeline del tuo account. Puoi anche visualizzare i dettagli per ogni pipeline, inclusa la data dell'ultima esecuzione delle operazioni nella pipeline, se una transizione tra fasi è abilitata o disabilitata, se eventuali operazioni non sono riuscite e altre informazioni. Inoltre, puoi visualizzare una pagina della cronologia che mostra i dettagli di tutte le esecuzioni della pipeline per le quali la cronologia è stata registrata.

> **Note**
>
> Quando si passa da una modalità di esecuzione specifica, la visualizzazione e la cronologia della pipeline potrebbero cambiare. Per ulteriori informazioni, consulta Impostare o modificare la modalità di esecuzione della pipeline.

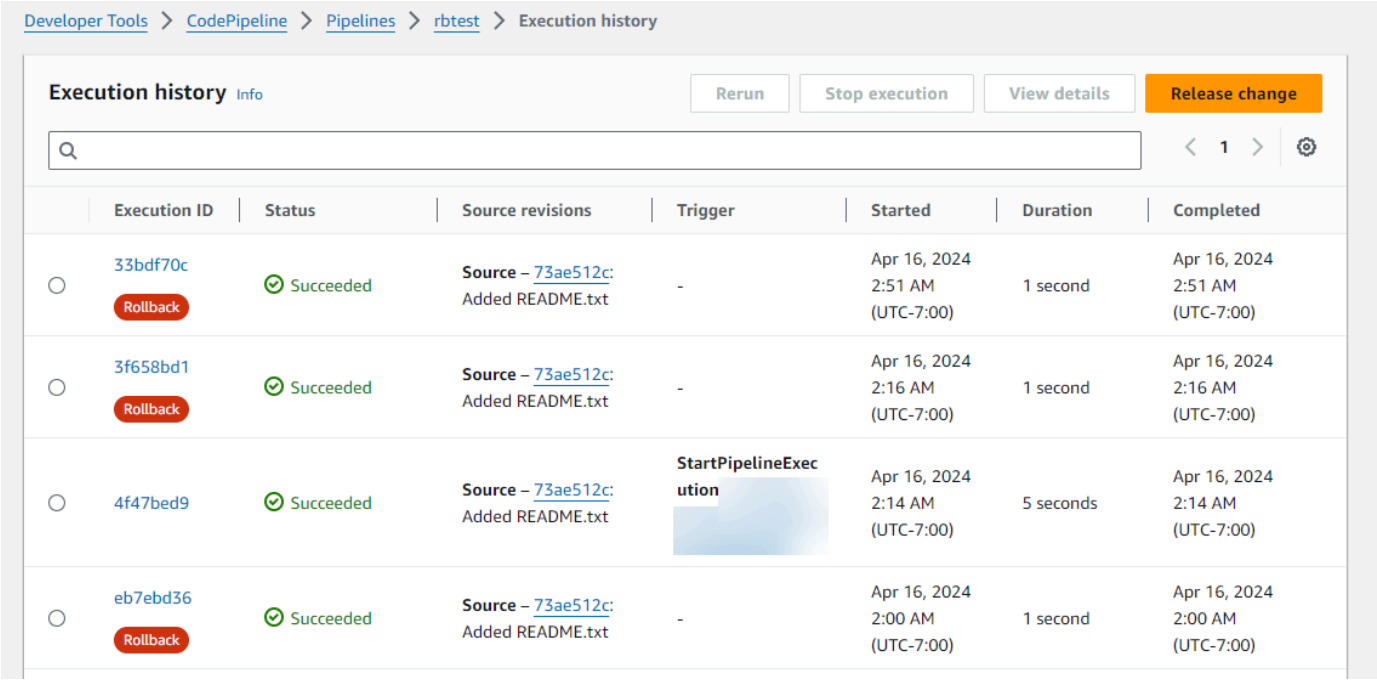
La cronologia delle esecuzioni viene conservata fino a 12 mesi.

Puoi utilizzare la console per visualizzare la cronologia delle esecuzioni in una pipeline, inclusi lo stato, le revisioni di origine e i dettagli temporali di ogni esecuzione.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account, insieme al relativo stato.
2. In Name (Nome), scegli il nome della pipeline.
3. Scegli View history (Visualizza cronologia).

> **Note**
>
> Per una pipeline in modalità di esecuzione PARALLEL, la vista principale della pipeline non mostra la struttura della pipeline o le esecuzioni in corso. Per una pipeline in modalità di esecuzione PARALLEL, si accede alla struttura della pipeline scegliendo l'ID per l'esecuzione che si desidera visualizzare dalla pagina della cronologia di esecuzione. Scegli Cronologia nella barra di navigazione a sinistra, scegli l'ID di esecuzione per l'esecuzione parallela, quindi visualizza la pipeline nella scheda Visualizzazione.



4. Visualizzare lo stato, le revisioni dell'origine, i dettagli delle modifiche e i trigger correlat a ciascuna esecuzione della pipeline. Le esecuzioni di pipeline che sono state ripristinate mostreranno il tipo di esecuzione Rollback nella schermata dei dettagli della console. Per l'esecuzione non riuscita che ha attivato il rollback automatico, viene visualizzato l'ID di esecuzione non riuscita.
5. Scegliere un'esecuzione. La vista dettagliata mostra i dettagli dell'esecuzione, la scheda Cronologia, la scheda Visualizzazione e la scheda Variabili. I valori delle variabili per le variabili a livello di pipeline vengono risolti al momento dell'esecuzione della pipeline e possono essere visualizzati nella cronologia di esecuzione per ogni esecuzione.

> **Note**
>
> Le variabili di output delle azioni della pipeline possono essere visualizzate nella scheda Variabili di output nella cronologia di ogni esecuzione di azione.

Visualizzazione dello stato dell'esecuzione (console)

Puoi visualizzare lo stato della pipeline in Status (Stato) nella pagina della cronologia delle esecuzioni. Scegli un collegamento ID di esecuzione e quindi visualizza lo stato dell'operazione.

I seguenti sono stati validi per pipeline, fasi e operazioni:

> **Note**
>
> I seguenti stati della pipeline si applicano anche a un'esecuzione della pipeline che è un'esecuzione in entrata. Per visualizzare un'esecuzione in entrata e il relativo stato, vedere. Visualizzare un'esecuzione in entrata (Console)

Stati a livello della pipeline

Stato della pipeline	Descrizione
InProgress	L'esecuzione della pipeline è attualmente in corso.
In arresto	L'esecuzione della pipeline si interrompe a causa di una richiesta di arrestare e attendere o interrompere e abbandonare l'esecuzione della pipeline.
Arrestato	Il processo di arresto è completo e l'esecuzione della pipeline viene interrotta.
Riuscito	L'esecuzione della pipeline è stata completata.
Sostituito	Mentre l'esecuzione di questa pipeline era in attesa del completamento della fase successiva, una nuova esecuzione della pipeline è avanzata e ha continuato nella pipeline.
Non riuscito	L'esecuzione della pipeline non è stata completata.

Stati a livello della fase

Stato della fase	Descrizione
InProgress	La fase è attualmente in esecuzione.
In arresto	L'esecuzione dello stage si interrompe a causa di una richiesta di arrestare e attendere o interrompere e abbandonare l'esecuzione della pipeline.

Stato della fase	Descrizione
Arrestato	Il processo di arresto è completo e l'esecuzione dello stage viene interrotta.
Riuscito	La fase è stata completata.
Non riuscito	La fase non è stata completata.

Stati a livello di operazione

Stato dell'operazione	Descrizione
InProgress	L'operazione è attualmente in esecuzione.
Abbandonati	L'azione è abbandonata a causa di una richiesta di arrestare e abbandonare l'esecuzione della pipeline.
Riuscito	L'operazione è stata completata.
Non riuscito	Per le operazioni di approvazione, lo stato NON RIUSCITA significa che l'operazione è stata rifiutata dal revisore o che non è riuscita a causa di un errore di configurazione dell'operazione.

Visualizzare un'esecuzione in entrata (Console)

È possibile utilizzare la console per visualizzare lo stato e i dettagli di un'esecuzione in entrata. Quando la transizione è abilitata o la fase diventa disponibile, un'esecuzione in entrata `InProgress` continua ed entra nella fase. Un'esecuzione in entrata con uno `Stopped` stato non entra nella fase. Lo stato di esecuzione in entrata cambia `Failed` se la pipeline viene modificata. Quando si modifica una pipeline, tutte le esecuzioni in corso non continuano e lo stato di esecuzione cambia in. `Failed`

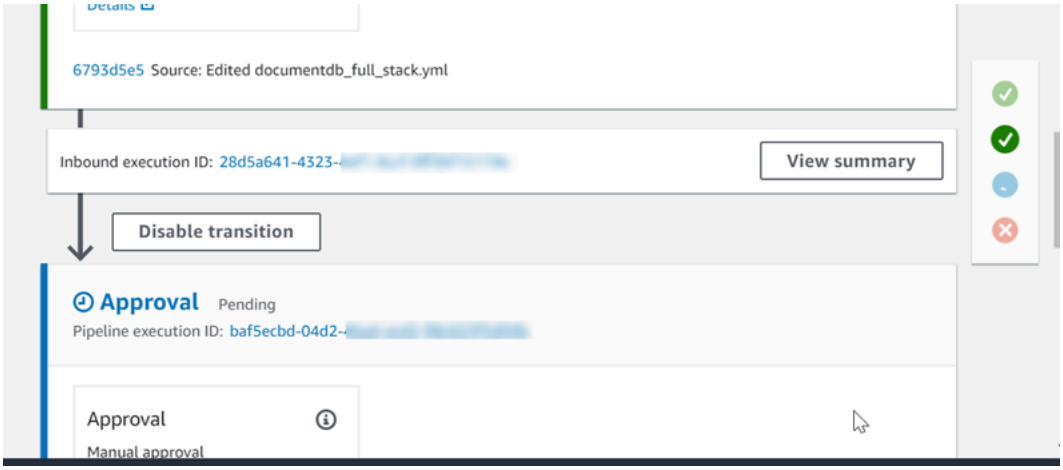
Se non vedi un'esecuzione in entrata, significa che non ci sono esecuzioni in sospeso in una fase di transizione disattivata.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Verranno visualizzati i nomi di tutte le pipeline associate al tuo AWS account.

2. Scegli il nome della pipeline di cui desideri visualizzare l'esecuzione in entrata, esegui una delle seguenti operazioni:

   - Scegliere Visualizza. Nel diagramma della pipeline, nel campo ID di esecuzione in entrata davanti alla transizione disabilitata, puoi visualizzare l'ID di esecuzione in entrata.



     Scegli Visualizza riepilogo per visualizzare i dettagli dell'esecuzione, come l'ID di esecuzione, il trigger di origine e il nome della fase successiva.

   - Scegli la pipeline e scegli Visualizza cronologia.

Visualizzazione delle revisioni dell'origine delle esecuzioni della pipeline (console)

Puoi visualizzare i dettagli relativi agli artefatti di origine (artefatto di output originato nella prima fase di una pipeline) che sono utilizzati nell'esecuzione di una pipeline. I dettagli includono identificatori, ad esempio gli ID commit, i commenti di check-in e, quando si utilizza l'interfaccia a riga di comando, i numeri di versione delle operazioni di compilazione della pipeline. Per alcuni tipi di revisione, puoi visualizzare e aprire l'URL del commit. Le revisioni di origine sono composte dai seguenti elementi:

- Summary: informazioni di riepilogo sulla versione più recente dell'artefatto. Per GitHub e CodeCommit repository, il messaggio di commit. Per i bucket o le azioni Amazon S3, il contenuto fornito dall'utente di una codepipeline-artifact-revision-summary chiave specificata nei metadati dell'oggetto.
- revisionUrl: l'URL di revisione per la revisione dell'artefatto (ad esempio, l'URL del repository esterno).

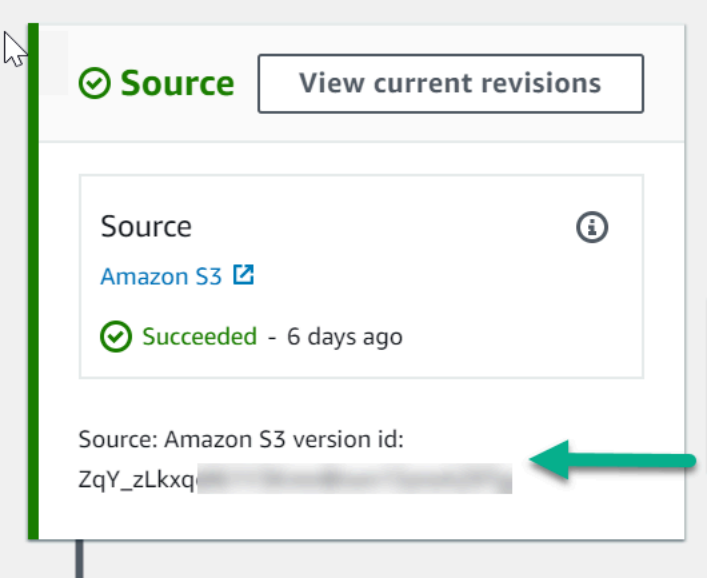
- revisionId: l'ID della revisione per la revisione dell'artefatto. Ad esempio, per una modifica alla fonte in un GitHub repository CodeCommit or, questo è l'ID di commit. Per gli artefatti archiviati nei nostri GitHub CodeCommit repository, l'ID di commit è collegato a una pagina dei dettagli del commit.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

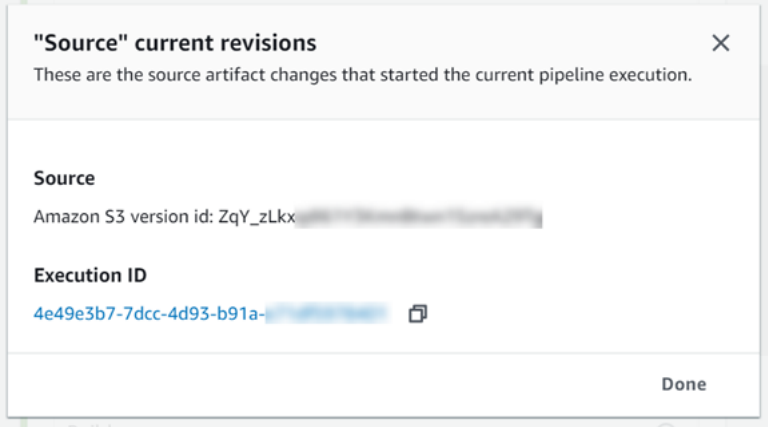
   Account AWS Verranno visualizzati i nomi di tutte le pipeline associate alla tua.

2. Scegliere il nome della pipeline per la quale si desidera visualizzare i dettagli di revisione dell'origine. Esegui una di queste operazioni:

   - Scegli View history (Visualizza cronologia). In Source revisions (Revisioni di origine), viene elencata la modifica all'origine di ogni esecuzione.
   - Individuare un'operazione per la quale si desidera visualizzare i dettagli di revisione dell'origine, quindi trovare le informazioni di revisione nella parte inferiore della relativa fase:



   Scegliere Visualizza revisioni correnti per visualizzare le informazioni di origine. Ad eccezione degli artefatti archiviati nei bucket Amazon S3, gli identificatori come gli ID di commit in questa visualizzazione dettagliata delle informazioni sono collegati alle pagine di informazioni di origine degli artefatti.



Visualizzazione delle esecuzioni delle operazioni (console)

Puoi visualizzare i dettagli di un'operazione per una pipeline, ad esempio l'ID di esecuzione dell'operazione, gli artefatti di input, gli artefatti di output e lo stato. Puoi visualizzare i dettagli di un'operazione scegliendo una pipeline nella console, quindi scegliendo un ID di esecuzione.

1. Accedi e apri la console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home. AWS Management Console CodePipeline

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. Scegli il nome della pipeline di cui desideri visualizzare i dettagli dell'operazione, quindi scegli View history (Visualizza cronologia).
3. In Execution ID (ID di esecuzione), scegli l'ID di esecuzione di cui desideri visualizzare i dettagli di esecuzione dell'operazione.
4. Puoi visualizzare le seguenti informazioni sulla scheda Timeline (Pianificazione):

   a. In Action name (Nome operazione), scegli il link per aprire una pagina dei dettagli per l'operazione, in cui è possibile visualizzare lo stato, il nome della fase, il nome dell'operazione, i dati di configurazione e le informazioni sugli artefatti.

   b. In Provider, scegli il link per visualizzare i dettagli del provider dell'operazione. Ad esempio, nella pipeline dell'esempio precedente, se si sceglie tra CodeDeploy le fasi di gestione temporanea o di produzione, viene visualizzata la pagina della CodeDeploy console per l' CodeDeploy applicazione configurata per quella fase.

Visualizzazione delle informazioni sugli artefatti delle operazioni e sull'archivio di artefatti (console)

Puoi visualizzare i dettagli sugli artefatti di input e di output per un'operazione. Puoi anche scegliere un link che porta alle informazioni sugli artefatti per quell'operazione. Poiché l'archivio di artefatti utilizza la funzione Versioni multiple, ogni esecuzione di operazione ha un percorso univoco per gli artefatti di input e output.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. Scegli il nome della pipeline di cui desideri visualizzare i dettagli dell'operazione, quindi scegli View history (Visualizza cronologia).
3. In Execution ID (ID di esecuzione), scegli l'ID di esecuzione di cui desideri visualizzare i dettagli dell'operazione.
4. Nella scheda Timeline, in Action name (Nome operazione), scegli il link per aprire una pagina di dettagli per l'operazione.
5. Nella pagina dei dettagli, nella scheda Esecuzione, visualizza lo stato e la tempistica dell'esecuzione dell'azione.
6. Nella scheda Configurazione, visualizza la configurazione delle risorse per l'azione (ad esempio, il nome del progetto di CodeBuild build).
7. Nella scheda Artefatti, visualizza i dettagli dell'artefatto in Tipo di artefatto e Fornitore di artefatto. Scegli il link in Artifact name (Nome artefatto) per visualizzare gli artefatti nell'archivio di artefatti.
8. Nella scheda Variabili di output, visualizza le variabili risolte dalle azioni nella pipeline per l'esecuzione dell'azione.

Visualizzazione dei dettagli e della cronologia della pipeline (CLI)

Puoi eseguire i seguenti comandi per visualizzare i dettagli relativi alla pipeline e alle esecuzioni della pipeline:

- list-pipelinescomando per visualizzare un riepilogo di tutte le pipeline associate al tuo. Account AWS
- Comando get-pipeline per rivedere i dettagli di una singola pipeline.

- list-pipeline-executions per visualizzare i riepiloghi delle esecuzioni più recenti per una pipeline.
- get-pipeline-execution per visualizzare informazioni relative a un'esecuzione di una pipeline, inclusi i dettagli sugli artefatti, l'ID di esecuzione della pipeline, nonché il nome, la versione e lo stato della pipeline.
- Comando get-pipeline-state per visualizzare pipeline, fase e stato dell'operazione.
- list-action-executions per visualizzare i dettagli di esecuzione delle operazioni per una pipeline.

Argomenti

Visualizza la cronologia di esecuzione con **`list-pipeline-executions`** (CLI)

Puoi visualizzare la cronologia di esecuzione della pipeline.

- Per visualizzare i dettagli relativi alle esecuzioni passate di una pipeline, eseguire il comando list-pipeline-executions, specificando il nome univoco della pipeline. Ad esempio, per visualizzare i dettagli sullo stato corrente di una pipeline denominata *`MyFirstPipeline`*, inserisci quanto segue:

  ```
  aws codepipeline list-pipeline-executions --pipeline-name MyFirstPipeline
  ```

  Questo comando restituisce informazioni di riepilogo relative a tutte le esecuzioni della pipeline per le quali è stata registrata la cronologia. Il riepilogo include le ore di inizio e di fine, la durata e lo stato.

  Le esecuzioni di pipeline di cui è stato eseguito il rollback mostreranno il tipo di esecuzione. `Rollback` Per l'esecuzione non riuscita che ha attivato il rollback automatico, viene visualizzato l'ID di esecuzione non riuscita.

  L'esempio seguente mostra i dati restituiti per una pipeline denominata *`MyFirstPipeline`*che ha avuto tre esecuzioni:

```
{
    "pipelineExecutionSummaries": [
        {
            "pipelineExecutionId": "eb7ebd36-353a-4551-90dc-18ca5EXAMPLE",
            "status": "Succeeded",
            "startTime": "2024-04-16T09:00:28.185000+00:00",
            "lastUpdateTime": "2024-04-16T09:00:29.665000+00:00",
            "sourceRevisions": [
                {
                    "actionName": "Source",
                    "revisionId": "revision_ID",
                    "revisionSummary": "Added README.txt",
                    "revisionUrl": "console-URL"
                }
            ],
            "trigger": {
                "triggerType": "StartPipelineExecution",
                "triggerDetail": "trigger_ARN"
            },
            "executionMode": "SUPERSEDED"
        },
        {
            "pipelineExecutionId": "fcd61d8b-4532-4384-9da1-2aca1EXAMPLE",
            "status": "Succeeded",
            "startTime": "2024-04-16T08:58:56.601000+00:00",
            "lastUpdateTime": "2024-04-16T08:59:04.274000+00:00",
            "sourceRevisions": [
                {
                    "actionName": "Source",
                    "revisionId": "revision_ID",
                    "revisionSummary": "Added README.txt",
                    "revisionUrl": "console_URL"
                }
            ],
            "trigger": {
                "triggerType": "StartPipelineExecution",
                "triggerDetail": "trigger_ARN"
            },
            "executionMode": "SUPERSEDED"
        }
```

Per visualizzare più dettagli relativi all'esecuzione di una pipeline, eseguire il comando get-pipeline-execution, specificando l'ID univoco dell'esecuzione della pipeline. Ad esempio, per visualizzare più dettagli relativi alla prima esecuzione nell'esempio precedente, immettere quanto segue:

```
aws codepipeline get-pipeline-execution --pipeline-name MyFirstPipeline --pipeline-execution-id 7cf7f7cb-3137-539g-j458-d7eu3EXAMPLE
```

Questo comando restituisce informazioni di riepilogo relative a un'esecuzione di una pipeline, inclusi i dettagli sugli artefatti, l'ID di esecuzione della pipeline, nonché il nome, la versione e lo stato della pipeline.

L'esempio seguente mostra i dati restituiti per una pipeline denominata: *MyFirstPipeline*

```
{
    "pipelineExecution": {
        "pipelineExecutionId": "3137f7cb-7cf7-039j-s83l-d7eu3EXAMPLE",
        "pipelineVersion": 2,
        "pipelineName": "MyFirstPipeline",
        "status": "Succeeded",
        "artifactRevisions": [
            {
                "created": 1496380678.648,
                "revisionChangeIdentifier": "1496380258.243",
                "revisionId": "7636d59f3c461cEXAMPLE8417dbc6371",
                "name": "MyApp",
                "revisionSummary": "Updating the application for feature 12-4820"
            }
        ]
    }
}
```

Visualizza lo stato della pipeline con **`get-pipeline-state`** (CLI)

Puoi utilizzare la CLI per visualizzare lo stato della pipeline, della fase e dell'azione.

- Per visualizzare i dettagli relativi allo stato corrente di una pipeline, eseguire il comando get-pipeline-state, specificando il nome univoco della pipeline. Ad esempio, per visualizzare i dettagli sullo stato corrente di una pipeline denominata *MyFirstPipeline*, inserisci quanto segue:

```
aws codepipeline get-pipeline-state --name MyFirstPipeline
```

Questo comando restituisce lo stato corrente di tutte le fasi della pipeline e lo stato delle operazioni in tali fasi.

L'esempio seguente mostra i dati restituiti per una pipeline a tre fasi denominata *MyFirstPipeline*, in cui le prime due fasi e azioni mostrano il successo, la terza mostra l'errore e la transizione tra la seconda e la terza fase è disabilitata:

```
{
    "updated": 1427245911.525,
    "created": 1427245911.525,
    "pipelineVersion": 1,
    "pipelineName": "MyFirstPipeline",
    "stageStates": [
        {
            "actionStates": [
                {
                    "actionName": "Source",
                    "entityUrl": "https://console.aws.amazon.com/s3/home?#",
                    "latestExecution": {
                        "status": "Succeeded",
                        "lastStatusChange": 1427298837.768
                    }
                }
            ],
            "stageName": "Source"
        },
        {
            "actionStates": [
                {
                    "actionName": "Deploy-CodeDeploy-Application",
                    "entityUrl": "https://console.aws.amazon.com/codedeploy/home?
#",
                    "latestExecution": {
                        "status": "Succeeded",
                        "lastStatusChange": 1427298939.456,
                        "externalExecutionUrl": "https://console.aws.amazon.com/?
#",
                        "externalExecutionId": ""c53dbd42-This-Is-An-Example"",
                        "summary": "Deployment Succeeded"
```

```
                    }
                }
            ],
            "inboundTransitionState": {
                "enabled": true
            },
            "stageName": "Staging"
        },
        {
            "actionStates": [
                {
                    "actionName": "Deploy-Second-Deployment",
                    "entityUrl": "https://console.aws.amazon.com/codedeploy/home?
#",
                    "latestExecution": {
                        "status": "Failed",
                        "errorDetails": {
                            "message": "Deployment Group is already deploying
 deployment ...",
                            "code": "JobFailed"
                        },
                        "lastStatusChange": 1427246155.648
                    }
                }
            ],
            "inboundTransitionState": {
                "disabledReason": "Disabled while I investigate the failure",
                "enabled": false,
                "lastChangedAt": 1427246517.847,
                "lastChangedBy": "arn:aws:iam::80398EXAMPLE:user/CodePipelineUser"
            },
            "stageName": "Production"
        }
    ]
}
```

Visualizza lo stato di esecuzione in entrata con **`get-pipeline-state`** (CLI)

È possibile utilizzare la CLI per visualizzare lo stato di esecuzione in entrata. Quando la transizione è abilitata o lo stadio diventa disponibile, un'esecuzione in entrata `InProgress` continua ed entra nella fase. Un'esecuzione in entrata con uno `Stopped` stato non entra nella fase. Lo stato di esecuzione

in entrata cambia `Failed` se la pipeline viene modificata. Quando si modifica una pipeline, tutte le esecuzioni in corso non continuano e lo stato di esecuzione cambia in. `Failed`

- Per visualizzare i dettagli relativi allo stato corrente di una pipeline, eseguire il comando get-pipeline-state, specificando il nome univoco della pipeline. Ad esempio, per visualizzare i dettagli sullo stato corrente di una pipeline denominata *MyFirstPipeline*, inserisci quanto segue:

```
aws codepipeline get-pipeline-state --name MyFirstPipeline
```

Questo comando restituisce lo stato corrente di tutte le fasi della pipeline e lo stato delle operazioni in tali fasi. L'output mostra anche l'ID di esecuzione della pipeline in ogni fase e se esiste un ID di esecuzione in entrata per una fase con una transizione disabilitata.

L'esempio seguente mostra i dati restituiti per una pipeline a due fasi denominata *MyFirstPipeline*, in cui la prima fase mostra una transizione abilitata e un'esecuzione corretta della pipeline, mentre la seconda fase, denominata`Beta`, mostra una transizione disabilitata e un ID di esecuzione in entrata. L'esecuzione in entrata può avere uno `InProgress` stato, o. `Stopped` `FAILED`

```
{
    "pipelineName": "MyFirstPipeline",
    "pipelineVersion": 2,
    "stageStates": [
        {
            "stageName": "Source",
            "inboundTransitionState": {
                "enabled": true
            },
            "actionStates": [
                {
                    "actionName": "SourceAction",
                    "currentRevision": {
                        "revisionId": "PARcnxX_u0SMRBnKh83pHL09.zPRLLMu"
                    },
                    "latestExecution": {
                        "actionExecutionId": "14c8b311-0e34-4bda-EXAMPLE",
                        "status": "Succeeded",
                        "summary": "Amazon S3 version id: PARcnxX_u0EXAMPLE",
                        "lastStatusChange": 1586273484.137,
                        "externalExecutionId": "PARcnxX_u0EXAMPLE"
                    },
```

```
                    "entityUrl": "https://console.aws.amazon.com/s3/home?#"
                }
            ],
            "latestExecution": {
                "pipelineExecutionId": "27a47e06-6644-42aa-EXAMPLE",
                "status": "Succeeded"
            }
        },
        {
            "stageName": "Beta",
            "inboundExecution": {
                "pipelineExecutionId": "27a47e06-6644-42aa-958a-EXAMPLE",
                "status": "InProgress"
            },
            "inboundTransitionState": {
                "enabled": false,
                "lastChangedBy": "USER_ARN",
                "lastChangedAt": 1586273583.949,
                "disabledReason": "disabled"
            },
                    "currentRevision": {
            "actionStates": [
                {
                    "actionName": "BetaAction",
                    "latestExecution": {
                        "actionExecutionId": "a748f4bf-0b52-4024-98cf-EXAMPLE",
                        "status": "Succeeded",
                        "summary": "Deployment Succeeded",
                        "lastStatusChange": 1586272707.343,
                        "externalExecutionId": "d-KFGF3EXAMPLE",
                        "externalExecutionUrl": "https://us-
west-2.console.aws.amazon.com/codedeploy/home?#/deployments/d-KFGF3WTS2"
                    },
                    "entityUrl": "https://us-west-2.console.aws.amazon.com/
codedeploy/home?#/applications/my-application"
                }
            ],
            "latestExecution": {
                "pipelineExecutionId": "f6bf1671-d706-4b28-EXAMPLE",
                "status": "Succeeded"
            }
        }
    ],
    "created": 1585622700.512,
```

```
    "updated": 1586273472.662
}
```

Visualizza lo stato e le revisioni dell'origine con **`get-pipeline-execution`** (CLI)

Puoi visualizzare i dettagli relativi agli artefatti di origine (artefatti di output originati nella prima fase di una pipeline) che sono utilizzati nell'esecuzione di una pipeline. I dettagli includono identificatori, ad esempio gli ID di commit, i commenti di check-in, il tempo trascorso dalla creazione o dall'aggiornamento dell'artefatto e, quando si utilizza l'interfaccia a riga di comando, i numeri di versione di operazioni di compilazione. Per alcuni tipi di revisione, puoi visualizzare e aprire l'URL del commit per la versione artefatto. Le revisioni di origine sono composte dai seguenti elementi:

- Summary: informazioni di riepilogo sulla versione più recente dell'artefatto. Per GitHub e AWS CodeCommit repository, il messaggio di commit. Per i bucket o le azioni Amazon S3, il contenuto fornito dall'utente di una codepipeline-artifact-revision-summary chiave specificata nei metadati dell'oggetto.
- revisionUrl: l'ID commit per la revisione dell'artefatto. Per gli artefatti archiviati nei GitHub nostri AWS CodeCommit repository, l'ID di commit è collegato a una pagina dei dettagli del commit.

Puoi eseguire il comando get-pipeline-execution per visualizzare le informazioni relative alle revisioni di origine più recenti che erano incluse nell'esecuzione di una pipeline. Dopo la prima esecuzione del comando get-pipeline-state per ottenere informazioni dettagliate su tutte le fasi in una pipeline, identifica l'ID di esecuzione che si applica a una fase per la quale sono richiesti i dettagli di revisione di origine. Quindi utilizzi l'ID di esecuzione nel comando get-pipeline-execution. (Dato che le fasi in una pipeline potrebbero essere state completate durante esecuzioni differenti della pipeline, potrebbero avere ID di esecuzione differenti)

In altre parole, se desideri visualizzare i dettagli relativi agli artefatti attualmente in fase Gestione temporanea, esegui il comando get-pipeline-state, identifica l'ID di esecuzione corrente della fase Gestione temporanea e quindi esegui il comando get-pipeline-execution utilizzando tale ID di esecuzione.

Per visualizzare le revisioni dello stato e dell'origine in una pipeline

1. Apri un terminale (Linux, macOS o Unix) o un prompt dei comandi (Windows) e usali AWS CLI per eseguire il comando. get-pipeline-state Per una pipeline denominata *`MyFirstPipeline`*, devi inserire:

```
aws codepipeline get-pipeline-state --name MyFirstPipeline
```

Questo comando restituisce lo stato più recente di una pipeline, incluso l'ultimo ID di esecuzione della pipeline per ogni fase.

2. Per visualizzare i dettagli relativi all'esecuzione di una pipeline, esegui il comando get-pipeline-execution, specificando il nome univoco della pipeline e l'ID di esecuzione della pipeline dell'esecuzione per la quale si desidera visualizzare i dettagli dell'artefatto. Ad esempio, per visualizzare i dettagli sull'esecuzione di una pipeline denominata, con l'ID di esecuzione *MyFirstPipeline*3137F7CB-7CF7-039J-S83L-D7EU3Example, è necessario immettere quanto segue:

```
aws codepipeline get-pipeline-execution --pipeline-name MyFirstPipeline --pipeline-
execution-id 3137f7cb-7cf7-039j-s83l-d7eu3EXAMPLE
```

Questo comando restituisce le informazioni relative a ogni revisione di origine che fa parte dell'esecuzione della pipeline e che identifica le informazioni sulla pipeline. Vengono incluse solo le informazioni sulle fasi della pipeline che sono state incluse in tale esecuzione. È possibile che la pipeline contenga altre fasi che non facevano parte di tale esecuzione della pipeline.

L'esempio seguente mostra i dati restituiti per una porzione di pipeline denominata, in cui un artefatto denominato "" è archiviato in un repository: *MyFirstPipeline*MyApp GitHub

3.

```
{
    "pipelineExecution": {
        "artifactRevisions": [
            {
                "created": 1427298837.7689769,
                "name": "MyApp",
                "revisionChangeIdentifier": "1427298921.3976923",
                "revisionId": "7636d59f3c461cEXAMPLE8417dbc6371",
                "revisionSummary": "Updating the application for feature 12-4820",
                "revisionUrl": "https://api.github.com/repos/anycompany/MyApp/git/
commits/7636d59f3c461cEXAMPLE8417dbc6371"
            }
        ],
        "pipelineExecutionId": "3137f7cb-7cf7-039j-s83l-d7eu3EXAMPLE",
        "pipelineName": "MyFirstPipeline",
        "pipelineVersion": 2,
        "status": "Succeeded",
```

```
        "executionMode": "SUPERSEDED",
        "executionType": "ROLLBACK",
        "rollbackMetadata": {
            "rollbackTargetPipelineExecutionId": "4f47bed9-6998-476c-a49d-
e60beEXAMPLE"
         }
    }
}
```

Visualizza le esecuzioni delle azioni con **`list-action-executions`** (CLI)

Puoi visualizzare i dettagli di esecuzione di un'operazione per una pipeline, ad esempio l'ID di esecuzione dell'operazione, gli artefatti di input, gli artefatti di output, i risultati dell'esecuzione e lo stato. Puoi fornire il filtro ID di esecuzione per ottenere un elenco di operazioni nell'esecuzione di una pipeline:

> **Note**
>
> La cronologia dettagliata delle esecuzioni è disponibile per le esecuzioni a partire dal 21 febbraio 2019 in poi.

- Per visualizzare le esecuzioni di operazioni per una pipeline, esegui una delle seguenti operazioni:
  - Per visualizzare i dettagli relativi a tutte le esecuzioni di operazioni in una pipeline, esegui il comando list-action-executions, specificando il nome univoco della pipeline. Ad esempio, per visualizzare le esecuzioni di azioni in una pipeline denominata *`MyFirstPipeline`*, inserisci quanto segue:

    ```
    aws codepipeline list-action-executions --pipeline-name MyFirstPipeline
    ```

    L'esempio seguente mostra una porzione di output di esempio per questo comando:

    ```
    {
        "actionExecutionDetails": [
            {
                "actionExecutionId": "ID",
                "lastUpdateTime": 1552958312.034,
    ```

```
            "startTime": 1552958246.542,
            "pipelineExecutionId": "Execution_ID",
            "actionName": "Build",
            "status": "Failed",
            "output": {
                "executionResult": {
                    "externalExecutionUrl": "Project_ID",
                    "externalExecutionSummary": "Build terminated with state:
 FAILED",
                    "externalExecutionId": "ID"
                },
                "outputArtifacts": []
            },
            "stageName": "Beta",
            "pipelineVersion": 8,
            "input": {
                "configuration": {
                    "ProjectName": "java-project"
                },
                "region": "us-east-1",
                "inputArtifacts": [
                    {
                        "s3location": {
                            "bucket": "codepipeline-us-east-1-ID",
                            "key": "MyFirstPipeline/MyApp/Object.zip"
                        },
                        "name": "MyApp"
                    }
                ],
                "actionTypeId": {
                    "version": "1",
                    "category": "Build",
                    "owner": "AWS",
                    "provider": "CodeBuild"
                }
            }
        },

. . .
```

- Per visualizzare tutte le esecuzioni di operazioni in un'esecuzione di pipeline, esegui il comando list-action-executions, specificando il nome univoco della pipeline e l'ID

di esecuzione. Ad esempio, per visualizzare le esecuzioni di operazioni relative a un *Execution_ID*, immetti quanto segue:

```
aws codepipeline list-action-executions --pipeline-name MyFirstPipeline --filter
 pipelineExecutionId=Execution_ID
```

- L'esempio seguente mostra una porzione di output di esempio per questo comando:

```
{
    "actionExecutionDetails": [
        {
            "stageName": "Beta",
            "pipelineVersion": 8,
            "actionName": "Build",
            "status": "Failed",
            "lastUpdateTime": 1552958312.034,
            "input": {
                "configuration": {
                    "ProjectName": "java-project"
                },
                "region": "us-east-1",
                "actionTypeId": {
                    "owner": "AWS",
                    "category": "Build",
                    "provider": "CodeBuild",
                    "version": "1"
                },
                "inputArtifacts": [
                    {
                        "s3location": {
                            "bucket": "codepipeline-us-east-1-ID",
                            "key": "MyFirstPipeline/MyApp/Object.zip"
                        },
                        "name": "MyApp"
                    }
                ]
            },


. . .
```

Impostare o modificare la modalità di esecuzione della pipeline

È possibile impostare la modalità di esecuzione per la pipeline per specificare come vengono gestite più esecuzioni.

Per ulteriori informazioni sulle modalità di esecuzione della pipeline, vedere. Funzionamento delle esecuzioni pipeline

> **Important**
>
> Per le pipeline in modalità PARALLEL, quando si modifica la modalità di esecuzione della pipeline su QUEUED o SUPERSEDED, lo stato della pipeline non visualizzerà lo stato aggiornato come PARALLEL. Per ulteriori informazioni, consulta Le pipeline modificate dalla modalità PARALLEL mostreranno una modalità di esecuzione precedente.

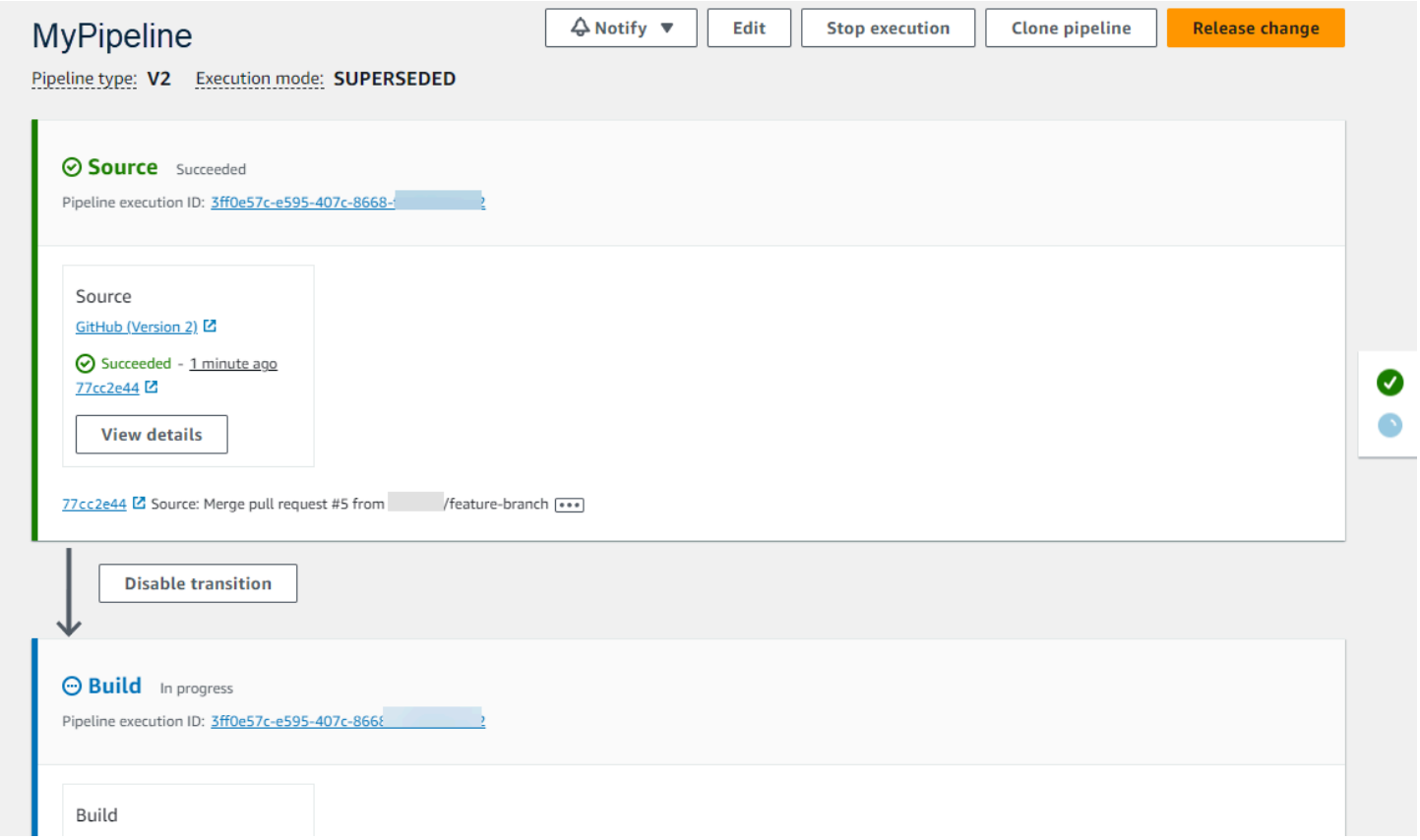
> **Important**
>
> Per le tubazioni in modalità PARALLEL, quando si modifica la modalità di esecuzione della pipeline su QUEUED o SUPERSEDED, la definizione della pipeline per la pipeline in ciascuna modalità non verrà aggiornata. Per ulteriori informazioni, consulta Le tubazioni in modalità PARALLEL hanno una definizione di pipeline obsoleta se modificate quando si passa alla modalità QUEUED o SUPERSEDED.

Considerazioni sulla visualizzazione delle modalità di esecuzione

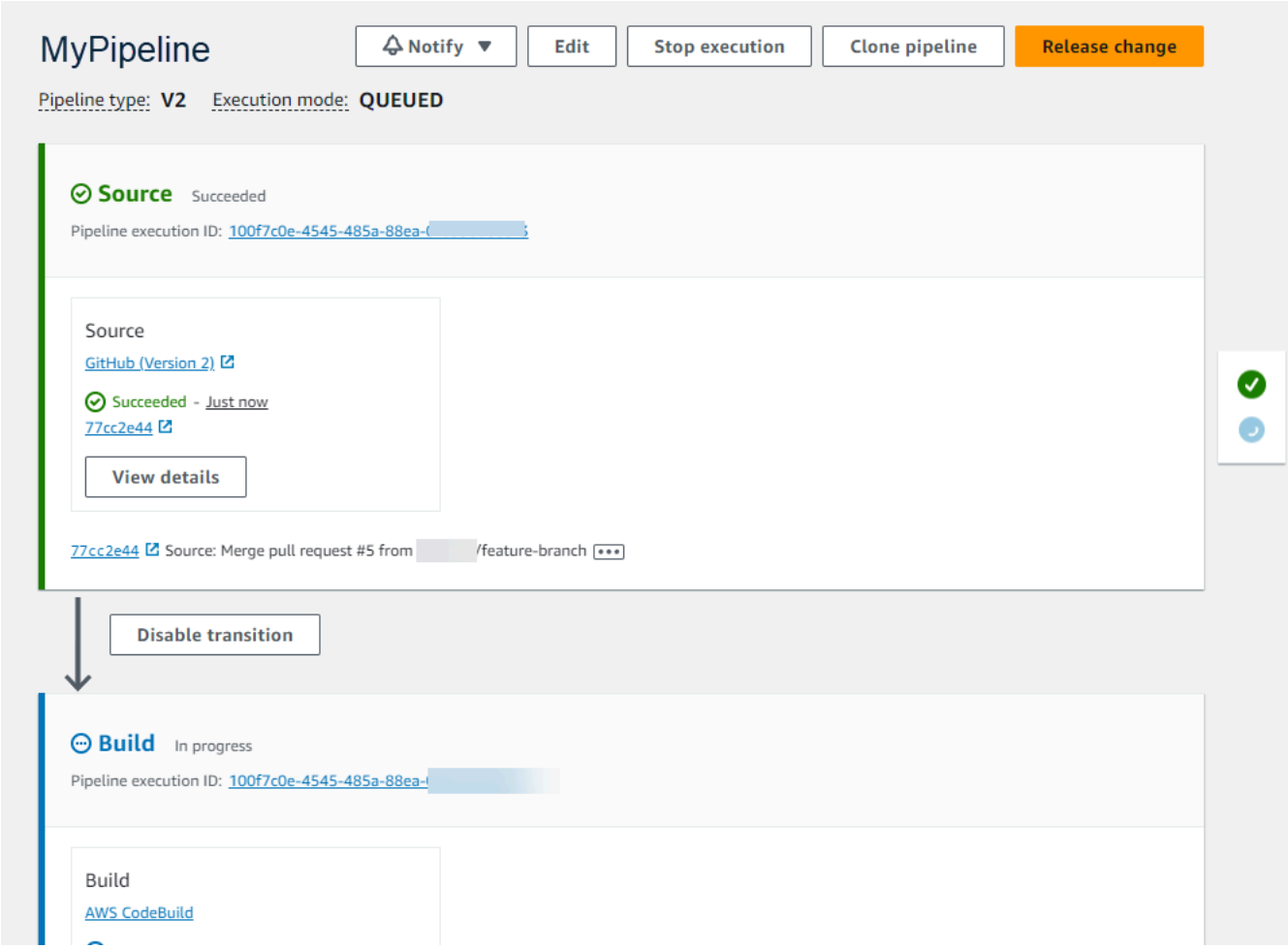
Esistono considerazioni sulla visualizzazione delle pipeline in modalità di esecuzione specifiche.

Per le modalità SUPERSEDED e QUEUED, utilizzate la visualizzazione pipeline per vedere le esecuzioni in corso e fate clic sull'ID di esecuzione per visualizzare i dettagli e la cronologia. Per la modalità PARALLEL, fate clic sull'ID di esecuzione per visualizzare l'esecuzione in corso nella scheda Visualizzazione.

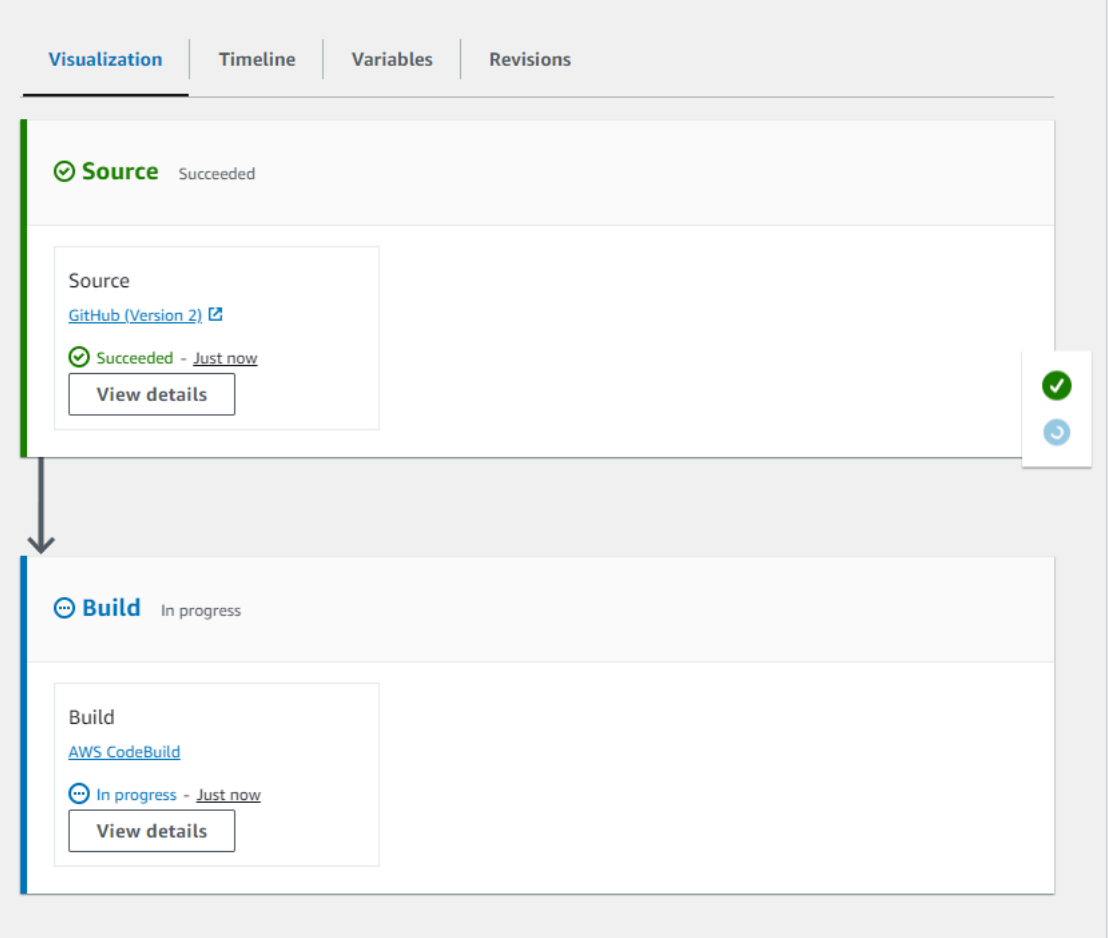
Di seguito viene illustrata la visualizzazione per la modalità SUPERSEDED in. CodePipeline



Di seguito viene illustrata la visualizzazione per la modalità QUEUED in. CodePipeline



Di seguito viene mostrata la visualizzazione per la modalità PARALLEL in CodePipeline.



Considerazioni sul passaggio da una modalità di esecuzione all'altra

Di seguito sono riportate le considerazioni relative alle tubazioni relative alla modifica della modalità della pipeline. Quando si passa da una modalità di esecuzione all'altra in modalità Modifica e si salva la modifica, alcune viste o stati potrebbero modificarsi.

Ad esempio, quando si passa dalla modalità PARALLEL a una modalità QUEUED o SUPERSEDED, l'esecuzione avviata in modalità PARALLEL continuerà a essere eseguita. Questi possono essere visualizzati nella pagina della cronologia delle esecuzioni. La visualizzazione della pipeline mostrerà l'esecuzione eseguita in precedenza in modalità QUEUED o SUPERSEDED o in caso contrario uno stato vuoto.

Come altro esempio, quando si passa dalla modalità QUEUED o SUPERSEDED alla modalità PARALLEL, non verrà più visualizzata la pagina di visualizzazione/stato della pipeline. Per visualizzare un'esecuzione in modalità PARALLEL, utilizzate la scheda di visualizzazione nella

pagina dei dettagli dell'esecuzione. Le esecuzioni avviate in modalità SOSTITUITA o IN CODA verranno annullate.

La tabella seguente fornisce ulteriori dettagli.

Cambio di modalità	Dettagli sull'esecuzione in sospeso e attiva	Dettagli sullo stato della pipeline
SOSTITUITO a SOSTITUITO / SOSTITUITO a CODA	• Le esecuzioni attive vengono annullate dopo il completamento delle azioni in corso. • Le esecuzioni in sospeso vengono annullate.	Lo stato della pipeline, ad esempio annullato, viene mantenuto tra la versione della prima modalità e la seconda modalità.
QUEUED a QUEUED/QU EUED a SOSTITUSEDED	• Le esecuzioni attive vengono annullate al completamento delle azioni in corso. • Le esecuzioni in sospeso vengono annullate.	Lo stato della pipeline, ad esempio annullato, viene mantenuto tra la versione della prima modalità e la seconda modalità.
PARALLEL a PARALLEL	Tutte le esecuzioni possono essere eseguite indipende ntemente dagli aggiornamenti delle definizioni della pipeline.	Vuoto. La modalità parallela non ha uno stato di pipeline.
SOSTITUITO in PARALLELO/ IN CODA in PARALLELO	• Le esecuzioni attive vengono annullate al completamento delle azioni in corso. • Le esecuzioni in sospeso vengono annullate.	Vuoto. La modalità parallela non ha uno stato di pipeline.

Imposta o modifica la modalità di esecuzione della pipeline (console)

È possibile utilizzare la console per impostare la modalità di esecuzione della pipeline.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi e lo stato di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegliere il nome della pipeline da modificare.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. Nella pagina Modifica, scegliete Modifica: proprietà della tubazione.
5. Scegliete la modalità per la vostra pipeline.

   - Sostituito
   - In coda (è richiesto il tipo di tubazione V2)
   - Parallela (è richiesto il tipo di tubazione V2)
6. Nella pagina Modifica, scegli Fine.

Imposta la modalità di esecuzione della pipeline (CLI)

Per utilizzare il AWS CLI per impostare la modalità di esecuzione della pipeline, utilizzate il `create-pipeline` comando or. `update-pipeline`

1. Apri una sessione terminale (Linux, macOS o Unix) o un prompt dei comandi (Windows) ed esegui il get-pipeline comando per copiare la struttura della pipeline in un file JSON. Ad esempio, per una pipeline denominata **`MyFirstPipeline`**, immettere il seguente comando:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   ```

   Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.
2. Apri il file JSON in qualsiasi editor di testo semplice e modifica la struttura del file in modo che rifletta la modalità di esecuzione della pipeline che desideri impostare, ad esempio QUEUED.

   ```
   "executionMode": "QUEUED"
   ```

   L'esempio seguente mostra come impostare la modalità di esecuzione su QUEUED in una pipeline di esempio con due fasi.

   ```
   {
   ```

```
    "pipeline": {
        "name": "MyPipeline",
        "roleArn": "arn:aws:iam::111122223333:role/service-role/
AWSCodePipelineServiceRole-us-east-1-dkpippe",
        "artifactStore": {
            "type": "S3",
            "location": "bucket"
        },
        "stages": [
            {
                "name": "Source",
                "actions": [
                    {
                        "name": "Source",
                        "actionTypeId": {
                            "category": "Source",
                            "owner": "AWS",
                            "provider": "CodeCommit",
                            "version": "1"
                        },
                        "runOrder": 1,
                        "configuration": {
                            "BranchName": "main",
                            "OutputArtifactFormat": "CODE_ZIP",
                            "PollForSourceChanges": "true",
                            "RepositoryName": "MyDemoRepo"
                        },
                        "outputArtifacts": [
                            {
                                "name": "SourceArtifact"
                            }
                        ],
                        "inputArtifacts": [],
                        "region": "us-east-1",
                        "namespace": "SourceVariables"
                    }
                ]
            },
            {
                "name": "Build",
                "actions": [
                    {
                        "name": "Build",
                        "actionTypeId": {
```

```
                            "category": "Build",
                            "owner": "AWS",
                            "provider": "CodeBuild",
                            "version": "1"
                        },
                        "runOrder": 1,
                        "configuration": {
                            "ProjectName": "MyBuildProject"
                        },
                        "outputArtifacts": [
                            {
                                "name": "BuildArtifact"
                            }
                        ],
                        "inputArtifacts": [
                            {
                                "name": "SourceArtifact"
                            }
                        ],
                        "region": "us-east-1",
                        "namespace": "BuildVariables"
                    }
                ]
            }
        ],
        "version": 1,
        "executionMode": "QUEUED"
    }
}
```

3. Se stai utilizzando la struttura della pipeline recuperata utilizzando il comando get-pipeline, questa deve essere modificata nel file JSON. Rimuovi le righe `metadata` dal file per consentire al comando update-pipeline di utilizzarlo. Rimuovere la sezione dalla struttura della pipeline nel file JSON (le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`)

   Ad esempio, rimuovere dalla struttura le seguenti righe:

```
"metadata": {
  "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
  "created": "date",
  "updated": "date"
  }
```

Salvare il file.

4. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline:

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

```
aws codepipeline update-pipeline --cli-input-json file://pipeline.json
```

Questo comando restituisce l'intera struttura della pipeline modificata.

> **Note**
>
> Il comando update-pipeline arresta la pipeline. Se è in corso di elaborazione una versione durante l'esecuzione del comando update-pipeline, tale elaborazione viene arrestata. Per eseguire tale versione nella pipeline aggiornata, avviare la pipeline manualmente.

Riprovare una fase fallita o azioni non riuscite in una fase

È possibile riprovare una fase che ha avuto esito negativo senza dover eseguire nuovamente una pipeline dall'inizio. A tale scopo, riprovate le azioni non riuscite in una fase oppure riprovate tutte le azioni nella fase a partire dalla prima azione nella fase. Quando riprovi a eseguire le azioni non riuscite in una fase, tutte le azioni ancora in corso continuano a funzionare e le azioni fallite vengono nuovamente attivate. Quando riprovi una fase fallita dalla prima azione nella fase, non è possibile che nella fase siano in corso azioni. Prima di poter ritentare una fase, è necessario che tutte le azioni siano fallite oppure che alcune azioni siano fallite e altre abbiano avuto successo.

> **Important**
>
> Riprovare una fase non riuscita riprova tutte le azioni della fase dalla prima azione nella fase e riprovare le azioni fallite riprova tutte le azioni fallite nella fase. Ciò sovrascrive gli artefatti di output delle azioni precedentemente eseguite con successo nella stessa esecuzione. Sebbene gli artefatti possano essere sostituiti, la cronologia di esecuzione delle azioni precedentemente riuscite viene comunque conservata.

Se si utilizza la console per visualizzare una pipeline, sullo stage viene visualizzato il pulsante Riprova fase o Riprova azioni fallite che può essere riprovato.

Se si utilizza la AWS CLI, è possibile utilizzare il get-pipeline-state comando per determinare se alcune azioni sono fallite.

> **Note**
>
> Nei seguenti casi, potresti non essere in grado di riprovare una fase:
>
> - Tutte le azioni nella fase hanno avuto esito positivo, pertanto lo stato della fase non è fallito.
> - La struttura complessiva della pipeline è cambiata dopo il fallimento della fase.
> - È già in corso un altro tentativo di ripetizione nella fase.

Argomenti

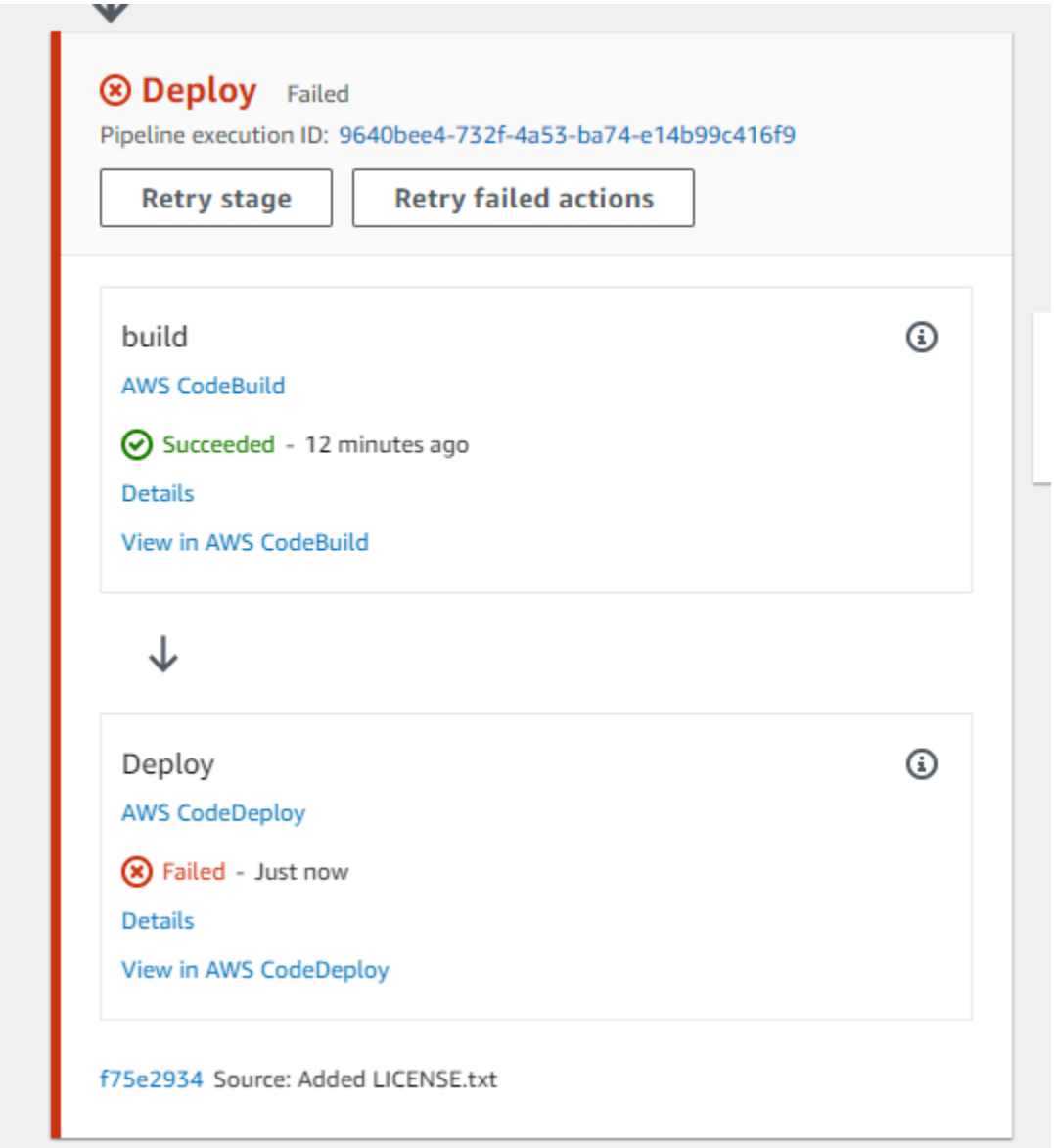
Riprova una fase fallita (console)

Per riprovare una fase fallita o azioni non riuscite in una fase: console

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.

2. In Name (Nome), scegli il nome della pipeline.
3. Individuate la fase in cui si è verificata l'azione non riuscita, quindi scegliete una delle seguenti opzioni:
   - Per riprovare tutte le azioni nella fase, scegli Riprova fase.
   - Per riprovare solo le azioni fallite nella fase, scegli Riprova le azioni fallite.



Se tutte le operazioni ripetute nella fase vengono completate, l'esecuzione della pipeline continua.

Riprova una fase fallita (CLI)

Per riprovare una fase fallita o azioni non riuscite in una fase - CLI

Per utilizzare il comando AWS CLI per riprovare tutte le azioni o tutte le azioni non riuscite, esegui il retry-stage-execution comando con i seguenti parametri:

```
--pipeline-name <value>
--stage-name <value>
--pipeline-execution-id <value>
--retry-mode ALL_ACTIONS/FAILED_ACTIONS
```

> **Note**
>
> I valori per cui è possibile utilizzare `retry-mode` sono `FAILED_ACTIONS` e`ALL_ACTIONS`.

1. In un terminale (Linux, macOS o Unix) o nel prompt dei comandi (Windows), esegui il retry-stage-executioncomando, come illustrato nell'esempio seguente per una pipeline denominata. `MyPipeline`

   ```
   aws codepipeline retry-stage-execution --pipeline-name MyPipeline --stage-name
    Deploy --pipeline-execution-id b59babff-5f34-EXAMPLE --retry-mode FAILED_ACTIONS
   ```

   L'output restituisce l'ID di esecuzione:

   ```
   {
       "pipelineExecutionId": "b59babff-5f34-EXAMPLE"
   }
   ```

2. Puoi anche eseguire il comando con un file di input JSON. Crea innanzitutto un file JSON che identifichi la pipeline, la fase contenente le operazioni non riuscite e l'esecuzione della pipeline più recente in tale fase. Quindi, esegui il comando retry-stage-execution con il parametro `--cli-input-json`. Per recuperare i dettagli necessari per il file JSON, è più semplice utilizzare il comando get-pipeline-state.

   a. In un terminale (Linux, macOS o Unix) o nel prompt dei comandi (Windows), esegui il get-pipeline-statecomando su una pipeline. Ad esempio, per una pipeline denominata MyFirstPipeline, è necessario digitare qualcosa di simile al seguente:

```
aws codepipeline get-pipeline-state --name MyFirstPipeline
```

La risposta al comando include le informazioni sullo stato della pipeline per ogni fase. Nel seguente esempio, la risposta indica che una o più operazioni non sono riuscite nella fase Gestione temporanea:

```
{
    "updated": 1427245911.525,
    "created": 1427245911.525,
    "pipelineVersion": 1,
    "pipelineName": "MyFirstPipeline",
    "stageStates": [
        {
            "actionStates": [...],
            "stageName": "Source",
            "latestExecution": {
                "pipelineExecutionId": "9811f7cb-7cf7-SUCCESS",
                "status": "Succeeded"
            }
        },
        {
            "actionStates": [...],
            "stageName": "Staging",
            "latestExecution": {
                "pipelineExecutionId": "3137f7cb-7cf7-EXAMPLE",
                "status": "Failed"
            }
        }
    ]
}
```

b. In un editor di testo normale, creare un file in formato JSON in cui verranno registrate le informazioni riportate di seguito:

- Il nome della pipeline contenente le operazioni non riuscite.
- Il nome della fase contenente le operazioni non riuscite.
- L'ID dell'ultima esecuzione della pipeline nella fase
- La modalità di ripetizione.

MyFirstPipeline Nell'esempio precedente, il file avrebbe un aspetto simile al seguente:

```
{
    "pipelineName": "MyFirstPipeline",
    "stageName": "Staging",
    "pipelineExecutionId": "3137f7cb-7cf7-EXAMPLE",
    "retryMode": "FAILED_ACTIONS"
}
```

c. Salvare il file con un nome come **`retry-failed-actions.json`**.

d. Chiama il file creato quando esegui il comando retry-stage-execution. Per esempio:

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

```
aws codepipeline retry-stage-execution --cli-input-json file://retry-failed-
actions.json
```

e. Per visualizzare i risultati del nuovo tentativo, apri la CodePipeline console e scegli la pipeline che contiene le azioni fallite oppure usa nuovamente il get-pipeline-state comando. Per ulteriori informazioni, consulta Visualizza le pipeline e i dettagli in CodePipeline.

Configurazione del rollback dello stage

È possibile ripristinare una fase e riportarla a un'esecuzione che in quella fase ha avuto esito positivo. È possibile preconfigurare una fase per il rollback in caso di errore oppure ripristinare manualmente una fase. L'operazione di rollback comporterà una nuova esecuzione. L'esecuzione della pipeline di destinazione scelta per il rollback viene utilizzata per recuperare le revisioni e le variabili di origine.

Il tipo di esecuzione, standard o rollback, viene visualizzato nella cronologia della pipeline, nello stato della pipeline e nei dettagli di esecuzione della pipeline.

Argomenti

- Considerazioni sui rollback

- Eseguite il rollback di uno stage manualmente
- Configura una fase per il rollback automatico
- Visualizza lo stato di rollback nell'elenco delle esecuzioni
- Visualizza i dettagli sullo stato del rollback

Considerazioni sui rollback

Le considerazioni relative al rollback dello stage sono le seguenti:

- Non è possibile ripristinare uno stadio di origine.
- La pipeline può tornare a un'esecuzione precedente solo se l'esecuzione precedente è stata avviata nella versione corrente della struttura della pipeline.
- Non è possibile ripristinare un ID di esecuzione di destinazione che sia un tipo di esecuzione di rollback.

Eseguite il rollback di uno stage manualmente

È possibile ripristinare manualmente uno stage utilizzando la console o la CLI. La pipeline può tornare a un'esecuzione precedente solo se l'esecuzione precedente è stata avviata nella versione corrente della struttura della pipeline.

È inoltre possibile configurare una fase per il rollback automatico in caso di errore, come descritto in. Configura una fase per il rollback automatico

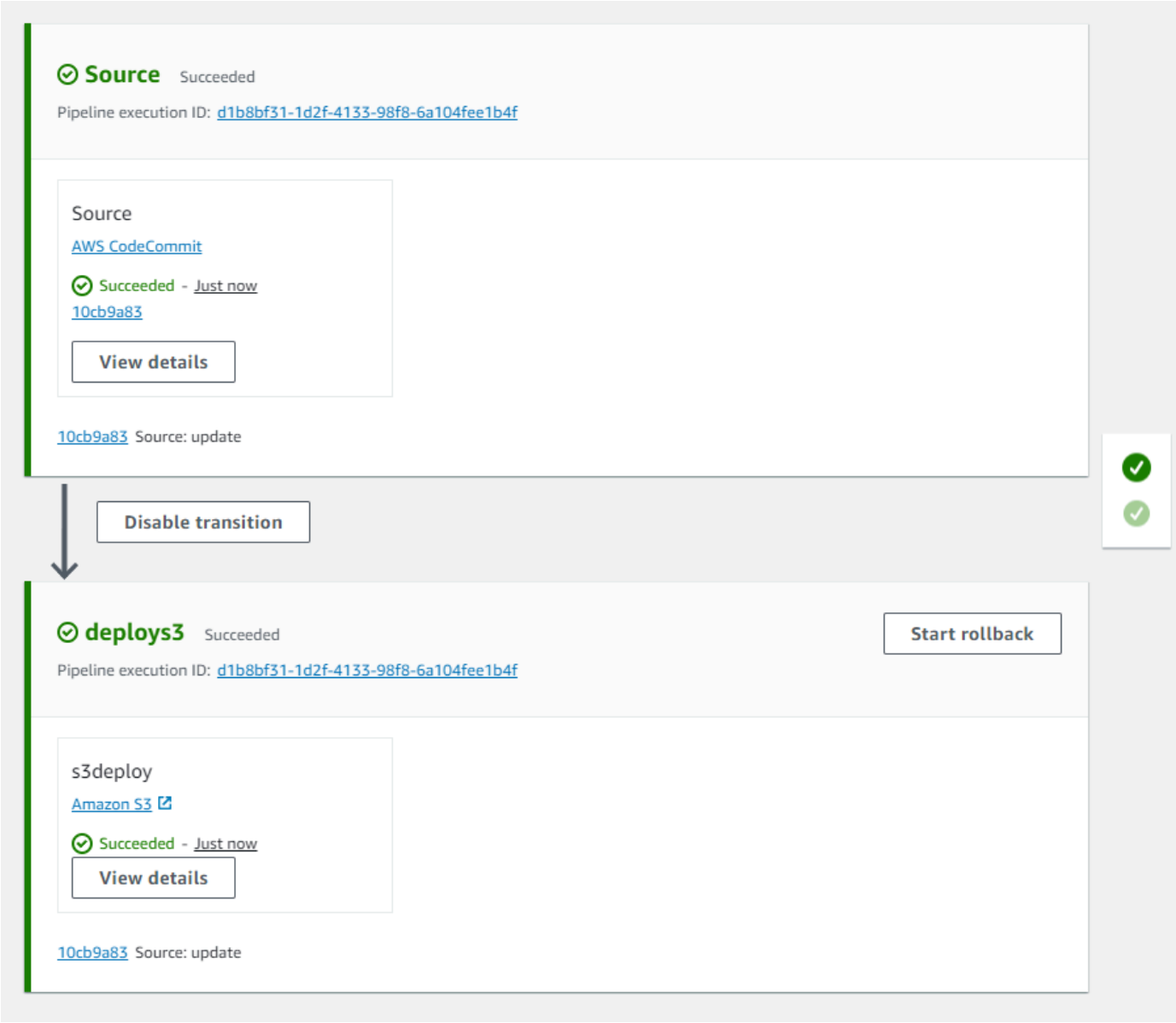
Ripristina manualmente uno stage (console)

È possibile utilizzare la console per ripristinare manualmente uno stage fino all'esecuzione della pipeline di destinazione. Quando si esegue il rollback di una fase, viene visualizzata un'etichetta Rollback sulla visualizzazione della pipeline nella console.

Ripristina manualmente uno stage (console)

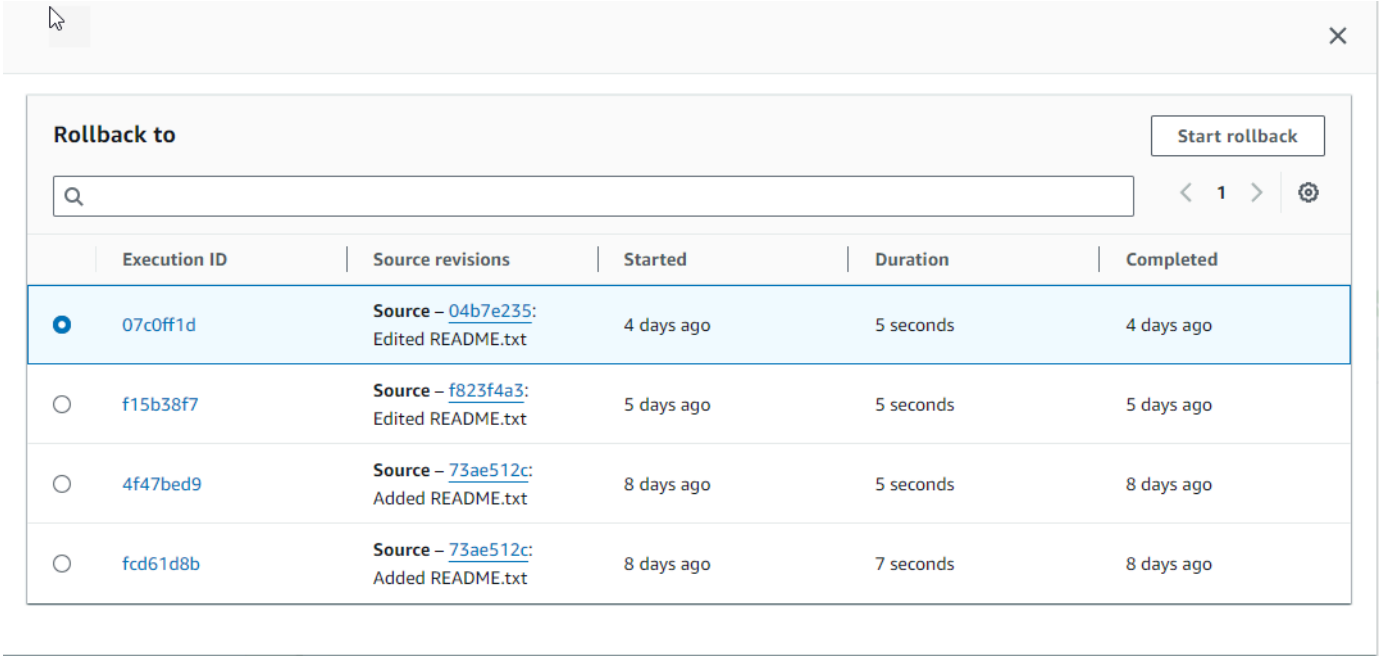
1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi e lo stato di tutte le pipeline associate al tuo AWS account.
2. In Nome, scegliete il nome della pipeline con lo stage da ripristinare.

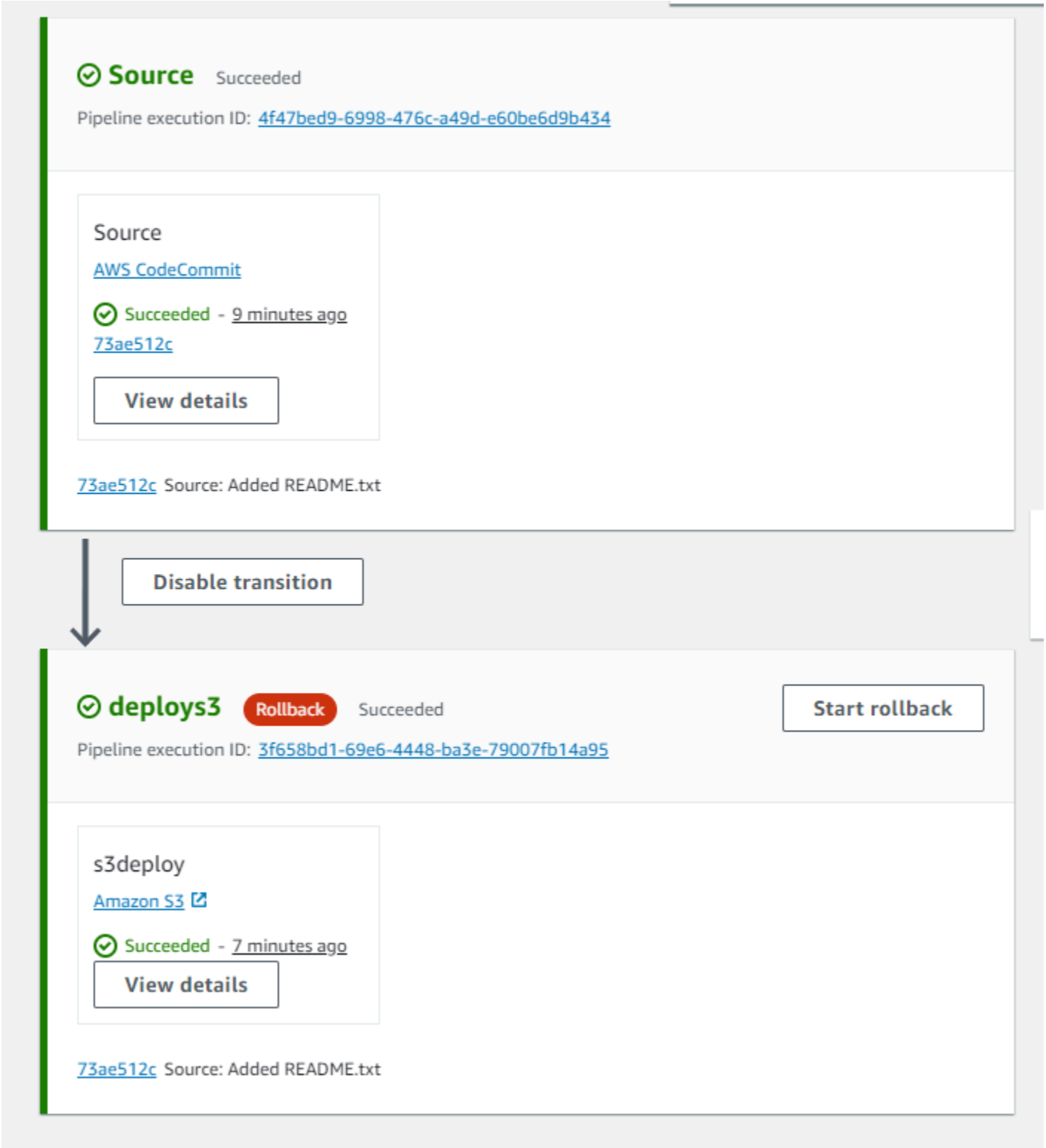


3. Sul palco, scegli Avvia rollback. Viene visualizzato il pulsante Ripristina alla pagina.
4. Scegliete l'esecuzione di destinazione a cui desiderate ripristinare lo stage.

> **Note**
>
> L'elenco delle esecuzioni della pipeline di destinazione disponibili includerà tutte le esecuzioni nell'attuale versione della pipeline a partire dal 1° febbraio 2024.



Il diagramma seguente mostra un esempio di rollback stage con il nuovo ID di esecuzione.



Ripristina manualmente uno stage (CLI)

Per utilizzare il AWS CLI ripristino manuale di uno stage, utilizzate il `rollback-stage` comando.

È anche possibile ripristinare uno stadio manualmente, come descritto in[Eseguite il rollback di uno stage manualmente](#).

> **Note**
>
> L'elenco delle esecuzioni della pipeline di destinazione disponibili includerà tutte le esecuzioni nell'attuale versione della pipeline a partire dal 1° febbraio 2024.

Per ripristinare manualmente uno stage (CLI)

1. Il comando CLI per il rollback manuale richiederà l'ID di esecuzione di un'esecuzione della pipeline precedentemente riuscita nella fase. Per ottenere l'ID di esecuzione della pipeline di destinazione da specificare, utilizzate il list-pipeline-executions comando con un filtro che restituirà le esecuzioni riuscite nella fase. Apri un terminale (Linux, macOS o Unix) o il prompt dei comandi (Windows) e usa AWS CLI per eseguire il `list-pipeline-executions` comando, specificando il nome della pipeline e il filtro per le esecuzioni riuscite nella fase. In questo esempio, l'output elencherà le esecuzioni della pipeline per la pipeline denominata e per le esecuzioni riuscite nella fase denominata MyFirstPipeline . `deploys3`

   ```
   aws codepipeline list-pipeline-executions --pipeline-name MyFirstPipeline --filter
    succeededInStage={stageName=deploys3}
   ```

   Nell'output, copiate l'ID di esecuzione dell'esecuzione precedentemente riuscita che desiderate specificare per il rollback. Lo utilizzerai nel passaggio successivo come ID di esecuzione di destinazione.

2. Apri un terminale (Linux, macOS o Unix) o il prompt dei comandi (Windows) e usa AWS CLI per eseguire il `rollback-stage` comando, specificando il nome della pipeline, il nome dello stage e l'esecuzione di destinazione a cui desideri eseguire il rollback. Ad esempio, per ripristinare una fase denominata Deploy per una pipeline denominata: *`MyFirstPipeline`*

   ```
   aws codepipeline rollback-stage --pipeline-name MyFirstPipeline --stage-name Deploy
    --target-pipeline-execution-id bc022580-4193-491b-8923-9728dEXAMPLE
   ```

   L'output restituisce l'ID di esecuzione per la nuova esecuzione ripristinata. Si tratta di un ID separato che utilizza le revisioni e i parametri di origine dell'esecuzione di destinazione specificata.

Configura una fase per il rollback automatico

È possibile configurare le fasi di una pipeline per il rollback automatico in caso di errore. Quando la fase fallisce, viene ripristinata l'ultima esecuzione riuscita. La pipeline può tornare a un'esecuzione precedente solo se l'esecuzione precedente è stata avviata nella versione corrente della struttura della pipeline. Poiché la configurazione del rollback automatico fa parte della definizione della pipeline, la fase della pipeline eseguirà il rollback automatico solo dopo l'esecuzione corretta della pipeline nella fase di pipeline.

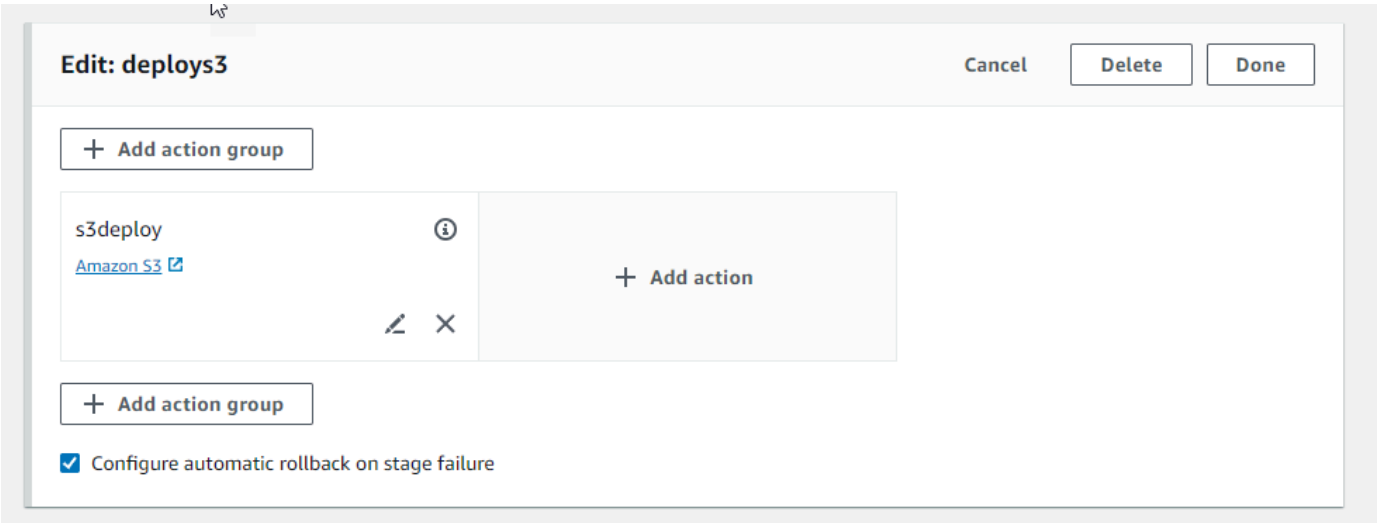
Configura una fase per il rollback automatico (console)

È possibile ripristinare una fase fino a un'esecuzione precedente specificata con successo. Per ulteriori informazioni, RollbackStageconsulta la Guida alle CodePipeline API.

Configura una fase per il rollback automatico (console)

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi e lo stato di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegliere il nome della pipeline da modificare.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. Nella pagina Modifica, per l'azione che desideri modificare, scegli Modifica fase.
5. Scegli Configura il rollback automatico in caso di errore sullo stage. Salva le modifiche alla tua pipeline.



Configurare una fase per il rollback automatico (CLI)

AWS CLI Per configurare una fase fallita in modo da ripristinare automaticamente l'esecuzione riuscita più recente, utilizzate i comandi per creare o aggiornare una pipeline come descritto in and. Creare una pipeline in CodePipeline Modificare una tubazione in CodePipeline

- Apri un terminale (Linux, macOS o Unix) o il prompt dei comandi (Windows) e usalo AWS CLI per eseguire il `update-pipeline` comando, specificando la condizione di errore nella struttura della pipeline. L'esempio seguente configura il rollback automatico per uno stage denominato: `S3Deploy`

```
{
                "name": "S3Deploy",
                "actions": [
                    {
                        "name": "s3deployaction",
                        "actionTypeId": {
                            "category": "Deploy",
                            "owner": "AWS",
                            "provider": "S3",
                            "version": "1"
                        },
                        "runOrder": 1,
                        "configuration": {
                            "BucketName": "static-website-bucket",
                            "Extract": "false",
                            "ObjectKey": "SampleApp.zip"
                        },
                        "outputArtifacts": [],
                        "inputArtifacts": [
                            {
                                "name": "SourceArtifact"
                            }
                        ],
                        "region": "us-east-1"
                    }
                ],
                "onFailure": {
                    "result": "ROLLBACK"
                }
            }
```

Per ulteriori informazioni sulla configurazione delle condizioni di errore per lo stage rollback, consulta FailureConditionsl'API Reference. CodePipeline

Configura una fase per il rollback automatico ()AWS CloudFormation

Da utilizzare AWS CloudFormation per configurare uno stadio per il rollback automatico in caso di errore, utilizzate il `OnFailure` parametro. In caso di errore, lo stage tornerà automaticamente all'ultima esecuzione riuscita.

```
OnFailure:
     Result: ROLLBACK
```

- Aggiorna il modello come mostrato nel frammento seguente. L'esempio seguente configura il rollback automatico per uno staged denominato: `Release`

```
AppPipeline:
  Type: AWS::CodePipeline::Pipeline
  Properties:
    RoleArn:
      Ref: CodePipelineServiceRole
    Stages:
      -
        Name: Source
        Actions:
          -
            Name: SourceAction
            ActionTypeId:
              Category: Source
              Owner: AWS
              Version: 1
              Provider: S3
            OutputArtifacts:
              -
                Name: SourceOutput
            Configuration:
              S3Bucket:
                Ref: SourceS3Bucket
              S3ObjectKey:
                Ref: SourceS3ObjectKey
            RunOrder: 1
```

```
    -
      Name: Release
      Actions:
        -
          Name: ReleaseAction
          InputArtifacts:
            -
              Name: SourceOutput
          ActionTypeId:
            Category: Deploy
            Owner: AWS
            Version: 1
            Provider: CodeDeploy
          Configuration:
            ApplicationName:
              Ref: ApplicationName
            DeploymentGroupName:
              Ref: DeploymentGroupName
          RunOrder: 1
     OnFailure:
          Result: ROLLBACK
  ArtifactStore:
    Type: S3
    Location:
      Ref: ArtifactStoreS3Location
    EncryptionKey:
      Id: arn:aws:kms:useast-1:ACCOUNT-ID:key/KEY-ID
      Type: KMS
  DisableInboundStageTransitions:
    -
      StageName: Release
      Reason: "Disabling the transition until integration tests are completed"
  Tags:
    - Key: Project
      Value: ProjectA
    - Key: IsContainerBased
      Value: 'true'
```

Per ulteriori informazioni sulla configurazione delle condizioni di errore per lo stage rollback, consulta la sezione successiva della Guida per OnFailurel'*StageDeclaration*utente.AWS CloudFormation

Visualizza lo stato di rollback nell'elenco delle esecuzioni

È possibile visualizzare lo stato e l'ID di esecuzione di destinazione per un'esecuzione di rollback.

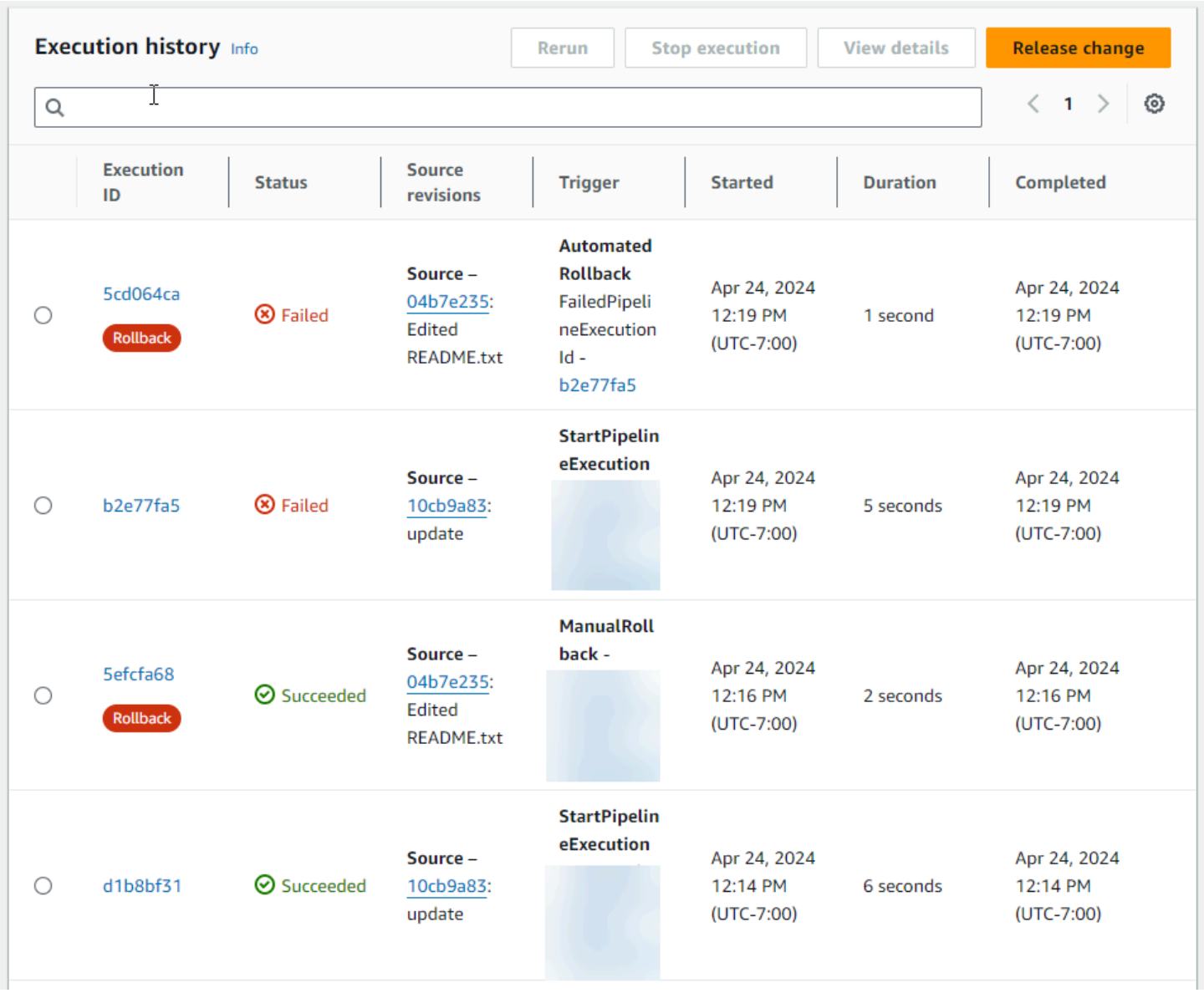
Visualizza lo stato di rollback nell'elenco delle esecuzioni (console)

È possibile utilizzare la console per visualizzare lo stato e l'ID di esecuzione di destinazione per un'esecuzione di rollback nell'elenco di esecuzione.

Visualizza lo stato di esecuzione del rollback nell'elenco delle esecuzioni (console)

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Account AWS Vengono visualizzati i nomi e lo stato di tutte le pipeline associate al tuo.
2. In Nome, scegliete il nome della pipeline che desiderate visualizzare.
3. Scegliere History (Cronologia). L'elenco delle esecuzioni mostra l'etichetta Rollback.



Scegliete l'ID di esecuzione di cui desiderate visualizzare i dettagli.

Visualizza lo stato di rollback con **`list-pipeline-executions`** (CLI)

È possibile utilizzare la CLI per visualizzare lo stato e l'ID di esecuzione di destinazione per un'esecuzione di rollback.

- Apri un terminale (Linux, macOS o Unix) o un prompt dei comandi (Windows) e usali AWS CLI per eseguire il comando: list-pipeline-executions

```
aws codepipeline list-pipeline-executions --pipeline-name MyFirstPipeline
```

Questo comando restituisce un elenco di tutte le esecuzioni completate associate alla pipeline.

L'esempio seguente mostra i dati restituiti per una pipeline denominata nel *MyFirstPipeline*punto in cui un'esecuzione di rollback ha avviato la pipeline.

```
{
    "pipelineExecutionSummaries": [
        {
            "pipelineExecutionId": "eb7ebd36-353a-4551-90dc-18ca5EXAMPLE",
            "status": "Succeeded",
            "startTime": "2024-04-16T09:00:28.185000+00:00",
            "lastUpdateTime": "2024-04-16T09:00:29.665000+00:00",
            "sourceRevisions": [
                {
                    "actionName": "Source",
                    "revisionId": "revision_ID",
                    "revisionSummary": "Added README.txt",
                    "revisionUrl": "console-URL"
                }
            ],
            "trigger": {
                "triggerType": "ManualRollback",
                "triggerDetail": "{arn:aws:sts::<account_ID>:assumed-role/<role>"}"
            },
            "executionMode": "SUPERSEDED",
            "executionType": "ROLLBACK",
            "rollbackMetadata": {
                "rollbackTargetPipelineExecutionId":
 "f15b38f7-20bf-4c9e-94ed-2535eEXAMPLE"
            }
        },
        {
            "pipelineExecutionId": "fcd61d8b-4532-4384-9da1-2aca1EXAMPLE",
            "status": "Succeeded",
            "startTime": "2024-04-16T08:58:56.601000+00:00",
            "lastUpdateTime": "2024-04-16T08:59:04.274000+00:00",
            "sourceRevisions": [
                {
                    "actionName": "Source",
                    "revisionId": "revision_ID",
                    "revisionSummary": "Added README.txt",
                    "revisionUrl": "console_URL"
```

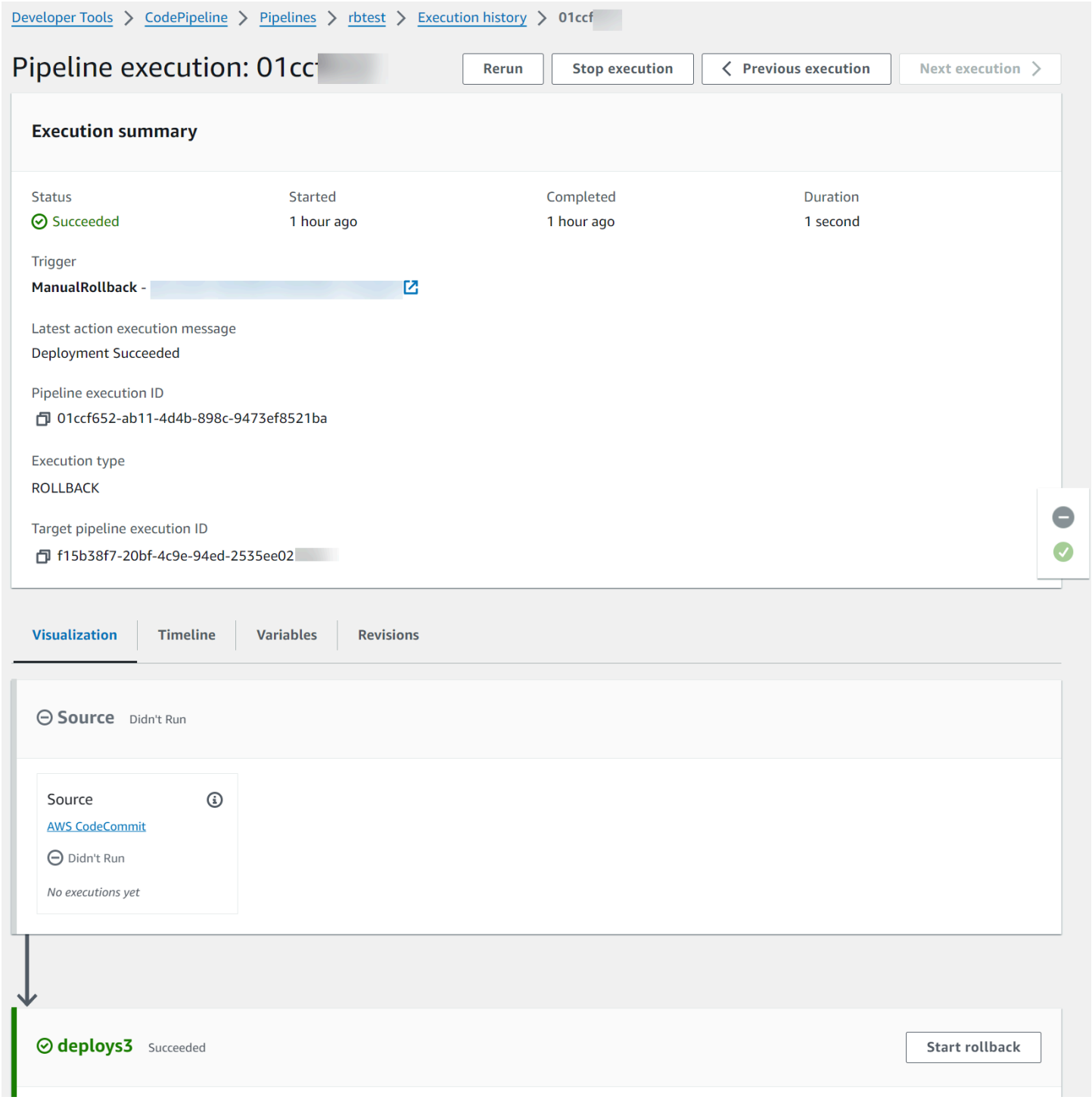
```
                }
            ],
            "trigger": {
                "triggerType": "StartPipelineExecution",
                "triggerDetail": "arn:aws:sts::<account_ID>:assumed-role/<role>"
            },
            "executionMode": "SUPERSEDED"
        },
        {
            "pipelineExecutionId": "5cd064ca-bff7-425f-8653-f41d9EXAMPLE",
            "status": "Failed",
            "startTime": "2024-04-24T19:19:50.781000+00:00",
            "lastUpdateTime": "2024-04-24T19:19:52.119000+00:00",
            "sourceRevisions": [
                {
                    "actionName": "Source",
                    "revisionId": "<revision_ID>",
                    "revisionSummary": "Edited README.txt",
                    "revisionUrl": "<revision_URL>"
                }
            ],
            "trigger": {
                "triggerType": "AutomatedRollback",
                "triggerDetail": "{\"FailedPipelineExecutionId\":
\"b2e77fa5-9285-4dea-ae66-4389EXAMPLE\"}"
            },
            "executionMode": "SUPERSEDED",
            "executionType": "ROLLBACK",
            "rollbackMetadata": {
                "rollbackTargetPipelineExecutionId": "5efcfa68-d838-4ca7-
a63b-4a743EXAMPLE"
            }
         },
```

Visualizza i dettagli sullo stato del rollback

È possibile visualizzare lo stato e l'ID di esecuzione di destinazione per un'esecuzione di rollback.

Visualizza lo stato del rollback nella pagina di dettaglio (console)

È possibile utilizzare la console per visualizzare lo stato e l'ID di esecuzione della pipeline di destinazione per un'esecuzione di rollback.



Visualizza i dettagli del rollback con **`get-pipeline-execution`** (CLI)

Le esecuzioni di pipeline che sono state ripristinate verranno visualizzate nell'output relativo all'esecuzione della pipeline.

- Per visualizzare i dettagli relativi a una pipeline, eseguire il comando get-pipeline-execution, specificando il nome univoco della pipeline. Ad esempio, per visualizzare i dettagli su una pipeline denominata *`MyFirstPipeline`*, inserisci quanto segue:

  ```
  aws codepipeline get-pipeline-execution --pipeline-name MyFirstPipeline --pipeline-
  execution-id 3f658bd1-69e6-4448-ba3e-79007EXAMPLE
  ```

  Questo comando restituisce la struttura della pipeline.

  L'esempio seguente mostra i dati restituiti per una parte di una pipeline denominata *`MyFirstPipeline`*, in cui vengono visualizzati l'ID di esecuzione del rollback e i metadati.

  ```
  {
      "pipelineExecution": {
          "pipelineName": "MyFirstPipeline",
          "pipelineVersion": 6,
          "pipelineExecutionId": "2004a94e-8b46-4c34-a695-c8d20EXAMPLE",
          "status": "Succeeded",
          "artifactRevisions": [
              {
                  "name": "SourceArtifact",
                  "revisionId": "<ID>",
                  "revisionSummary": "Added README.txt",
                  "revisionUrl": "<console_URL>"
              }
          ],
          "trigger": {
              "triggerType": "ManualRollback",
              "triggerDetail": "arn:aws:sts::<account_ID>:assumed-role/<role>"
          },
          "executionMode": "SUPERSEDED",
          "executionType": "ROLLBACK",
          "rollbackMetadata": {
              "rollbackTargetPipelineExecutionId": "4f47bed9-6998-476c-a49d-
  e60beEXAMPLE"
          }
      }
  ```

```
}
```

Visualizza lo stato di rollback con **`get-pipeline-state`** (CLI)

Le esecuzioni di pipeline che sono state ripristinate verranno visualizzate nell'output per ottenere lo stato della pipeline.

- Per visualizzare i dettagli relativi a una pipeline, eseguire il comando get-pipeline-state, specificando il nome univoco della pipeline. Ad esempio, per visualizzare i dettagli sullo stato di una pipeline denominata *`MyFirstPipeline`*, inserisci quanto segue:

  ```
  aws codepipeline get-pipeline-state --name MyFirstPipeline
  ```

  L'esempio seguente mostra i dati restituiti con il tipo di esecuzione rollback.

  ```
  {
      "pipelineName": "MyFirstPipeline",
      "pipelineVersion": 7,
      "stageStates": [
          {
              "stageName": "Source",
              "inboundExecutions": [],
              "inboundTransitionState": {
                  "enabled": true
              },
              "actionStates": [
                  {
                      "actionName": "Source",
                      "currentRevision": {
                          "revisionId": "<Revision_ID>"
                      },
                      "latestExecution": {
                          "actionExecutionId": "13bbd05d-
  b439-4e35-9c7e-887cb789b126",
                          "status": "Succeeded",
                          "summary": "update",
                          "lastStatusChange": "2024-04-24T20:13:45.799000+00:00",
                          "externalExecutionId": "10cbEXAMPLEID"
                      },
                      "entityUrl": "console-url",
                      "revisionUrl": "console-url"
  ```

```
                }
            ],
            "latestExecution": {
                "pipelineExecutionId": "cf95a8ca-0819-4279-ae31-03978EXAMPLE",
                "status": "Succeeded"
            }
        },
        {
            "stageName": "deploys3",
            "inboundExecutions": [],
            "inboundTransitionState": {
                "enabled": true
            },
            "actionStates": [
                {
                    "actionName": "s3deploy",
                    "latestExecution": {
                        "actionExecutionId":
 "3bc4e3eb-75eb-45b9-8574-8599aEXAMPLE",
                        "status": "Succeeded",
                        "summary": "Deployment Succeeded",
                        "lastStatusChange": "2024-04-24T20:14:07.577000+00:00",
                        "externalExecutionId": "mybucket/SampleApp.zip"
                    },
                    "entityUrl": "console-URL"
                }
            ],
            "latestExecution": {
                "pipelineExecutionId": "fdf6b2ae-1472-4b00-9a83-1624eEXAMPLE",
                "status": "Succeeded",
                "type": "ROLLBACK"
            }
        }
    ],
    "created": "2024-04-15T21:29:01.635000+00:00",
    "updated": "2024-04-24T20:12:24.480000+00:00"
}
```

Lavorare con le azioni in CodePipeline

In AWS CodePipeline, un'azione fa parte della sequenza in una fase di una pipeline. Si tratta di un'attività eseguita sull'artefatto in tale fase. Le operazioni nella pipeline sono disposte in un ordine specifico, in sequenza o in parallelo, secondo quanto stabilito nella configurazione della fase.

CodePipeline fornisce supporto per sei tipi di azioni:

- Origine
- Creazione
- Test
- Implementazione
- Approval
- Invoke

Per informazioni sui prodotti Servizio AWS e i servizi dei partner che puoi integrare nella tua pipeline in base al tipo di azione, consultaIntegrazioni con tipi di CodePipeline azioni.

Argomenti

Lavorare con i tipi di azione

I tipi di azione sono azioni preconfigurate che il fornitore crea per i clienti utilizzando uno dei modelli di integrazione supportati in AWS CodePipeline.

È possibile richiedere, visualizzare e aggiornare i tipi di azione. Se il tipo di azione è stato creato per il tuo account come proprietario, puoi utilizzarlo AWS CLI per visualizzare o aggiornare le proprietà e la struttura del tipo di azione. Se sei il fornitore o il proprietario del tipo di azione, i tuoi clienti possono scegliere l'azione e aggiungerla alle loro pipeline dopo che sarà disponibile in CodePipeline.

> **Note**
>
> Puoi creare azioni sul `custom` `owner` campo da eseguire con un job worker. Non le crei con un modello di integrazione. Per informazioni sulle azioni personalizzate, vedereCreare e aggiungere un'azione personalizzata in CodePipeline.

Componenti del tipo di azione

I seguenti componenti costituiscono un tipo di azione.

- ID del tipo di azione: l'ID è composto dalla categoria, dal proprietario, dal provider e dalla versione. L'esempio seguente mostra un ID di tipo di azione con un proprietario`ThirdParty`, una categoria di`Test`, un provider denominato `TestProvider` e una versione di`1`.

```
{
    "Category": "Test",
    "Owner": "ThirdParty",
    "Provider": "TestProvider",
    "Version": "1"
},
```

- Configurazione dell'esecutore: il modello di integrazione, o motore di azione, specificato al momento della creazione dell'azione. Quando si specifica l'esecutore per un tipo di azione, si sceglie uno dei due tipi:
  - Lambda: il proprietario del tipo di azione scrive l'integrazione come una funzione Lambda, che viene richiamata da CodePipeline ogni volta che è disponibile un lavoro per l'azione.
  - JobWorker: Il proprietario del tipo di azione scrive l'integrazione come job worker che analizza i lavori disponibili nelle pipeline dei clienti. Il job worker esegue quindi il lavoro e invia i risultati del lavoro CodePipeline utilizzando le API. CodePipeline

> **Note**
>
> Il modello di integrazione dei lavoratori non è il modello di integrazione preferito.

- Elementi di input e output: limiti per gli artefatti che il proprietario del tipo di azione designa per i clienti dell'azione.
- Autorizzazioni: la strategia di autorizzazione che designa i clienti che possono accedere al tipo di azione di terze parti. Le strategie di autorizzazione disponibili dipendono dal modello di integrazione scelto per il tipo di azione.
- URL: collegamenti diretti a risorse con cui il cliente può interagire, come la pagina di configurazione del proprietario del tipo di azione.

Argomenti

Richiedi un tipo di azione

Quando viene richiesto un nuovo tipo di CodePipeline azione da un provider di terze parti, il tipo di azione viene creato per il proprietario del tipo di azione in CodePipeline e il proprietario può gestire e visualizzare il tipo di azione.

Un tipo di azione può essere un'azione privata o pubblica. Quando viene creato, il tipo di azione è privato. Per richiedere la modifica di un tipo di azione in un'azione pubblica, contatta il team CodePipeline di assistenza.

Prima di creare il file di definizione dell'azione, le risorse per l'esecutore e la richiesta del tipo di azione per il CodePipeline team, devi scegliere un modello di integrazione.

Fase 1: Scegli il tuo modello di integrazione

Scegli il tuo modello di integrazione e poi crea la configurazione per quel modello. Dopo aver scelto il modello di integrazione, è necessario configurare le risorse di integrazione.

- Per il modello di integrazione Lambda, si crea una funzione Lambda e si aggiungono le autorizzazioni. Aggiungi le autorizzazioni alla funzione Lambda del tuo integratore per fornire al servizio CodePipeline le autorizzazioni per richiamarlo utilizzando il service principal:. CodePipeline `codepipeline.amazonaws.com` Le autorizzazioni possono essere aggiunte utilizzando o la riga di comando. AWS CloudFormation
  - Esempio di aggiunta di autorizzazioni utilizzando: AWS CloudFormation

```
  CodePipelineLambdaBasedActionPermission:
    Type: 'AWS::Lambda::Permission'
    Properties:
      Action: 'lambda:invokeFunction'
      FunctionName: {"Fn::Sub": "arn:${AWS::Partition}:lambda:${AWS::Region}:
${AWS::AccountId}:function:function-name"}
      Principal: codepipeline.amazonaws.com
```

  - Documentazione per la riga di comando
- Per il modello di integrazione dei job worker, crei un'integrazione con un elenco di account consentiti in cui il job worker effettua sondaggi per le offerte di lavoro con le CodePipeline API.

Fase 2: Creare un file di definizione del tipo di azione

Si definisce un tipo di azione in un file di definizione del tipo di azione utilizzando JSON. Nel file, includi la categoria di azione, il modello di integrazione utilizzato per gestire il tipo di azione e le proprietà di configurazione.

> **Note**
>
> Dopo aver creato un'azione pubblica, non puoi modificare la proprietà del tipo di azione in `properties` da `optional` a `required`. Inoltre, non puoi modificare il `owner`.

Per ulteriori informazioni sui parametri del file di definizione del tipo di azione, consulta ActionTypeDeclaratione UpdateActionTypenell'CodePipeline API Reference.

Il file di definizione del tipo di azione contiene otto sezioni:

- `description`: la descrizione del tipo di azione da aggiornare.
- `executor`: Informazioni sull'esecutore per un tipo di azione creato con un modello di integrazione supportato, `Lambda` oppure`job worker`. È possibile fornire solo uno dei due

`jobWorkerExecutorConfiguration` o `lambdaExecutorConfiguration`, in base al tipo di esecutore.

- `configuration`: Risorse per la configurazione del tipo di azione, in base al modello di integrazione scelto. Per il modello di integrazione Lambda, usa la funzione Lambda ARN. Per il modello di integrazione del job worker, utilizza l'account o l'elenco di account da cui esegue il job worker.
- `jobTimeout`: Il timeout in secondi per il lavoro. L'esecuzione di un'azione può consistere in più processi. Questo è il timeout per un singolo processo e non per l'intera esecuzione dell'azione.

  > **Note**
  >
  > Per il modello di integrazione Lambda, il timeout massimo è di 15 minuti.

- `policyStatementsTemplate`: La dichiarazione politica che specifica le autorizzazioni nell'account del CodePipeline cliente necessarie per eseguire correttamente l'esecuzione di un'azione.
- `type`: il modello di integrazione utilizzato per creare e aggiornare il tipo di azione, oppure`Lambda`. `JobWorker`

- `id`: La categoria, il proprietario, il provider e l'ID di versione per il tipo di azione:
  - `category`: Il tipo di azione può essere intrapreso nella fase: Source, Build, Deploy, Test, Invoke o Approval.
  - `provider`: Il fornitore del tipo di azione chiamato, ad esempio la società del fornitore o il nome del prodotto. Il nome del provider viene fornito al momento della creazione del tipo di azione.
  - `owner`: Il creatore del tipo di azione chiamato: `AWS` o`ThirdParty`.
  - `version`: Una stringa utilizzata per la versione del tipo di azione. Per la prima versione, imposta il numero di versione su 1.
- `inputArtifactDetails`: Il numero di artefatti da aspettarsi dalla fase precedente della pipeline.
- `outputArtifactDetails`: Il numero di artefatti da aspettarsi dal risultato della fase relativa al tipo di azione.
- `permissions`: dettagli che identificano gli account autorizzati a utilizzare il tipo di azione.
- `properties`: i parametri necessari per il completamento delle attività del progetto.
  - `description`: La descrizione della proprietà che viene visualizzata agli utenti.
  - `optional`: Se la proprietà di configurazione è facoltativa.

  - `noEcho`: Se il valore del campo inserito dal cliente viene omesso dal registro. Se `true`, allora il valore viene oscurato quando viene restituito con una richiesta GetPipeline API.
  - `key`: Se la proprietà di configurazione è una chiave.
  - `queryable`: Se la proprietà viene utilizzata con il polling. Un tipo di azione può avere fino a una proprietà interrogabile. Se una è presente, la proprietà deve essere obbligatoria e non segreta.
  - `name`: il nome della proprietà che viene visualizzato agli utenti.
- `urls`: CodePipeline viene visualizzato un elenco degli URL per gli utenti.
  - `entityUrlTemplate`: URL alle risorse esterne per il tipo di azione, ad esempio una pagina di configurazione.
  - `executionUrlTemplate`: URL ai dettagli dell'ultima esecuzione dell'azione.
  - `revisionUrlTemplate`: URL visualizzato nella CodePipeline console alla pagina in cui i clienti possono aggiornare o modificare la configurazione dell'azione esterna.
  - `thirdPartyConfigurationUrl`: URL di una pagina in cui gli utenti possono iscriversi a un servizio esterno ed eseguire la configurazione iniziale dell'azione fornita da tale servizio.

Il codice seguente mostra un esempio di file di definizione del tipo di azione.

```
{
   "actionType": {
      "description": "string",
      "executor": {
         "configuration": {
            "jobWorkerExecutorConfiguration": {
               "pollingAccounts": [ "string" ],
               "pollingServicePrincipals": [ "string" ]
            },
            "lambdaExecutorConfiguration": {
               "lambdaFunctionArn": "string"
            }
         },
         "jobTimeout": number,
         "policyStatementsTemplate": "string",
         "type": "string"
      },
      "id": {
         "category": "string",
         "owner": "string",
         "provider": "string",
```

```
         "version": "string"
      },
      "inputArtifactDetails": {
         "maximumCount": number,
         "minimumCount": number
      },
      "outputArtifactDetails": {
         "maximumCount": number,
         "minimumCount": number
      },
      "permissions": {
         "allowedAccounts": [ "string" ]
      },
      "properties": [
         {
            "description": "string",
            "key": boolean,
            "name": "string",
            "noEcho": boolean,
            "optional": boolean,
            "queryable": boolean
         }
      ],
      "urls": {
         "configurationUrl": "string",
         "entityUrlTemplate": "string",
         "executionUrlTemplate": "string",
         "revisionUrlTemplate": "string"
      }
   }
}
```

Fase 3: Registra la tua integrazione con CodePipeline

Per registrare il tipo di azione CodePipeline, contatta il team CodePipeline di assistenza con la tua richiesta.

Il team CodePipeline di assistenza registra la nuova integrazione del tipo di azione apportando modifiche alla codebase del servizio. CodePipeline registra due nuove azioni: un'azione pubblica e un'azione privata. Utilizzi l'azione privata per i test e, quando sei pronto, attivi l'azione pubblica per servire il traffico dei clienti.

Per registrare una richiesta di integrazione con Lambda

- Invia una richiesta al team CodePipeline di assistenza utilizzando il seguente modulo.

```
This issue will track the onboarding of [Name] in CodePipeline.


[Contact engineer] will be the primary point of contact for this integration.

Name of the action type as you want it to appear to customers: Example.com Testing

Initial onboard checklist:

1. Attach an action type definition file in JSON format. This includes the schema
 for the action type

2. A list of test accounts for the allowlist which can access the new action type
 [{account, account_name}]

3. The Lambda function ARN

4. List of Regioni AWS where your action will be available

5. Will this be available as a public action?
```

Per registrare una richiesta di integrazione tra lavoratori e lavoratori

- Invia una richiesta al team CodePipeline di assistenza utilizzando il seguente modulo.

```
This issue will track the onboarding of [Name] in CodePipeline.

[Contact engineer] will be the primary point of contact for this integration.


Name of the action type as you want it to appear to customers: Example.com Testing

Initial onboard checklist:

1. Attach an action type definition file in JSON format. This includes the schema
 for the action type.
```

```
2. A list of test accounts for the allowlist which can access the new action type
 [{account, account_name}]

3. URL information:
Website URL: https://www.example.com/%TestThirdPartyName%/%TestVersionNumber%

Example URL pattern where customers will be able to review their configuration
 information for the action: https://www.example.com/%TestThirdPartyName%/
%customer-ID%/%CustomerActionConfiguration%

Example runtime URL pattern: https://www.example.com/%TestThirdPartyName%/
%customer-ID%/%TestRunId%

4. List of Regioni AWS where your action will be available

5. Will this be available as a public action?
```

Fase 4: Attiva la tua nuova integrazione

Contatta il team CodePipeline di assistenza quando sei pronto per utilizzare pubblicamente la nuova integrazione.

Aggiungi un tipo di azione disponibile a una pipeline (console)

Aggiungi il tuo tipo di azione a una pipeline in modo da poterlo testare. Puoi farlo creando una nuova pipeline o modificandone una esistente.

> **Note**
>
> Se il tipo di azione è un'azione di categoria di origine, creazione o distribuzione, è possibile aggiungerla creando una pipeline. Se il tipo di azione è nella categoria di test, devi aggiungerlo modificando una pipeline esistente.

Per aggiungere il tipo di azione a una pipeline esistente dalla console CodePipeline

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nell'elenco delle pipeline, scegli la pipeline in cui desideri aggiungere il tipo di azione.
3. Nella pagina di visualizzazione di riepilogo della pipeline, scegli Modifica.

4. Scegliete di modificare la fase. Nella fase in cui desideri aggiungere il tipo di azione, scegli Aggiungi gruppo di azioni. Viene visualizzata la pagina Modifica azione.
5. Nella pagina Modifica azione, in Nome azione, inserisci un nome per l'azione. Questo è il nome visualizzato per la fase della pipeline.
6. In Action provider, scegli il tipo di azione dall'elenco.

   Nota che il valore nell'elenco si basa su quello `provider` specificato nel file di definizione del tipo di azione.
7. In Input artifacts, inserite il nome dell'artefatto in questo formato:

   `Artifactname``::`*`FileName`*

   Si noti che le quantità minime e massime consentite sono definite in base a `inputArtifactDetails` quanto specificato nel file di definizione del tipo di azione.
8. Scegli Connect a<Action_Name>.

   Si apre una finestra del browser che si collega al sito Web creato per il tipo di azione.
9. Accedi al tuo sito web come cliente e completa i passaggi che un cliente esegue per utilizzare il tuo tipo di azione. I passaggi variano a seconda della categoria di azione, del sito Web e della configurazione, ma in genere includono un'azione di completamento che riporta il cliente alla pagina Modifica azione.
10. Nella pagina CodePipeline Modifica azione, vengono visualizzati i campi di configurazione aggiuntivi per l'azione. I campi visualizzati sono le proprietà di configurazione specificate nel file di definizione dell'azione. Immettete le informazioni nei campi personalizzati per il tipo di azione.

    Ad esempio, se il file di definizione dell'azione specifica una proprietà denominata`Host`, nella pagina Modifica azione viene visualizzato un campo con l'etichetta Host relativa all'azione.
11. In Output artifacts, inserite il nome dell'artefatto in questo formato:

    `Artifactname``::`*`FileName`*

    Notate che le quantità minime e massime consentite sono definite in base a `outputArtifactDetails` quanto specificato nel file di definizione del tipo di azione.
12. Scegliete Fine per tornare alla pagina dei dettagli della pipeline.

> **Note**
>
> I tuoi clienti possono opzionalmente utilizzare la CLI per aggiungere il tipo di azione alla loro pipeline.

13. Per testare la tua azione, esegui una modifica all'origine specificata nella fase di origine della pipeline o segui i passaggi in Avvio manuale di una pipeline.

Per creare una pipeline con il tuo tipo di azione, segui i passaggi indicati Creare una pipeline in CodePipeline e scegli il tipo di azione tra tutte le fasi che testerai.

Visualizza un tipo di azione

Puoi utilizzare la CLI per visualizzare il tipo di azione. Utilizzate il get-action-type comando per visualizzare i tipi di azione che sono stati creati utilizzando un modello di integrazione.

Per visualizzare un tipo di azione

1. Crea un file JSON di input e assegna un nome al file`file.json`. Aggiungi l'ID del tipo di azione in formato JSON come mostrato nell'esempio seguente.

```
{
    "category": "Test",
    "owner": "ThirdParty",
    "provider": "TestProvider",
    "version": "1"
}
```

2. In una finestra di terminale o nella riga di comando, esegui il get-action-type comando.

```
aws codepipeline get-action-type --cli-input-json file://file.json
```

   Questo comando restituisce l'output della definizione dell'azione per un tipo di azione. Questo esempio mostra un tipo di azione creato con il modello di integrazione Lambda.

```
{
    "actionType": {
        "executor": {
            "configuration": {
```

```
                "lambdaExecutorConfiguration": {
                    "lambdaFunctionArn": "arn:aws:lambda:us-west-2:<account-
id>:function:my-function"
                }
            },
            "type": "Lambda"
        },
        "id": {
            "category": "Test",
            "owner": "ThirdParty",
            "provider": "TestProvider",
            "version": "1"
        },
        "inputArtifactDetails": {
            "minimumCount": 0,
            "maximumCount": 1
        },
        "outputArtifactDetails": {
            "minimumCount": 0,
            "maximumCount": 1
        },
        "permissions": {
            "allowedAccounts": [
                "<account-id>"
            ]
        },
        "properties": []
    }
}
```

Aggiornare un tipo di azione

È possibile utilizzare la CLI per modificare i tipi di azioni creati con un modello di integrazione.

Per un tipo di azione pubblica, non puoi aggiornare il proprietario, non puoi modificare le proprietà opzionali in obbligatorie e puoi solo aggiungere nuove proprietà opzionali.

1. Usa il `get-action-type` comando per ottenere la struttura per il tuo tipo di azione. Copia la struttura.

2. Crea un file JSON di input e assegnagli un nome. `action.json` Incolla al suo interno la struttura del tipo di azione che hai copiato nel passaggio precedente. Aggiorna tutti i parametri che desideri modificare. Puoi anche aggiungere parametri opzionali.

   Per ulteriori informazioni sui parametri per il file di input, vedere la descrizione del file di definizione dell'azione inFase 2: Creare un file di definizione del tipo di azione.

   L'esempio seguente mostra come aggiornare un tipo di azione di esempio creato con il modello di integrazione Lambda. Questo esempio apporta le seguenti modifiche:

   - Cambia il `provider` nome in`TestProvider1`.
   - Aggiunge un limite di timeout del lavoro di 900 secondi.
   - Aggiunge una proprietà di configurazione dell'azione denominata `Host` che viene visualizzata al cliente che utilizza l'azione.

```
{
    "actionType": {
        "executor": {
            "configuration": {
                "lambdaExecutorConfiguration": {
                    "lambdaFunctionArn": "arn:aws:lambda:us-west-2:<account-
id>:function:my-function"
                }
            },
            "type": "Lambda",
            "jobTimeout": 900
        },
        "id": {
            "category": "Test",
            "owner": "ThirdParty",
            "provider": "TestProvider1",
            "version": "1"
        },
        "inputArtifactDetails": {
            "minimumCount": 0,
            "maximumCount": 1
        },
        "outputArtifactDetails": {
            "minimumCount": 0,
            "maximumCount": 1
        },
        "permissions": {
```

```
            "allowedAccounts": [
                "account-id"
            ]
        },
        "properties": {
         "description": "Owned build action parameter description",
         "optional": true,
         "noEcho": false,
         "key": true,
         "queryable": false,
         "name": "Host"
         }
    }
}
```

3. Nel terminale o nella riga di comando, esegui il update-action-type comando

```
aws codepipeline update-action-type --cli-input-json file://action.json
```

Questo comando restituisce l'output del tipo di azione in modo che corrisponda ai parametri aggiornati.

Creare e aggiungere un'azione personalizzata in CodePipeline

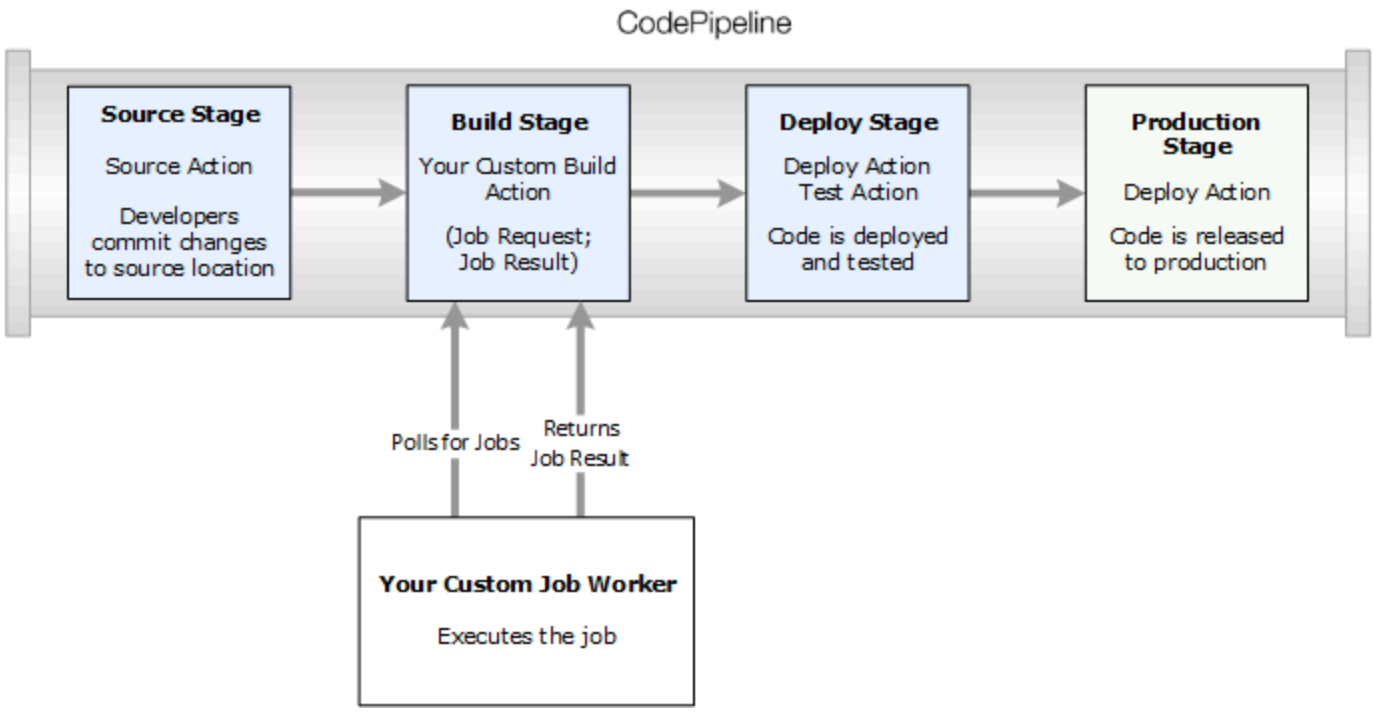
AWS CodePipeline include una serie di azioni che consentono di configurare, compilare, testare e distribuire risorse per il processo di rilascio automatizzato. Se il processo di rilascio comprende attività che non sono incluse nelle operazioni predefinite, ad esempio un processo di compilazione sviluppato internamente o una suite di test, puoi creare un'operazione personalizzata a tale scopo e includerla nella pipeline. Puoi utilizzare il AWS CLI per creare azioni personalizzate nelle pipeline associate al tuo AWS account.

Puoi creare azioni personalizzate per le seguenti categorie di AWS CodePipeline azioni:

- Un'operazione di compilazione personalizzata che consente di compilare o trasformare gli elementi
- Un'operazione di distribuzione personalizzata che consente di distribuire elementi in uno o più server, siti Web o repository
- Un'operazione di test personalizzata che consente di configurare ed eseguire test automatici
- Un'operazione di richiamo personalizzata che consente di eseguire funzioni

Quando si crea un'azione personalizzata, è necessario creare anche un job worker che elabori le richieste di lavoro CodePipeline per questa azione personalizzata, esegua il lavoro e restituisca il risultato dello stato a CodePipeline. Questo job worker può essere localizzato su qualsiasi computer o risorsa purché abbia accesso all'endpoint pubblico di. CodePipeline Per gestire facilmente l'accesso e la sicurezza, prendi in considerazione la possibilità di ospitare il tuo job worker su un'istanza Amazon EC2.

Il seguente diagramma mostra una visualizzazione generale di una pipeline che include un'operazione di compilazione personalizzata:



Quando una pipeline include un'azione personalizzata come parte di una fase, la pipeline creerà una richiesta di lavoro. Un esecutore del processo personalizzato rileva tale richiesta ed esegue il processo (in questo esempio, un processo personalizzato utilizzando un software di compilazione di terze parti). Al termine dell'operazione, l'esecutore del processo restituisce un esito positivo o negativo. Se viene ricevuto un risultato positivo, la pipeline fornirà la revisione e i relativi artefatti all'azione successiva. Se viene restituito un errore, la pipeline non fornirà la revisione all'azione successiva nella pipeline.

> **Note**
>
> Queste istruzioni presuppongono che la procedura in Guida introduttiva con CodePipeline sia già stata completata.

Argomenti

Creazione di un'operazione personalizzata

Per creare un'azione personalizzata con AWS CLI

1. Apri un editor di testo e crea un file JSON per l'operazione personalizzata che include la categoria dell'operazione, il provider di operazioni e le eventuali impostazioni richieste dall'operazione personalizzata. Ad esempio, per creare un'operazione di compilazione personalizzata che richiede una sola proprietà, l'aspetto del file JSON potrebbe essere simile al seguente:

```
{
    "category": "Build",
    "provider": "My-Build-Provider-Name",
    "version": "1",
    "settings": {
        "entityUrlTemplate": "https://my-build-instance/job/{Config:ProjectName}/",
        "executionUrlTemplate": "https://my-build-instance/job/
{Config:ProjectName}/lastSuccessfulBuild/{ExternalExecutionId}/"
    },
    "configurationProperties": [{
        "name": "ProjectName",
        "required": true,
        "key": true,
        "secret": false,
        "queryable": false,
        "description": "The name of the build project must be provided when this
 action is added to the pipeline.",
        "type": "String"
```

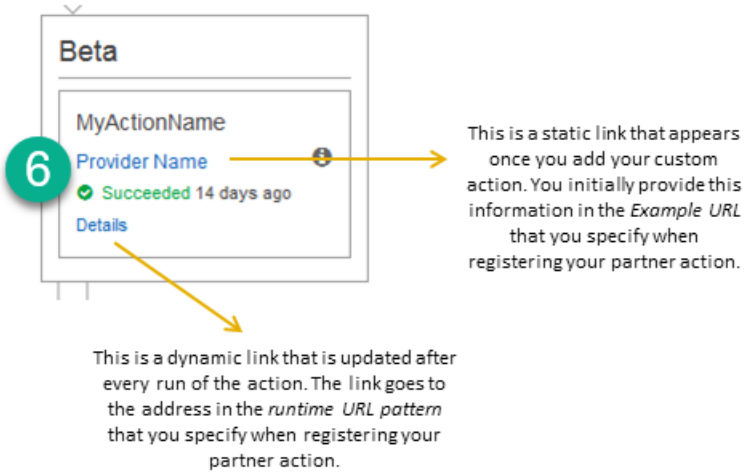
```
    }],
    "inputArtifactDetails": {
        "maximumCount": integer,
        "minimumCount": integer
    },
    "outputArtifactDetails": {
        "maximumCount": integer,
        "minimumCount": integer
    },
    "tags": [{
      "key": "Project",
      "value": "ProjectA"
    }]
}
```

In questo esempio all'operazione personalizzata viene aggiunto del tagging includendo la chiave di tag `Project` e il valore `ProjectA` all'operazione personalizzata. Per ulteriori informazioni sull'aggiunta di tag alle risorse CodePipeline, consultaAssegnazione di tag alle risorse.

Nel file JSON sono incluse due proprietà, `entityUrlTemplate` e `executionUrlTemplate`. Puoi fare riferimento a un nome nelle proprietà di configurazione dell'operazione personalizzata all'interno dei modelli URL seguendo il formato di `{Config:name}`, purché la proprietà di configurazione sia richiesta e non segreta. Ad esempio, nell'esempio precedente, il `entityUrlTemplate` valore si riferisce alla proprietà *`ProjectName`*di configurazione.

- `entityUrlTemplate`: il collegamento statico che fornisce informazioni sul provider di servizi per l'operazione. Nell'esempio, il sistema di compilazione include un collegamento statico a ciascun progetto di compilazione. Il formato del collegamento varia a seconda del provider di compilazione (o, se stai creando un tipo di operazione diversa, ad esempio di test, altro provider di servizi). Quando viene aggiunta un'operazione personalizzata, questo formato di collegamento è necessario per consentire all'utente di scegliere tale collegamento per aprire un browser a una pagina del sito Web che fornisce le specifiche per il progetto di compilazione (o ambiente di test).
- `executionUrlTemplate`: il collegamento dinamico che verrà aggiornato con le informazioni relative all'esecuzione corrente o più recente dell'operazione. Quando l'esecutore del processo personalizzato aggiorna lo stato di un processo (ad esempio, riuscito, non riuscito o in corso), fornirà anche un `externalExecutionId` che verrà utilizzato per completare il collegamento. Questo collegamento può essere utilizzato per fornire i dettagli relativi all'esecuzione di un'operazione.

Ad esempio, quando visualizzi l'operazione nella pipeline, vengono mostrati i seguenti due collegamenti:



1

Questo collegamento statico viene visualizzato dopo l'aggiunta di un'operazione personalizzata e punta all'indirizzo in `entityUrlTemplate`, che viene specificato quando si crea l'operazione personalizzata.

2

Questo collegamento dinamico viene aggiornato dopo ogni esecuzione dell'operazione e punta all'indirizzo in `executionUrlTemplate`, che viene specificato quando si crea l'operazione personalizzata.

Per ulteriori informazioni su questi tipi di link, nonché su `RevisionURLTemplate` e `ThirdPartyURL`, consulta ActionTypeSettingse CreateCustomActionTypenell'CodePipeline API Reference. Per ulteriori informazioni sui requisiti della struttura dell'operazione e su come creare un'operazione, consulta CodePipeline riferimento alla struttura della tubazione.

2. Salva il file JSON e assegnagli un nome facile da ricordare (ad esempio, *`MyCustomAction`*.json).

3. Apri una sessione del terminale (Linux, OS X, Unix) o un prompt dei comandi (Windows) su un computer in cui hai installato AWS CLI.
4. Usa il AWS CLI per eseguire il aws codepipeline create-custom-action-type comando, specificando il nome del file JSON che hai appena creato.

   Ad esempio, per creare un'azione personalizzata di creazione:

   > **Important**
   >
   > Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

   ```
   aws codepipeline create-custom-action-type --cli-input-json
    file://MyCustomAction.json
   ```

5. Questo comando restituisce l'intera struttura dell'operazione personalizzata creata, nonché la proprietà di configurazione dell'operazione `JobList` che viene aggiunta per l'utente. Quando aggiungi l'operazione personalizzata a una pipeline, puoi utilizzare `JobList` per specificare per quali progetti del provider è possibile eseguire il polling per processi. In mancanza di configurazione, tutti i processi disponibili verranno restituiti quando l'esecutore del processo personalizzato esegue il polling dei processi.

   Ad esempio, il comando precedente potrebbe restituire una struttura simile alla seguente:

   ```
   {
       "actionType": {
           "inputArtifactDetails": {
               "maximumCount": 1,
               "minimumCount": 1
          },
          "actionConfigurationProperties": [
               {
                   "secret": false,
                   "required": true,
                   "name": "ProjectName",
                   "key": true,
                   "description": "The name of the build project must be provided when
    this action is added to the pipeline."
               }
   ```

```
        ],
        "outputArtifactDetails": {
            "maximumCount": 0,
            "minimumCount": 0
        },
        "id": {
            "category": "Build",
            "owner": "Custom",
            "version": "1",
            "provider": "My-Build-Provider-Name"
        },
        "settings": {
            "entityUrlTemplate": "https://my-build-instance/job/
{Config:ProjectName}/",
            "executionUrlTemplate": "https://my-build-instance/job/mybuildjob/
lastSuccessfulBuild/{ExternalExecutionId}/"
        }
    }
}
```

> **Note**
>
> Come parte dell'output del create-custom-action-type comando, la `id` sezione include`"owner": "Custom"`. CodePipeline assegna automaticamente `Custom` come proprietario i tipi di azione personalizzati. Questo valore non può essere assegnato o modificato quando utilizzi il comando create-custom-action-type o update-pipeline.

Creazione di un esecutore del processo per l'operazione personalizzata

Le azioni personalizzate richiedono un job worker che analizzi le richieste di lavoro CodePipeline relative all'azione personalizzata, esegua il lavoro e restituisca il risultato dello stato a. CodePipeline Il job worker può trovarsi su qualsiasi computer o risorsa purché abbia accesso all'endpoint pubblico per. CodePipeline

Esistono molti modi per designare l'esecutore del processo. Le sezioni seguenti forniscono alcune indicazioni pratiche per lo sviluppo del job worker personalizzato per CodePipeline.

Argomenti

- Scelta e configurazione di una strategia di gestione delle autorizzazioni per l'esecutore del processo
- Sviluppo di un esecutore del processo per l'operazione personalizzata
- Architettura dell'esecutore del processo personalizzato ed esempi

Scelta e configurazione di una strategia di gestione delle autorizzazioni per l'esecutore del processo

Per sviluppare un job worker personalizzato per le tue azioni personalizzate CodePipeline, avrai bisogno di una strategia per l'integrazione della gestione degli utenti e delle autorizzazioni.

La strategia più semplice consiste nell'aggiungere l'infrastruttura necessaria per il job worker personalizzato creando istanze Amazon EC2 con un ruolo di istanza IAM, che consentono di scalare facilmente le risorse necessarie per l'integrazione. Puoi utilizzare l'integrazione integrata con AWS per semplificare l'interazione tra il tuo job worker personalizzato e. CodePipeline

Per configurare istanze Amazon EC2

1. Scopri di più su Amazon EC2 e determina se è la scelta giusta per la tua integrazione. Per informazioni, consulta Amazon EC2 - Virtual Server Hosting.
2. Inizia a creare le tue istanze Amazon EC2. Per informazioni, consulta Getting Started with Amazon EC2 Linux Instances.

Un'altra strategia da considerare consiste nell'utilizzare la federazione delle identità con IAM per integrare il sistema e le risorse esistenti del provider di identità. Questa strategia è particolarmente utile se disponi già di un provider delle identità aziendali o se le risorse sono già configurate per supportare gli utenti che utilizzano provider delle identità Web. La federazione delle identità consente di concedere un accesso sicuro alle AWS risorse CodePipeline, anche senza dover creare o gestire utenti IAM. Puoi utilizzare le caratteristiche e le policy per requisiti di sicurezza delle password e rotazione delle credenziali. Puoi utilizzare applicazioni di esempio come modelli per il tuo progetto.

Per configurare la federazione delle identità

1. Scopri di più sulla federazione delle identità IAM. Per informazioni, consulta Gestione della federazione.
2. Rivedi gli esempi negli Scenari per la concessione dell'accesso temporaneo per identificare lo scenario per l'accesso temporaneo più adatto alle esigenze dell'operazione personalizzata.

3. Rivedere gli esempi di codice della federazione delle identità rilevanti per l'infrastruttura, ad esempio:

   - Applicazione di esempio della federazione delle identità per un caso d'uso di Active Directory

4. Iniziare a configurare la federazione delle identità. Per informazioni, consulta la Guida per l'utente di Identity Providers and Federation in IAM.

Crea uno dei seguenti comandi da utilizzare in Job Worker per eseguire azioni personalizzate e job worker. Account AWS

Gli utenti necessitano dell'accesso programmatico se desiderano interagire con utenti AWS esterni a. AWS Management Console Il modo per concedere l'accesso programmatico dipende dal tipo di utente che accede. AWS

Per fornire agli utenti l'accesso programmatico, scegli una delle seguenti opzioni.

Quale utente necessita dell'accesso programmatico?	Per	Come
Identità della forza lavoro (Utenti gestiti nel centro identità IAM)	Utilizza credenziali temporane e per firmare le richieste programmatiche agli AWS CLI AWS SDK o alle API. AWS	Segui le istruzioni per l'interfa ccia che desideri utilizzare. • Per la AWS CLI, consulta Configurazione dell'uso AWS IAM Identity Center nella Guida AWS CLI per l'utente.AWS Command Line Interface • Per AWS SDK, strumenti e AWS API, consulta l'autenticazione IAM Identity Center nella Guida di riferimento agli AWS SDK e agli strumenti.
IAM	Utilizza credenziali temporane e per firmare le richieste	Segui le istruzioni in Uso delle credenziali temporanee con

Quale utente necessita dell'accesso programmatico?	Per	Come
	programmatiche agli SDK o alle API AWS CLI. AWS AWS	AWS risorse nella Guida per l'utente IAM.
IAM	(Non consigliato) Utilizza credenziali a lungo termine per firmare le richieste programmatiche agli AWS CLI AWS SDK o alle API. AWS	Segui le istruzioni per l'interfaccia che desideri utilizzare. • Per la AWS CLI, consulta Autenticazione tramite credenziali utente IAM nella Guida per l'utente.AWS Command Line Interface • Per gli AWS SDK e gli strumenti, consulta Autenticazione tramite credenziali a lungo termine nella Guida di riferimento agli SDK e agli AWS strumenti. • Per le AWS API, consulta Gestione delle chiavi di accesso per gli utenti IAM nella Guida per l'utente IAM.

Di seguito è riportato una policy di esempio che puoi creare per l'utilizzo con l'esecutore del processo personalizzato. Questa policy è intesa solo come un esempio ed è fornita senza modifiche.

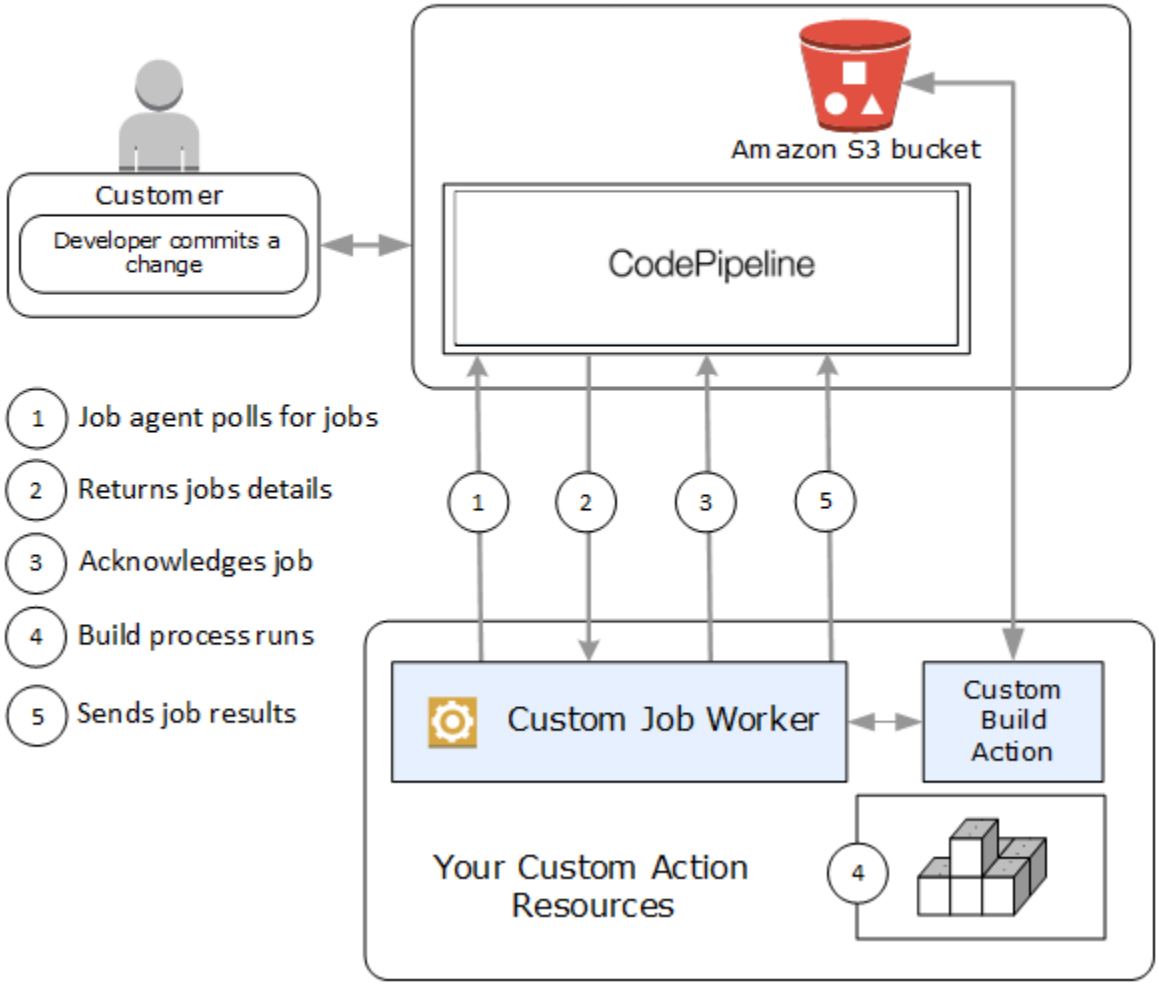
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codepipeline:PollForJobs",
        "codepipeline:AcknowledgeJob",
        "codepipeline:GetJobDetails",
```

```
        "codepipeline:PutJobSuccessResult",
        "codepipeline:PutJobFailureResult"
      ],
      "Resource": [
        "arn:aws:codepipeline:us-east-2::actionType:custom/Build/MyBuildProject/1/"
      ]
    }
  ]
}
```

> **Note**
>
> Prendi in considerazione l'utilizzo della policy gestita. `AWSCodePipelineCustomActionAccess`

Sviluppo di un esecutore del processo per l'operazione personalizzata

Dopo aver scelto la strategia di gestione delle autorizzazioni, dovresti considerare in che modo interagirà il tuo collaboratore. CodePipeline Il seguente diagramma di alto livello mostra il flusso di lavoro di un'azione personalizzata e di un job worker per un processo di compilazione.



1. Il tuo job worker effettua sondaggi CodePipeline per i lavori che utilizzano`PollForJobs`.
2. Quando una pipeline viene attivata da una modifica nella sua fase di origine (ad esempio, quando uno sviluppatore esegue il commit di una modifica), viene avviato il processo di rilascio automatico. Il processo continua fino alla fase in cui l'azione personalizzata è stata configurata. Quando raggiunge l'obiettivo dell'utente in questa fase, mette in CodePipeline coda un lavoro. Questo processo verrà visualizzato se l'esecutore del processo chiama nuovamente `PollForJobs` per ottenere lo stato. Acquisisce i dettagli del processo da `PollForJobs` e li restituisce all'esecutore del processo.
3. L'impiegato chiama `AcknowledgeJob` per inviare CodePipeline un riconoscimento di lavoro. CodePipeline restituisce un riconoscimento che indica che il lavoratore deve continuare il lavoro (`InProgress`) oppure, se più di un lavoratore sta effettuando sondaggi per le offerte di lavoro e un altro lavoratore ha già richiesto il lavoro, verrà restituita una risposta di

`InvalidNonceException` errore. Dopo il `InProgress` riconoscimento, CodePipeline attende la restituzione dei risultati.

4. Il job worker avvia l'azione personalizzata sulla revisione, quindi l'azione viene eseguita. Oltre a qualsiasi altra azione, l'azione personalizzata restituisce un risultato al job worker. Nell'esempio di un'azione di creazione personalizzata, l'azione estrae gli artefatti dal bucket Amazon S3, li crea e riporta gli artefatti creati correttamente nel bucket Amazon S3.
5. Durante l'esecuzione dell'azione, il job worker può effettuare la chiamata `PutJobSuccessResult` con un token di continuazione (la serializzazione dello stato del job generato dal job worker, ad esempio un identificatore di build in formato JSON o una chiave oggetto Amazon S3), oltre alle `ExternalExecutionId` informazioni che verranno utilizzate per compilare il link. `executionUrlTemplate` Ciò aggiornerà la visualizzazione della pipeline sulla console con un collegamento funzionante ai dettagli specifici dell'azione mentre è in corso. Anche se non richiesto, si tratta di una best practice perché consente agli utenti di visualizzare lo stato dell'operazione personalizzata mentre è in esecuzione.

   Una volta chiamato `PutJobSuccessResult`, il processo è considerato completato. Viene creato un nuovo lavoro CodePipeline che include il token di continuazione. Questo processo verrà visualizzato se l'esecutore del processo chiama nuovamente `PollForJobs`. Questo nuovo processo può essere utilizzato per controllare lo stato dell'operazione. Al termine dell'operazione, viene restituito un token di continuazione oppure non viene restituito un token di continuazione.

   > **Note**
   >
   > Se l'esecutore del processo esegue tutto il lavoro per un'operazione personalizzata, valuta se suddividere l'elaborazione dell'esecutore del processo in almeno due fasi. La prima fase stabilisce la pagina dei dettagli dell'operazione. Una volta creata la pagina dei dettagli, puoi serializzare lo stato dell'esecutore del processo e restituirlo come un token di continuazione, soggetto a limiti delle dimensioni (consulta Quote in AWS CodePipeline). Ad esempio, puoi scrivere lo stato dell'operazione nella stringa utilizzata come il token di continuazione. La seconda fase (e le fasi successive) dell'elaborazione dell'esecutore del processo eseguono il lavoro effettivo dell'operazione. Il passaggio finale restituisce l'esito positivo o negativo CodePipeline, senza alcun token di continuazione nel passaggio finale.

   Per ulteriori informazioni sull'utilizzo del token di continuazione, consulta le specifiche riportate `PutJobSuccessResult` nell'CodePipeline API Reference.

6. Una volta completata l'azione personalizzata, il job worker restituisce il risultato dell'azione personalizzata CodePipeline chiamando una delle due API:
   - `PutJobSuccessResult`senza un token di continuazione, che indica che l'azione personalizzata è stata eseguita correttamente
   - `PutJobFailureResult`, che indica che l'azione personalizzata non è stata eseguita correttamente

   A seconda del risultato, la pipeline continua nell'operazione successiva (esito positivo) o si interrompe (esito negativo).

Architettura dell'esecutore del processo personalizzato ed esempi

Dopo aver mappato il flusso di lavoro di alto livello, puoi creare l'esecutore del processo. Anche se le specifiche dell'operazione personalizzata determinano alla fine ciò che è richiesto per l'esecutore del processo, la maggior parte degli esecutori del processo includono le seguenti funzionalità:

- Ricerca di offerte di lavoro relative all' CodePipeline utilizzo`PollForJobs`.
- Riconoscimento delle offerte di lavoro e restituzione dei risultati all' CodePipeline utilizzo di `AcknowledgeJob`PutJobSuccessResult, e. `PutJobFailureResult`
- Recupero e/o inserimento di artefatti nel bucket Amazon S3 per la pipeline. Per scaricare elementi dal bucket Amazon S3, devi creare un client Amazon S3 che utilizzi la firma Signature Version 4 (Sig V4). Sig V4 è richiesto per. AWS KMS

  Per caricare artefatti nel bucket Amazon S3, devi inoltre configurare la richiesta Amazon S3 per utilizzare la crittografia. `PutObject` Attualmente solo AWS Key Management Service (AWS KMS) è supportato per la crittografia. AWS KMS usi AWS KMS keys. Per sapere se utilizzare una chiave gestita dal cliente Chiave gestita da AWS o una chiave gestita dal cliente per caricare gli artefatti, il job worker personalizzato deve esaminare i dati del lavoro e verificare la proprietà della chiave di crittografia. Se la proprietà è impostata, è necessario utilizzare l'ID della chiave gestita dal cliente durante la configurazione. AWS KMS Se la proprietà della chiave è nulla, si utilizza la. Chiave gestita da AWS CodePipeline utilizza il, Chiave gestita da AWS a meno che non sia configurato diversamente.

  Per un esempio che mostra come creare i AWS KMS parametri in Java o.NET, consulta Specificare AWS Key Management Service in Amazon S3 l'uso AWS degli SDK. Per ulteriori informazioni sul bucket Amazon S3 per CodePipeline, consulta. CodePipeline concetti

Un esempio più complesso di job worker personalizzato è disponibile su. GitHub Questo esempio è open source e viene fornito senza alcuna modifica.

- Job Worker di esempio per CodePipeline: scarica l'esempio dal GitHub repository.

Aggiunta di un'operazione personalizzata a una pipeline

Dopo aver creato un job worker, puoi aggiungere un'azione personalizzata a una pipeline creandone una nuova e selezionandola quando usi la procedura guidata Crea pipeline, modificando una pipeline esistente e aggiungendo l'azione personalizzata oppure utilizzando gli SDK o le AWS CLI API.

> **Note**
>
> Nella procedura guidata Create Pipeline (Crea pipeline), puoi creare una pipeline che include un'operazione personalizzata se si tratta di un'operazione di compilazione o di distribuzione . Se l'operazione personalizzata appartiene alla categoria di test, occorre aggiungerla modificando una pipeline esistente.

Argomenti

- Aggiunta di un'operazione personalizzata a una pipeline esistente (CLI)

Aggiunta di un'operazione personalizzata a una pipeline esistente (CLI)

Puoi utilizzare il AWS CLI per aggiungere un'azione personalizzata a una pipeline esistente.

1. Apri una sessione terminale (Linux, macOS o Unix) o un prompt dei comandi (Windows) ed esegui il get-pipeline comando per copiare la struttura della pipeline che desideri modificare in un file JSON. Ad esempio, per una pipeline denominata **`MyFirstPipeline`**, digitare il comando seguente:

   ```
   aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
   ```

   Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.
2. Aprire il file JSON in qualsiasi editor di testo e modificare la struttura del file per aggiungere l'operazione personalizzata a una fase esistente.

> **Note**
>
> Se si desidera eseguire l'operazione in parallelo con un'altra operazione in tale fase, assicurarsi di assegnargli lo stesso valore `runOrder` di tale operazione.

Ad esempio, per modificare la struttura di una pipeline per aggiungere una fase denominata Compila e per aggiungere un'operazione di compilazione personalizzata a tale fase, modificare il file JSON per aggiungere una fase Compila prima di una fase di distribuzione come segue:

```
,
    {
      "name": "MyBuildStage",
      "actions":  [
             {
               "inputArtifacts": [
               {
                  "name": "MyApp"
                }
                  ],
                   "name": "MyBuildCustomAction",
                   "actionTypeId": {
                       "category": "Build",
                       "owner": "Custom",
                       "version": "1",
                       "provider": "My-Build-Provider-Name"
                   },
                   "outputArtifacts": [
                       {
                         "name": "MyBuiltApp"
                       }
                   ],
                   "configuration": {
                       "ProjectName": "MyBuildProject"
                   },
                   "runOrder": 1
               }
           ]
       },
       {
           "name": "Staging",
```

```
            "actions": [
                    {
                        "inputArtifacts": [
                            {
                                "name": "MyBuiltApp"
                            }
                        ],
                        "name": "Deploy-CodeDeploy-Application",
                        "actionTypeId": {
                            "category": "Deploy",
                            "owner": "AWS",
                            "version": "1",
                            "provider": "CodeDeploy"
                        },
                        "outputArtifacts": [],
                        "configuration": {
                            "ApplicationName": "CodePipelineDemoApplication",
                            "DeploymentGroupName": "CodePipelineDemoFleet"
                        },
                        "runOrder": 1
                    }
                ]
            }
    ]
}
```

3. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline, in modo analogo al seguente:

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

```
aws codepipeline update-pipeline --cli-input-json file://pipeline.json
```

Questo comando restituisce l'intera struttura della pipeline modificata.

4. Apri la CodePipeline console e scegli il nome della pipeline che hai appena modificato.

La pipeline mostra le modifiche. La prossima volta che si modifica la sorgente originale, la pipeline eseguirà tale versione attraverso la struttura modificata delle pipeline stessa.

Etichetta un'azione personalizzata in CodePipeline

I tag sono coppie chiave-valore associate AWS alle risorse. Puoi utilizzare la console o la CLI per applicare tag alle tue azioni personalizzate in. CodePipeline Per informazioni sull'etichettatura CodePipeline delle risorse, sui casi d'uso, sui vincoli di chiavi e valori dei tag e sui tipi di risorse supportati, consulta. Assegnazione di tag alle risorse

È possibile aggiungere, eliminare e aggiornare i valori dei tag in un'operazione personalizzata. È possibile aggiungere fino a un massimo di 50 tag per ciascuna operazione personalizzata.

Argomenti

- Aggiunta di tag a un'operazione personalizzata
- Visualizzazione di tag per un'operazione personalizzata
- Modifica di tag per un'operazione personalizzata
- Rimozione di tag da un'operazione personalizzata

Aggiunta di tag a un'operazione personalizzata

Segui questi passaggi per utilizzare per aggiungere un tag AWS CLI a un'azione personalizzata. Per aggiungere un tag a un'operazione personalizzata al momento della creazione, consulta Creare e aggiungere un'azione personalizzata in CodePipeline.

In queste fasi, si assume che sia già installata una versione recente della AWS CLI o che sia aggiornata alla versione corrente. Per ulteriori informazioni, consulta l'argomento relativo all'installazione di AWS Command Line Interface.

Al terminale o alla riga di comando, eseguir il comando tag-resource, specificando l'ARN (Amazon Resource Name) dell'operazione personalizzata in cui aggiungere i tag e la chiave e il valore del tag che desideri aggiungere. È possibile aggiungere più tag a un'operazione personalizzata. Ad esempio, per etichettare un'azione personalizzata con due tag, una chiave di tag denominata *`TestActionType`*con il valore del *`UnitTest`*tag e una chiave di tag denominata *`ApplicationName`*con il valore del tag di *`MyApplication`*:

```
aws codepipeline tag-resource --resource-arn arn:aws:codepipeline:us-west-2:account-id:actiontype:Owner/Category/Provider/Version --tags key=TestActionType,value=UnitTest
 key=ApplicationName,value=MyApplication
```

In caso di successo, questo comando non restituisce alcun risultato.

Visualizzazione di tag per un'operazione personalizzata

Segui questi passaggi per utilizzare AWS CLI per visualizzare i AWS tag per un'azione personalizzata. Se non sono stati aggiunti tag, l'elenco restituito è vuoto.

Dal terminale o dalla riga di comando, esegui il comando list-tags-for-resource. Ad esempio, per visualizzare un elenco di valori di chiavi e di tag per un'operazione personalizzata con l'ARN `arn:aws:codepipeline:us-west-2:account-id:actiontype:Owner/Category/Provider/Version`:

```
aws codepipeline list-tags-for-resource --resource-arn arn:aws:codepipeline:us-west-2:account-id:actiontype:Owner/Category/Provider/Version
```

Se il comando viene eseguito correttamente, restituisce informazioni simili alle seguenti:

```
{
    "tags": {
        "TestActionType": "UnitTest",
        "ApplicationName": "MyApplication"
    }
}
```

Modifica di tag per un'operazione personalizzata

Segui questi passaggi per utilizzare AWS CLI per modificare un tag per un'azione personalizzata. È possibile modificare il valore di una chiave esistente o aggiungere un'altra chiave. È anche possibile rimuovere i tag da un'operazione personalizzata, come illustrato nella sezione successiva.

Al terminale o nella riga di comando, esegui il comando tag-resource specificando l'ARN (Amazon Resource Name) dell'operazione personalizzata in cui desideri aggiornare un tag e specificare la chiave e il valore di tag:

```
aws codepipeline tag-resource --resource-arn arn:aws:codepipeline:us-
west-2:account-id:actiontype:Owner/Category/Provider/Version --tags
 key=TestActionType,value=IntegrationTest
```

Rimozione di tag da un'operazione personalizzata

Segui questi passaggi per utilizzare AWS CLI per rimuovere un tag da un'azione personalizzata. Quando rimuovi i tag dalla risorsa associata, questi vengono eliminati.

> **Note**
>
> Se si elimina un'operazione personalizzata, tutte le associazioni di tag vengono rimosse dall'operazione personalizzata eliminata. Non è necessario rimuovere i tag prima di eliminare un'operazione personalizzata.

Al terminale o nella riga di comando, esegui il comando untag-resource specificando l'ARN dell'operazione personalizzata in cui desideri rimuovere i tag e la chiave di tag del tag che desideri rimuovere. Ad esempio, per rimuovere un tag su un'azione personalizzata con la chiave tag *TestActionType*:

```
aws codepipeline untag-resource --resource-arn arn:aws:codepipeline:us-west-2:account-
id:actiontype:Owner/Category/Provider/Version --tag-keys TestActionType
```

In caso di successo, questo comando non restituisce alcun risultato. Per verificare i tag associati all'operazione personalizzata, esegui il comando list-tags-for-resource.

Invoca una AWS Lambda funzione in una pipeline in CodePipeline

AWS Lambda è un servizio di elaborazione che consente di eseguire del codice senza la necessità di effettuare il provisioning o la gestione dei server. Puoi creare funzioni Lambda e aggiungerle come azioni nelle tue pipeline. Poiché Lambda consente di scrivere funzioni per eseguire quasi tutte le attività, è possibile personalizzare il funzionamento della pipeline.

> **Important**
>
> Non registrate l'evento JSON CodePipeline inviato a Lambda perché ciò può comportare la registrazione delle credenziali utente nei log. CloudWatch Il CodePipeline ruolo

utilizza un evento JSON per passare credenziali temporanee a Lambda sul campo. `artifactCredentials` Per un evento di esempio, consultare Evento JSON di esempio.

Ecco alcuni modi in cui le funzioni Lambda possono essere utilizzate nelle pipeline:

- Per creare risorse su richiesta in una fase di una pipeline, utilizzarle AWS CloudFormation ed eliminarle in un'altra fase.
- Distribuire versioni delle applicazioni senza tempi di inattività con una funzione Lambda che scambia i valori CNAME. AWS Elastic Beanstalk
- Da distribuire su istanze Docker di Amazon ECS.
- Per eseguire il backup delle risorse prima della compilazione o della distribuzione mediante la creazione di uno snapshot AMI.
- Per aggiungere l'integrazione con i prodotti di terze parti alla pipeline, ad esempio l'invio di messaggi a un client IRC.

> **Note**
>
> La creazione e l'esecuzione di funzioni Lambda potrebbero comportare addebiti sul tuo AWS account. Per ulteriori informazioni, consulta la sezione Prezzi di .

Questo argomento presuppone che tu conosca AWS CodePipeline AWS Lambda e sappia come creare pipeline, funzioni e le politiche e i ruoli IAM da cui dipendono. Questo argomento illustra come:

- Crea una funzione Lambda che verifica se una pagina Web è stata distribuita correttamente.
- Configura i ruoli di esecuzione CodePipeline e Lambda e le autorizzazioni necessarie per eseguire la funzione come parte della pipeline.
- Modifica una pipeline per aggiungere la funzione Lambda come azione.
- Testare l'operazione rilasciando manualmente una modifica.

> **Note**
>
> Quando si utilizza Cross-region Lambda invoke action CodePipeline in, lo stato dell'esecuzione lambda che utilizza PutJobSuccessResultPutJobFailureResulte deve essere

inviato alla AWS regione in cui è presente la funzione Lambda e non alla regione in cui esiste. CodePipeline

Questo argomento include funzioni di esempio per dimostrare la flessibilità dell'utilizzo delle funzioni Lambda in: CodePipeline

- Basic Lambda function
  - Creazione di una funzione Lambda di base da utilizzare con. CodePipeline
  - Se si restituisce un esito positivo o negativo, viene visualizzato il link Dettagli relativo all'azione. CodePipeline
- Funzione Python di esempio che utilizza un modello AWS CloudFormation
  - Utilizzo di parametri utente codificati in formato JSON per inoltrare più valori di configurazione alla funzione (`get_user_params`).
  - Interazione con artefatti .zip in un bucket dedicato agli artefatti (`get_template`).
  - Utilizzo di un token di prosecuzione per monitorare un processo asincrono dall'esecuzione prolungata (`continue_job_later`). Ciò consente all'azione di continuare e alla funzione di avere successo anche se supera un tempo di esecuzione di quindici minuti (un limite in Lambda).

Ogni funzione di esempio include informazioni sulle autorizzazioni che è necessario aggiungere al ruolo. Per informazioni sui limiti in AWS Lambda, consulta Limits nella Developer Guide.AWS Lambda

> **Important**
>
> Il codice di esempio, i ruoli e le policy inclusi in questo argomento sono solo esempi e vengono forniti senza alcuna modifica.

Argomenti

- Fase 1: creazione di una pipeline
- Fase 2: Creare la funzione Lambda
- Passaggio 3: aggiungere la funzione Lambda a una pipeline nella console CodePipeline
- Fase 4: testare la pipeline con la funzione Lambda
- Fase 5: fasi successive

- Evento JSON di esempio
- Funzioni di esempio aggiuntive

Fase 1: creazione di una pipeline

In questo passaggio, crei una pipeline a cui successivamente aggiungerai la funzione Lambda. Questa è la stessa pipeline creata in CodePipeline tutorial. Se la pipeline è ancora configurata per il tuo account e si trova nella stessa regione in cui prevedi di creare la funzione Lambda, puoi saltare questo passaggio.

Per creare la pipeline

1. Segui i primi tre passaggi Tutorial: creazione di una semplice pipeline (bucket S3) per creare un bucket Amazon S3, CodeDeploy risorse e una pipeline in due fasi. Scegli l'opzione Amazon Linux per i tuoi tipi di istanze. Puoi usare il nome che desideri per la pipeline, ad eccezione dei passaggi descritti in questo argomento. MyLambdaTestPipeline
2. Nella pagina di stato della pipeline, nell' CodeDeploy azione, scegli Dettagli. Nella pagina dei dettagli di distribuzione del gruppo di distribuzione, scegliere un ID istanza dall'elenco.
3. Nella console Amazon EC2, nella scheda Dettagli dell'istanza, copia l'indirizzo IP in Indirizzo IPv4 pubblico (ad esempio,). **`192.0.2.4`** Utilizzare questo indirizzo come la destinazione della funzione in AWS Lambda.

> **Note**
>
> La politica del ruolo di servizio predefinita per CodePipeline include le autorizzazioni Lambda necessarie per richiamare la funzione. Tuttavia, se hai modificato il ruolo di default del servizio o ne hai selezionato un altro, verifica che la policy del ruolo includa le autorizzazioni `lambda:InvokeFunction` e `lambda:ListFunctions`. Altrimenti, le pipeline che includono azioni Lambda falliranno.

Fase 2: Creare la funzione Lambda

In questo passaggio, crei una funzione Lambda che effettua una richiesta HTTP e verifica la presenza di una riga di testo su una pagina Web. Come parte di questo passaggio, devi anche creare

una policy IAM e un ruolo di esecuzione Lambda. Per ulteriori informazioni, consulta Modello di autorizzazioni nella Guida per gli sviluppatori di AWS Lambda .

Per creare il ruolo di esecuzione

1. Accedi AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Seleziona Policy, quindi scegli Create Policy (Crea policy). Scegli la scheda JSON e quindi incolla la seguente policy nel campo.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:*"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:*:*:*"
    },
    {
      "Action": [
        "codepipeline:PutJobSuccessResult",
        "codepipeline:PutJobFailureResult"
        ],
        "Effect": "Allow",
        "Resource": "*"
     }
  ]
}
```

3. Scegli Verifica policy.
4. Nella pagina Review policy (Rivedi policy), in Name (Nome), digita un nome per la policy (ad esempio, **`CodePipelineLambdaExecPolicy`**). In Description (Descrizione), immetti **`Enables Lambda to execute code`**.

   Scegliere Create Policy (Crea policy).

> **Note**
>
> Queste sono le autorizzazioni minime richieste per l'interazione di una funzione Lambda con Amazon CodePipeline . CloudWatch Se desideri espandere questo criterio per consentire funzioni che interagiscono con altre AWS risorse, devi modificare questo criterio per consentire le azioni richieste da tali funzioni Lambda.

5. Nella pagina del pannello di controllo delle policy, scegli Roles (Ruoli) e quindi seleziona Create role (Crea ruolo).
6. Nella pagina Crea ruolo, scegli Servizio AWS. Scegli Lambda, quindi seleziona Next: Permissions (Successivo: Autorizzazioni).
7. Nella pagina Allega criteri di autorizzazione, seleziona la casella di controllo accanto a CodePipelineLambdaExecPolicy, quindi scegli Avanti: Tag. Scegli Prossimo: Rivedi.
8. Nella pagina Review (Rivedi), in Role name (Nome ruolo), immetti il nome e scegli Create role (Crea ruolo).

Per creare la funzione Lambda di esempio da utilizzare con CodePipeline

1. Accedi AWS Management Console e apri la AWS Lambda console all'indirizzo https://console.aws.amazon.com/lambda/.
2. Nella pagina Functions (Funzioni), scegli Create function (Crea funzione).

> **Note**
>
> Se vedi una pagina di benvenuto anziché la pagina Lambda, scegli Inizia subito.

3. Nella pagina Create function (Crea funzione), scegliere Author from scratch (Crea da zero). In Nome funzione, inserisci un nome per la tua funzione Lambda (ad esempio,**`MyLambdaFunctionForAWSCodePipeline`**). In Runtime, scegli Node.js 20.x.
4. In Role (Ruolo) seleziona Choose an existing role (Scegli un ruolo esistente). In Existing role (Ruolo esistente), scegli il ruolo e quindi seleziona Create function (Crea funzione).

   Viene visualizzata la pagina dei dettagli per la funzione creata.
5. Copia il codice seguente nella casella Function code (Codice funzione):

> **Note**
>
> L'oggetto evento, sotto la CodePipeline chiave.job, contiene i dettagli del lavoro. Per un esempio completo del CodePipeline ritorno dell'evento JSON a Lambda, vedi. Evento JSON di esempio

```
import { CodePipelineClient, PutJobSuccessResultCommand,
 PutJobFailureResultCommand } from "@aws-sdk/client-codepipeline";
import http from 'http';
import assert from 'assert';

export const handler = (event, context) => {

    const codepipeline = new CodePipelineClient();

    // Retrieve the Job ID from the Lambda action
    const jobId = event["CodePipeline.job"].id;

    // Retrieve the value of UserParameters from the Lambda action configuration in
 CodePipeline, in this case a URL which will be
    // health checked by this function.
    const url =
 event["CodePipeline.job"].data.actionConfiguration.configuration.UserParameters;

    // Notify CodePipeline of a successful job
    const putJobSuccess = async function(message) {
        const command = new PutJobSuccessResultCommand({
            jobId: jobId
        });
         try {
             await codepipeline.send(command);
             context.succeed(message);
         } catch (err) {
             context.fail(err);
         }
    };

    // Notify CodePipeline of a failed job
    const putJobFailure = async function(message) {
        const command = new PutJobFailureResultCommand({
```

```
            jobId: jobId,
            failureDetails: {
                message: JSON.stringify(message),
                type: 'JobFailed',
                externalExecutionId: context.awsRequestId
            }
        });
        await codepipeline.send(command);
        context.fail(message);
    };

    // Validate the URL passed in UserParameters
    if(!url || url.indexOf('http://') === -1) {
        putJobFailure('The UserParameters field must contain a valid URL address to
test, including http:// or https://');
        return;
    }

    // Helper function to make a HTTP GET request to the page.
    // The helper will test the response and succeed or fail the job accordingly
    const getPage = function(url, callback) {
        var pageObject = {
            body: '',
            statusCode: 0,
            contains: function(search) {
                return this.body.indexOf(search) > -1;
            }
        };
        http.get(url, function(response) {
            pageObject.body = '';
            pageObject.statusCode = response.statusCode;

            response.on('data', function (chunk) {
                pageObject.body += chunk;
            });

            response.on('end', function () {
                callback(pageObject);
            });

            response.resume();
        }).on('error', function(error) {
            // Fail the job if our request failed
            putJobFailure(error);
```

```
        });
    };

    getPage(url, function(returnedPage) {
        try {
            // Check if the HTTP response has a 200 status
            assert(returnedPage.statusCode === 200);
            // Check if the page contains the text "Congratulations"
            // You can change this to check for different text, or add other tests
 as required
            assert(returnedPage.contains('Congratulations'));

            // Succeed the job
            putJobSuccess("Tests passed.");
        } catch (ex) {
            // If any of the assertions failed then fail the job
            putJobFailure(ex);
        }
    });
};
```

6. Non modificare i valori predefiniti per Handler e Role (Ruolo), **`CodePipelineLambdaExecRole`**.
7. In Basic settings (Impostazioni di base), per Timeout (Timeout), immetti **`20`** secondi.
8. Selezionare Salva.

Passaggio 3: aggiungere la funzione Lambda a una pipeline nella console CodePipeline

In questo passaggio, aggiungi una nuova fase alla pipeline, quindi aggiungi un'azione Lambda che richiama la tua funzione in quella fase.

Per aggiungere una fase

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Nella pagina Welcome (Benvenuto), scegliere la pipeline creata.
3. Nella pagina di visualizzazione della pipeline, scegliere Edit (Modifica).

4. Nella pagina Modifica, scegli + Aggiungi fase per aggiungere una fase dopo la fase di distribuzione con l' CodeDeploy azione. Immettere un nome per la fase (ad esempio **`LambdaStage`**), quindi scegliere Add stage (Aggiungi fase).

   > **Note**
   >
   > Puoi anche scegliere di aggiungere l'azione Lambda a una fase esistente. A scopo dimostrativo, stiamo aggiungendo la funzione Lambda come unica azione in una fase per consentirti di visualizzarne facilmente l'avanzamento man mano che gli artefatti avanzano attraverso una pipeline.

5. Scegliere + Add action group (+ Aggiungi gruppo di operazioni). In Modifica azione, in Nome azione, inserisci un nome per l'azione Lambda (ad esempio,**`MyLambdaAction`**). Alla voce Provider, scegliere AWS Lambda. In Nome funzione, scegli o inserisci il nome della tua funzione Lambda (ad esempio,**`MyLambdaFunctionForAWSCodePipeline`**). In Parametri utente, specifica l'indirizzo IP per l'istanza Amazon EC2 che hai copiato in precedenza (ad esempio,**`http://`*`192.0.2.4`***), quindi scegli Fine.

   > **Note**
   >
   > Questa sezione utilizza un indirizzo IP, ma in uno scenario reale è possibile fornire invece il nome del proprio sito web registrato (ad esempio **`http://`*`www.example.com`***). Per ulteriori informazioni sui dati e sui gestori degli eventi in AWS Lambda, consulta Programming Model nella Developer Guide.AWS Lambda

6. Nella pagina Edit action (Modifica operazione), scegli Save (Salva).

Fase 4: testare la pipeline con la funzione Lambda

Per testare la funzione, rilasciare la modifica più recente tramite la pipeline.

Per utilizzare la console per eseguire la versione più recente di un artefatto attraverso una pipeline

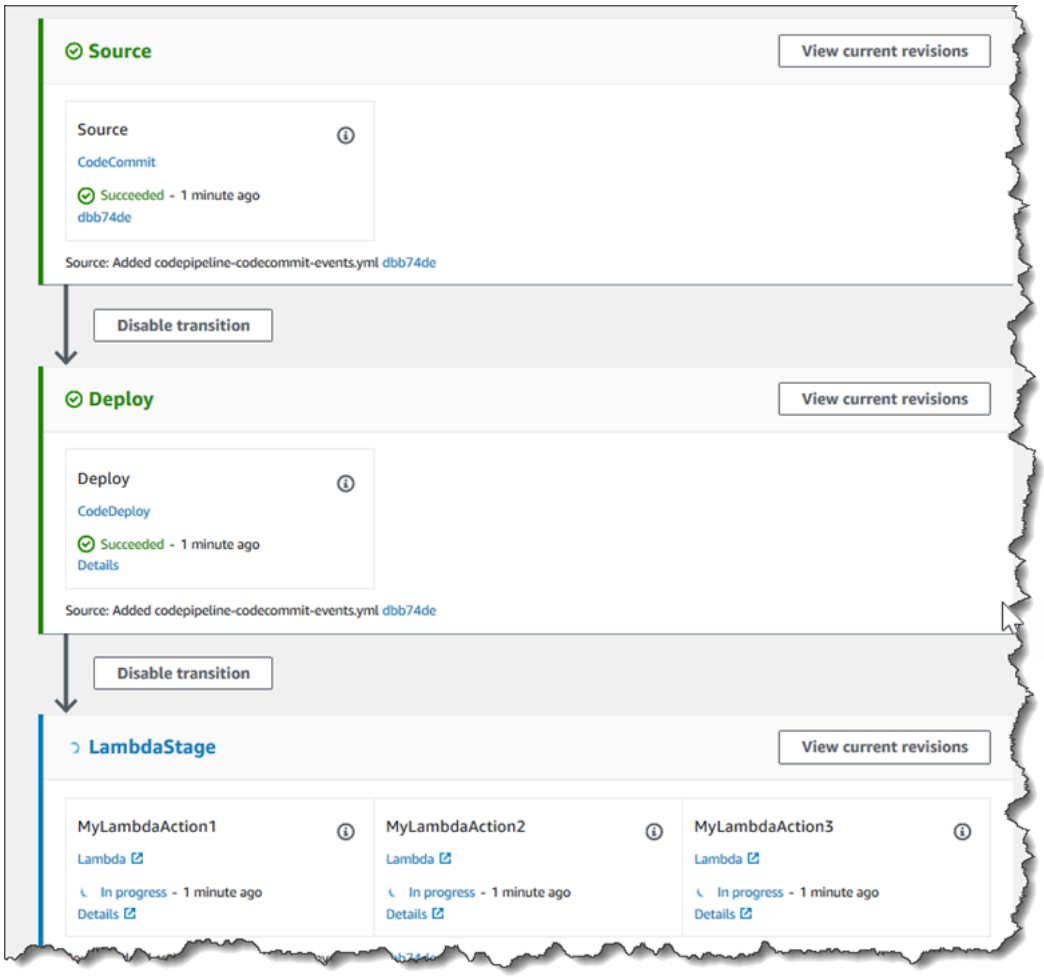
1. Nella pagina dei dettagli della pipeline, scegli Release change. In questo modo viene eseguita la revisione più recente disponibile in ogni percorso di origine specificato in un'operazione origine tramite la pipeline.

2. Quando l'azione Lambda è completa, scegli il link Dettagli per visualizzare il flusso di log della funzione in Amazon CloudWatch, inclusa la durata fatturata dell'evento. Se la funzione non funziona, il CloudWatch log fornisce informazioni sulla causa.

Fase 5: fasi successive

Ora che hai creato con successo una funzione Lambda e l'hai aggiunta come azione in una pipeline, puoi provare quanto segue:

- Aggiungi altre azioni Lambda al tuo stage per controllare altre pagine Web.
- Modifica la funzione Lambda per verificare la presenza di una stringa di testo diversa.
- Esplora le funzioni Lambda e crea e aggiungi le tue funzioni Lambda alle pipeline.



Dopo aver finito di sperimentare la funzione Lambda, valuta la possibilità di rimuoverla dalla pipeline, eliminarla AWS Lambda da ed eliminare il ruolo da IAM per evitare possibili addebiti. Per ulteriori informazioni, consulta Modificare una tubazione in CodePipeline, Eliminare una tubazione in CodePipeline ed Eliminazione dei ruoli o dei profili delle istanze.

Evento JSON di esempio

L'esempio seguente mostra un esempio di evento JSON inviato a CodePipeline Lambda da. La struttura dell'evento è simile alla risposta a `GetJobDetails API`, ma senza i tipi di dati `actionTypeId` e `pipelineContext`. Nell'evento in formato JSON e nella risposta all'API `GetJobDetails` sono inclusi due dettagli di configurazione dell'operazione, `FunctionName` e `UserParameters`. I valori in *`corsivo rosso`* sono esempi o spiegazioni, non valori reali.

```
{
    "CodePipeline.job": {
        "id": "11111111-abcd-1111-abcd-111111abcdef",
        "accountId": "111111111111",
        "data": {
            "actionConfiguration": {
                "configuration": {
                    "FunctionName": "MyLambdaFunctionForAWSCodePipeline",
                    "UserParameters": "some-input-such-as-a-URL"
                }
            },
            "inputArtifacts": [
                {
                    "location": {
                        "s3Location": {
                            "bucketName": "the name of the bucket configured as the
 pipeline artifact store in Amazon S3, for example codepipeline-us-east-2-1234567890",
                            "objectKey": "the name of the application, for example
 CodePipelineDemoApplication.zip"
                        },
                        "type": "S3"
                    },
                    "revision": null,
                    "name": "ArtifactName"
                }
            ],
            "outputArtifacts": [],
            "artifactCredentials": {
                "secretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
```

```
                "sessionToken": "MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w
 0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZ
 WF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIw
 EAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5
 jb20wHhcNMTEwNDI1MjA0NTIxWhcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBh
 MCVVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBb
 WF6b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMx
 HzAdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wgZ8wDQYJKoZIhvcNAQE
 BBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ21uUSfwfEvySWtC2XADZ4nB+BLYgVI
 k60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9TrDHudUZg3qX4waLG5M43q7Wgc/MbQ
 ITxOUSQv7c7ugFFDzQGBzZswY6786m86gpEIbb3OhjZnzcvQAaRHhdlQWIMm2nr
 AgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4nUhVVxYUntneD9+h8Mg9q6q+auN
 KyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0FkbFFBjvSfpJIlJ00zbhNYS5f6Guo
 EDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTbNYiytVbZPQUQ5Yaxu2jXnimvw
 3rrszlaEXAMPLE=",
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE"
            },
            "continuationToken": "A continuation token if continuing job",
            "encryptionKey": {
              "id": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
              "type": "KMS"
            }
        }
    }
}
```

Funzioni di esempio aggiuntive

Le seguenti funzioni Lambda di esempio mostrano funzionalità aggiuntive che è possibile utilizzare per le pipeline. CodePipeline Per utilizzare queste funzioni, potrebbe essere necessario modificare la policy per il ruolo di esecuzione Lambda, come indicato nell'introduzione di ogni esempio.

Argomenti

- Funzione Python di esempio che utilizza un modello AWS CloudFormation

Funzione Python di esempio che utilizza un modello AWS CloudFormation

L'esempio seguente mostra una funzione che crea o aggiorna uno stack basato su un modello fornito AWS CloudFormation . Il modello crea un bucket Amazon S3. È a solo a scopo dimostrativo, per ridurre al minimo i costi. Teoricamente, bisognerebbe eliminare lo stack prima caricare qualsiasi

cosa sul bucket. Se carichi file nel bucket, non puoi eliminare il bucket quando elimini lo stack. Per eliminare il bucket deve prima eliminare tutto il suo contenuto.

Questo esempio di Python presuppone che tu disponga di una pipeline che utilizza un bucket Amazon S3 come azione di origine o che tu abbia accesso a un bucket Amazon S3 con versione che puoi usare con la pipeline. Crei il AWS CloudFormation modello, lo comprimi e lo carichi in quel bucket come file.zip. È necessario quindi aggiungere alla pipeline un'operazione sorgente che recupera tale file.zip dal bucket.

> **Note**
>
> Se Amazon S3 è il fornitore di origine per la tua pipeline, puoi comprimere il file o i file sorgente in un unico .zip e caricare il file.zip nel tuo bucket di origine. È inoltre possibile caricare un singolo file decompresso; tuttavia, le operazioni a valle che si aspettano un file con estensione .zip avranno esito negativo.

Questo esempio illustra:

- L'utilizzo di parametri utente codificati in formato JSON per inoltrare più valori di configurazione alla funzione (`get_user_params`).
- L'interazione con artefatti .zip in un bucket dedicato agli artefatti (`get_template`).
- L'utilizzo di un token di prosecuzione per monitorare un processo asincrono dall'esecuzione prolungata (`continue_job_later`). Ciò consente all'azione di continuare e alla funzione di avere successo anche se supera un tempo di esecuzione di quindici minuti (un limite in Lambda).

Per utilizzare questa funzione Lambda di esempio, la policy per il ruolo di esecuzione Lambda deve disporre di `Allow` autorizzazioni in Amazon AWS CloudFormation S3 e CodePipeline, come illustrato in questa politica di esempio:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "logs:*"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:logs:*:*:*"
```

```
        },
        {
            "Action": [
                "codepipeline:PutJobSuccessResult",
                "codepipeline:PutJobFailureResult"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "cloudformation:DescribeStacks",
                "cloudformation:CreateStack",
                "cloudformation:UpdateStack"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "s3:*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Per creare il AWS CloudFormation modello, apri un qualsiasi editor di testo semplice e copia e incolla il codice seguente:

```
{
  "AWSTemplateFormatVersion" : "2010-09-09",
  "Description" : "CloudFormation template which creates an S3 bucket",
  "Resources" : {
    "MySampleBucket" : {
      "Type" : "AWS::S3::Bucket",
      "Properties" : {
      }
    }
  },
  "Outputs" : {
    "BucketName" : {
```

```
      "Value" : { "Ref" : "MySampleBucket" },
      "Description" : "The name of the S3 bucket"
    }
  }
}
```

Salvare il contenuto come file JSON con il nome **`template.json`** in una cartella denominata **`template-package`**. Crea un file compresso (.zip) di questa directory e del file denominato **`template-package.zip`** e carica il file compresso in un bucket Amazon S3 con versione. Se si dispone già di un bucket configurato per la pipeline, è possibile utilizzarlo. Quindi, modificare la pipeline per aggiungere un'operazione sorgente che recupera il file .zip. Assegna un nome all'output per questa azione. *`MyTemplate`* Per ulteriori informazioni, consulta Modificare una tubazione in CodePipeline.

> **Note**
>
> La funzione Lambda di esempio prevede questi nomi di file e una struttura compressa. Tuttavia, è possibile sostituire questo esempio con un AWS CloudFormation modello personalizzato. Se utilizzi il tuo modello, assicurati di modificare la politica per il ruolo di esecuzione Lambda per consentire qualsiasi funzionalità aggiuntiva richiesta AWS CloudFormation dal modello.

Per aggiungere il codice seguente come funzione in Lambda

1. Apri la console Lambda e scegli Crea funzione.
2. Nella pagina Create function (Crea funzione), scegliere Author from scratch (Crea da zero). In Nome funzione, inserisci un nome per la tua funzione Lambda.
3. In Runtime, scegli Python 2.7.
4. In Scegli o crea un ruolo di esecuzione, seleziona Usa un ruolo esistente. In Existing role (Ruolo esistente), scegli il ruolo e quindi seleziona Create function (Crea funzione).

   Viene visualizzata la pagina dei dettagli per la funzione creata.
5. Copia il codice seguente nella casella Function code (Codice funzione):

   ```
   from __future__ import print_function
   from boto3.session import Session
   ```

```
import json
import urllib
import boto3
import zipfile
import tempfile
import botocore
import traceback

print('Loading function')

cf = boto3.client('cloudformation')
code_pipeline = boto3.client('codepipeline')

def find_artifact(artifacts, name):
    """Finds the artifact 'name' among the 'artifacts'

    Args:
        artifacts: The list of artifacts available to the function
        name: The artifact we wish to use
    Returns:
        The artifact dictionary found
    Raises:
        Exception: If no matching artifact is found

    """
    for artifact in artifacts:
        if artifact['name'] == name:
            return artifact

    raise Exception('Input artifact named "{0}" not found in event'.format(name))

def get_template(s3, artifact, file_in_zip):
    """Gets the template artifact

    Downloads the artifact from the S3 artifact store to a temporary file
    then extracts the zip and returns the file containing the CloudFormation
    template.

    Args:
        artifact: The artifact to download
        file_in_zip: The path to the file within the zip containing the template

    Returns:
        The CloudFormation template as a string
```

```
    Raises:
        Exception: Any exception thrown while downloading the artifact or unzipping
 it

    """
    tmp_file = tempfile.NamedTemporaryFile()
    bucket = artifact['location']['s3Location']['bucketName']
    key = artifact['location']['s3Location']['objectKey']

    with tempfile.NamedTemporaryFile() as tmp_file:
        s3.download_file(bucket, key, tmp_file.name)
        with zipfile.ZipFile(tmp_file.name, 'r') as zip:
            return zip.read(file_in_zip)

def update_stack(stack, template):
    """Start a CloudFormation stack update

    Args:
        stack: The stack to update
        template: The template to apply

    Returns:
        True if an update was started, false if there were no changes
        to the template since the last update.

    Raises:
        Exception: Any exception besides "No updates are to be performed."

    """
    try:
        cf.update_stack(StackName=stack, TemplateBody=template)
        return True

    except botocore.exceptions.ClientError as e:
        if e.response['Error']['Message'] == 'No updates are to be performed.':
            return False
        else:
            raise Exception('Error updating CloudFormation stack
 "{0}"'.format(stack), e)

def stack_exists(stack):
    """Check if a stack exists or not
```

```
    Args:
        stack: The stack to check

    Returns:
        True or False depending on whether the stack exists

    Raises:
        Any exceptions raised .describe_stacks() besides that
        the stack doesn't exist.

    """
    try:
        cf.describe_stacks(StackName=stack)
        return True
    except botocore.exceptions.ClientError as e:
        if "does not exist" in e.response['Error']['Message']:
            return False
        else:
            raise e

def create_stack(stack, template):
    """Starts a new CloudFormation stack creation

    Args:
        stack: The stack to be created
        template: The template for the stack to be created with

    Throws:
        Exception: Any exception thrown by .create_stack()
    """
    cf.create_stack(StackName=stack, TemplateBody=template)

def get_stack_status(stack):
    """Get the status of an existing CloudFormation stack

    Args:
        stack: The name of the stack to check

    Returns:
        The CloudFormation status string of the stack such as CREATE_COMPLETE

    Raises:
        Exception: Any exception thrown by .describe_stacks()
```

```
    """
    stack_description = cf.describe_stacks(StackName=stack)
    return stack_description['Stacks'][0]['StackStatus']

def put_job_success(job, message):
    """Notify CodePipeline of a successful job

    Args:
        job: The CodePipeline job ID
        message: A message to be logged relating to the job status

    Raises:
        Exception: Any exception thrown by .put_job_success_result()

    """
    print('Putting job success')
    print(message)
    code_pipeline.put_job_success_result(jobId=job)

def put_job_failure(job, message):
    """Notify CodePipeline of a failed job

    Args:
        job: The CodePipeline job ID
        message: A message to be logged relating to the job status

    Raises:
        Exception: Any exception thrown by .put_job_failure_result()

    """
    print('Putting job failure')
    print(message)
    code_pipeline.put_job_failure_result(jobId=job, failureDetails={'message':
 message, 'type': 'JobFailed'})

def continue_job_later(job, message):
    """Notify CodePipeline of a continuing job

    This will cause CodePipeline to invoke the function again with the
    supplied continuation token.

    Args:
        job: The JobID
        message: A message to be logged relating to the job status
```

```
        continuation_token: The continuation token

    Raises:
        Exception: Any exception thrown by .put_job_success_result()

    """

    # Use the continuation token to keep track of any job execution state
    # This data will be available when a new job is scheduled to continue the
 current execution
    continuation_token = json.dumps({'previous_job_id': job})

    print('Putting job continuation')
    print(message)
    code_pipeline.put_job_success_result(jobId=job,
 continuationToken=continuation_token)

def start_update_or_create(job_id, stack, template):
    """Starts the stack update or create process

    If the stack exists then update, otherwise create.

    Args:
        job_id: The ID of the CodePipeline job
        stack: The stack to create or update
        template: The template to create/update the stack with

    """
    if stack_exists(stack):
        status = get_stack_status(stack)
        if status not in ['CREATE_COMPLETE', 'ROLLBACK_COMPLETE',
 'UPDATE_COMPLETE']:
            # If the CloudFormation stack is not in a state where
            # it can be updated again then fail the job right away.
            put_job_failure(job_id, 'Stack cannot be updated when status is: ' +
 status)
            return

        were_updates = update_stack(stack, template)

        if were_updates:
            # If there were updates then continue the job so it can monitor
            # the progress of the update.
            continue_job_later(job_id, 'Stack update started')
```

```
        else:
            # If there were no updates then succeed the job immediately
            put_job_success(job_id, 'There were no stack updates')
    else:
        # If the stack doesn't already exist then create it instead
        # of updating it.
        create_stack(stack, template)
        # Continue the job so the pipeline will wait for the CloudFormation
        # stack to be created.
        continue_job_later(job_id, 'Stack create started')

def check_stack_update_status(job_id, stack):
    """Monitor an already-running CloudFormation update/create

    Succeeds, fails or continues the job depending on the stack status.

    Args:
        job_id: The CodePipeline job ID
        stack: The stack to monitor

    """
    status = get_stack_status(stack)
    if status in ['UPDATE_COMPLETE', 'CREATE_COMPLETE']:
        # If the update/create finished successfully then
        # succeed the job and don't continue.
        put_job_success(job_id, 'Stack update complete')

    elif status in ['UPDATE_IN_PROGRESS', 'UPDATE_ROLLBACK_IN_PROGRESS',
    'UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS', 'CREATE_IN_PROGRESS',
    'ROLLBACK_IN_PROGRESS', 'UPDATE_COMPLETE_CLEANUP_IN_PROGRESS']:
        # If the job isn't finished yet then continue it
        continue_job_later(job_id, 'Stack update still in progress')

    else:
        # If the Stack is a state which isn't "in progress" or "complete"
        # then the stack update/create has failed so end the job with
        # a failed result.
        put_job_failure(job_id, 'Update failed: ' + status)

def get_user_params(job_data):
    """Decodes the JSON user parameters and validates the required properties.

    Args:
```

```
        job_data: The job data structure containing the UserParameters string which
 should be a valid JSON structure

    Returns:
        The JSON parameters decoded as a dictionary.

    Raises:
        Exception: The JSON can't be decoded or a property is missing.

    """
    try:
        # Get the user parameters which contain the stack, artifact and file
 settings
        user_parameters = job_data['actionConfiguration']['configuration']
['UserParameters']
        decoded_parameters = json.loads(user_parameters)

    except Exception as e:
        # We're expecting the user parameters to be encoded as JSON
        # so we can pass multiple values. If the JSON can't be decoded
        # then fail the job with a helpful message.
        raise Exception('UserParameters could not be decoded as JSON')

    if 'stack' not in decoded_parameters:
        # Validate that the stack is provided, otherwise fail the job
        # with a helpful message.
        raise Exception('Your UserParameters JSON must include the stack name')

    if 'artifact' not in decoded_parameters:
        # Validate that the artifact name is provided, otherwise fail the job
        # with a helpful message.
        raise Exception('Your UserParameters JSON must include the artifact name')

    if 'file' not in decoded_parameters:
        # Validate that the template file is provided, otherwise fail the job
        # with a helpful message.
        raise Exception('Your UserParameters JSON must include the template file
 name')

    return decoded_parameters

def setup_s3_client(job_data):
    """Creates an S3 client
```

```
    Uses the credentials passed in the event by CodePipeline. These
    credentials can be used to access the artifact bucket.

    Args:
        job_data: The job data structure

    Returns:
        An S3 client with the appropriate credentials

    """
    key_id = job_data['artifactCredentials']['accessKeyId']
    key_secret = job_data['artifactCredentials']['secretAccessKey']
    session_token = job_data['artifactCredentials']['sessionToken']

    session = Session(aws_access_key_id=key_id,
        aws_secret_access_key=key_secret,
        aws_session_token=session_token)
    return session.client('s3',
 config=botocore.client.Config(signature_version='s3v4'))

def lambda_handler(event, context):
    """The Lambda function handler

    If a continuing job then checks the CloudFormation stack status
    and updates the job accordingly.

    If a new job then kick of an update or creation of the target
    CloudFormation stack.

    Args:
        event: The event passed by Lambda
        context: The context passed by Lambda

    """
    try:
        # Extract the Job ID
        job_id = event['CodePipeline.job']['id']

        # Extract the Job Data
        job_data = event['CodePipeline.job']['data']

        # Extract the params
        params = get_user_params(job_data)
```

```
        # Get the list of artifacts passed to the function
        artifacts = job_data['inputArtifacts']

        stack = params['stack']
        artifact = params['artifact']
        template_file = params['file']

        if 'continuationToken' in job_data:
            # If we're continuing then the create/update has already been triggered
            # we just need to check if it has finished.
            check_stack_update_status(job_id, stack)
        else:
            # Get the artifact details
            artifact_data = find_artifact(artifacts, artifact)
            # Get S3 client to access artifact with
            s3 = setup_s3_client(job_data)
            # Get the JSON template file out of the artifact
            template = get_template(s3, artifact_data, template_file)
            # Kick off a stack update or create
            start_update_or_create(job_id, stack, template)

    except Exception as e:
        # If any other exceptions which we didn't expect are raised
        # then fail the job and log the exception message.
        print('Function failed due to exception.')
        print(e)
        traceback.print_exc()
        put_job_failure(job_id, 'Function exception: ' + str(e))

    print('Function complete.')
    return "Complete."
```

6. Lascia Handler al valore predefinito e lascia Role al nome selezionato o creato in precedenza,**`CodePipelineLambdaExecRole`**.
7. In Basic settings (Impostazioni di base), per Timeout, sostituisci l'impostazione predefinita di 3 secondi con **`20`**.
8. Selezionare Salva.
9. Dalla CodePipeline console, modifica la pipeline per aggiungere la funzione come azione in una fase della pipeline. Scegli Modifica per la fase della pipeline che desideri modificare e scegli Aggiungi gruppo di azioni. Nella pagina Modifica azione, in Nome azione, inserisci un nome per l'azione. In Action provider, scegli Lambda.

In Inserisci artefatti, scegli. `MyTemplate` In UserParameters, devi fornire una stringa JSON con tre parametri:

- Stack name (Nome stack)
- AWS CloudFormation nome del modello e percorso del file
- Artefatto di input

Utilizzare le parentesi graffe ({}) e separare i parametri con virgole. Ad esempio, per creare uno stack denominato, per una pipeline con l'elemento di input *`MyTestStack`*, in UserParameters, inserisci: {"stack»:» «*`MyTemplate`*, "file» :"template-package/template.json», "artifact»:» *`MyTestStack`*«}. *`MyTemplate`*

> **Note**
>
> Anche se avete specificato l'artefatto di input in, dovete specificare anche questo artefatto di input per l'azione in Input artifacts. UserParameters

10. Salva le modifiche alla pipeline, quindi rilascia manualmente una modifica per testare l'azione e la funzione Lambda.

Riprova un'azione fallita in una fase

È possibile riprovare una fase che ha avuto esito negativo senza dover eseguire nuovamente una pipeline dall'inizio. A tale scopo, riprovate le azioni non riuscite in una fase oppure riprovate tutte le azioni nella fase a partire dalla prima azione nella fase. Quando riprovi a eseguire le azioni non riuscite in una fase, tutte le azioni ancora in corso continuano a funzionare e le azioni fallite vengono nuovamente attivate. Quando riprovi una fase fallita dalla prima azione nella fase, non è possibile che nella fase siano in corso azioni. Prima di poter ritentare una fase, è necessario che tutte le azioni siano fallite oppure che alcune azioni siano fallite e altre abbiano avuto successo.

> **Important**
>
> Riprovare una fase non riuscita riprova tutte le azioni della fase dalla prima azione nella fase e riprovare le azioni fallite riprova tutte le azioni fallite nella fase. Ciò sovrascrive gli artefatti di output delle azioni precedentemente eseguite con successo nella stessa esecuzione.

Sebbene gli artefatti possano essere sostituiti, la cronologia di esecuzione delle azioni precedentemente riuscite viene comunque conservata.

Se si utilizza la console per visualizzare una pipeline, sullo stage viene visualizzato il pulsante Riprova fase o Riprova azioni fallite che può essere riprovato.

Se si utilizza la AWS CLI, è possibile utilizzare il get-pipeline-state comando per determinare se alcune azioni sono fallite.

Note

Nei seguenti casi, potresti non essere in grado di riprovare una fase:

- Tutte le azioni nella fase hanno avuto esito positivo, pertanto lo stato della fase non è fallito.
- La struttura complessiva della pipeline è cambiata dopo il fallimento della fase.
- È già in corso un altro tentativo di ripetizione nella fase.

Argomenti

Nuovo tentativo di operazioni non riuscite (console)

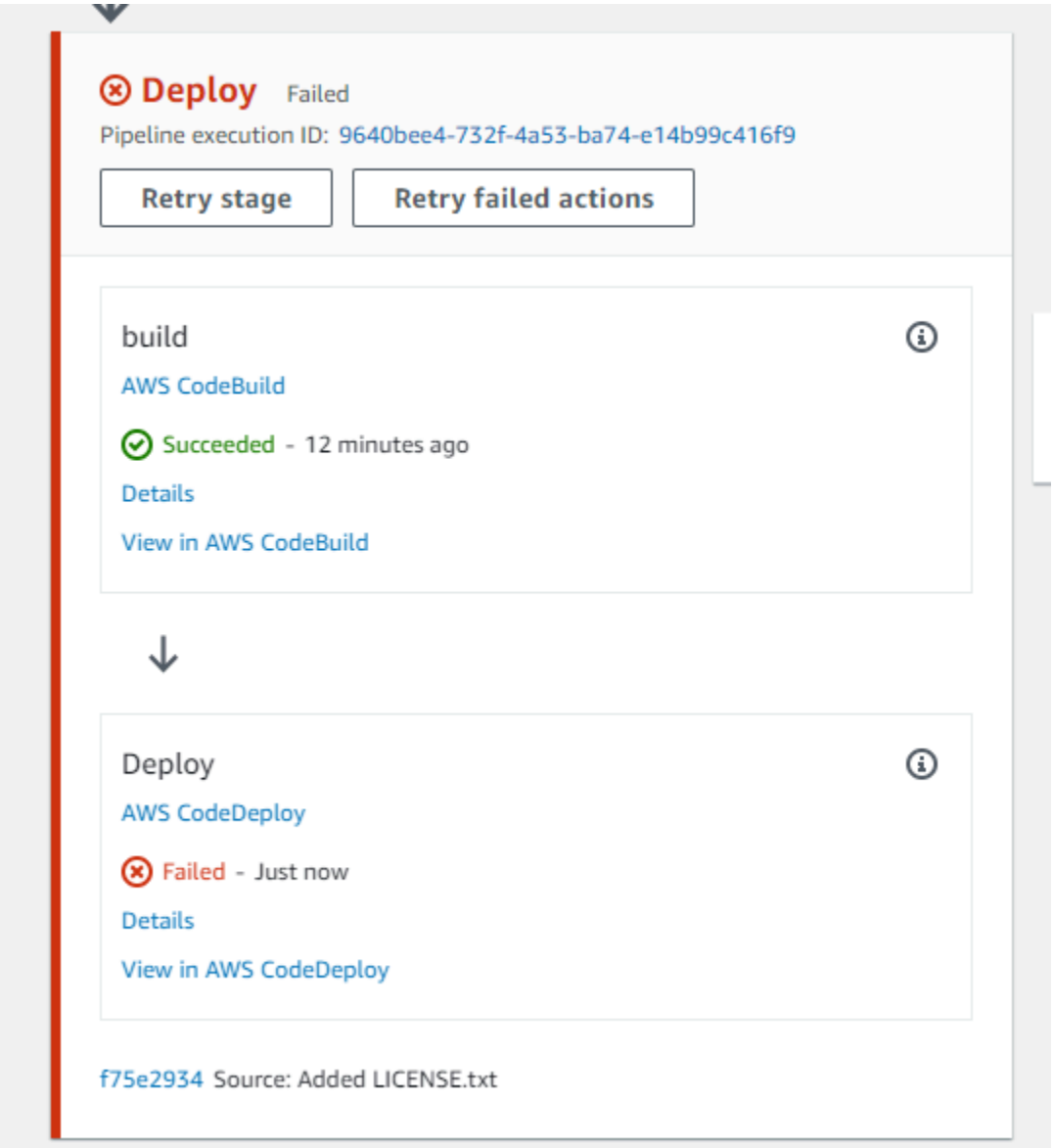
Per riprovare una fase fallita o azioni fallite in una fase: console

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegli il nome della pipeline.
3. Individuate la fase in cui si è verificata l'azione non riuscita, quindi scegliete una delle seguenti opzioni:

   - Per riprovare tutte le azioni nella fase, scegli Riprova fase.

- Per riprovare solo le azioni fallite nella fase, scegli Riprova le azioni non riuscite.



Se tutte le operazioni ripetute nella fase vengono completate, l'esecuzione della pipeline continua.

Nuovo tentativo di operazioni non riuscite (CLI)

Per riprovare una fase fallita o azioni non riuscite in una fase - CLI

Per utilizzare il comando AWS CLI per riprovare tutte le azioni o tutte le azioni non riuscite, esegui il retry-stage-execution comando con i seguenti parametri:

```
--pipeline-name <value>
--stage-name <value>
--pipeline-execution-id <value>
--retry-mode ALL_ACTIONS/FAILED_ACTIONS
```

> **Note**
>
> I valori per cui è possibile utilizzare `retry-mode` sono `FAILED_ACTIONS` e`ALL_ACTIONS`.

1. In un terminale (Linux, macOS o Unix) o nel prompt dei comandi (Windows), esegui il retry-stage-executioncomando, come illustrato nell'esempio seguente per una pipeline denominata. `MyPipeline`

   ```
   aws codepipeline retry-stage-execution --pipeline-name MyPipeline --stage-name
    Deploy --pipeline-execution-id b59babff-5f34-EXAMPLE --retry-mode FAILED_ACTIONS
   ```

   L'output restituisce l'ID di esecuzione:

   ```
   {
       "pipelineExecutionId": "b59babff-5f34-EXAMPLE"
   }
   ```

2. Puoi anche eseguire il comando con un file di input JSON. Crea innanzitutto un file JSON che identifichi la pipeline, la fase contenente le operazioni non riuscite e l'esecuzione della pipeline più recente in tale fase. Quindi, esegui il comando retry-stage-execution con il parametro `--cli-input-json`. Per recuperare i dettagli necessari per il file JSON, è più semplice utilizzare il comando get-pipeline-state.

   a. In un terminale (Linux, macOS o Unix) o nel prompt dei comandi (Windows), esegui il get-pipeline-statecomando su una pipeline. Ad esempio, per una pipeline denominata MyFirstPipeline, è necessario digitare qualcosa di simile al seguente:

      ```
      aws codepipeline get-pipeline-state --name MyFirstPipeline
      ```

      La risposta al comando include le informazioni sullo stato della pipeline per ogni fase. Nel seguente esempio, la risposta indica che una o più operazioni non sono riuscite nella fase Gestione temporanea:

```
{
    "updated": 1427245911.525,
    "created": 1427245911.525,
    "pipelineVersion": 1,
    "pipelineName": "MyFirstPipeline",
    "stageStates": [
        {
            "actionStates": [...],
            "stageName": "Source",
            "latestExecution": {
                "pipelineExecutionId": "9811f7cb-7cf7-SUCCESS",
                "status": "Succeeded"
            }
        },
        {
            "actionStates": [...],
            "stageName": "Staging",
            "latestExecution": {
                "pipelineExecutionId": "3137f7cb-7cf7-EXAMPLE",
                "status": "Failed"
            }
        }
    ]
}
```

b. In un editor di testo normale, creare un file in formato JSON in cui verranno registrate le informazioni riportate di seguito:

- Il nome della pipeline contenente le operazioni non riuscite.
- Il nome della fase contenente le operazioni non riuscite.
- L'ID dell'ultima esecuzione della pipeline nella fase
- La modalità di ripetizione.

MyFirstPipeline Nell'esempio precedente, il file avrebbe un aspetto simile al seguente:

```
{
    "pipelineName": "MyFirstPipeline",
    "stageName": "Staging",
    "pipelineExecutionId": "3137f7cb-7cf7-EXAMPLE",
    "retryMode": "FAILED_ACTIONS"
```

```
}
```

c. Salvare il file con un nome come **`retry-failed-actions.json`**.

d. Chiama il file creato quando esegui il comando retry-stage-execution. Per esempio:

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

```
aws codepipeline retry-stage-execution --cli-input-json file://retry-failed-
actions.json
```

e. Per visualizzare i risultati del nuovo tentativo, apri la CodePipeline console e scegli la pipeline che contiene le azioni fallite oppure usa nuovamente il get-pipeline-state comando. Per ulteriori informazioni, consulta Visualizza le pipeline e i dettagli in CodePipeline.

Gestisci le azioni di approvazione in CodePipeline

Inoltre AWS CodePipeline, puoi aggiungere un'azione di approvazione a una fase di una pipeline nel punto in cui desideri interrompere l'esecuzione della pipeline in modo che qualcuno con AWS Identity and Access Management le autorizzazioni richieste possa approvare o rifiutare l'azione.

Se l'operazione viene approvata, l'esecuzione della pipeline riprende. Se l'azione viene rifiutata, o se nessuno approva o rifiuta l'azione entro sette giorni dal momento in cui la pipeline ha raggiunto l'azione e si è interrotta, il risultato è lo stesso di un'azione fallita e l'esecuzione della pipeline non continua.

Puoi utilizzare le approvazioni manuali per questi motivi:

- Desideri che qualcuno esegua una revisione del codice o una revisione della gestione delle modifiche prima che venga consentita una revisione nella fase successiva di una pipeline.
- Desideri che qualcuno esegua il test di controllo di qualità manuale sulla versione più recente di un'applicazione o confermi l'integrità di un artefatto di compilazione, prima del rilascio.
- Desideri che qualcuno riveda il testo nuovo o aggiornato prima che venga pubblicato su un sito Web aziendale.

Opzioni di configurazione per le azioni di approvazione manuale in CodePipeline

CodePipeline fornisce tre opzioni di configurazione che è possibile utilizzare per informare gli approvatori dell'azione di approvazione.

Pubblica notifiche di approvazione Puoi configurare un'azione di approvazione per pubblicare un messaggio su un argomento di Amazon Simple Notification Service quando la pipeline interrompe l'azione. Amazon SNS invia il messaggio a tutti gli endpoint abbonati all'argomento. È necessario utilizzare un argomento creato nella stessa AWS regione della pipeline che includerà l'azione di approvazione. Quando crei un argomento, ti consigliamo di assegnargli un nome che ne identifichi lo scopo, in formati quali `MyFirstPipeline-us-east-2-approval`.

Quando pubblichi notifiche di approvazione su argomenti di Amazon SNS, puoi scegliere tra formati come destinatari e-mail o SMS, code SQS, endpoint HTTP/HTTPS o funzioni richiamate utilizzando Amazon SNS. AWS Lambda Per informazioni sulle notifiche tematiche di Amazon SNS, consulta i seguenti argomenti:

- Cos'è Amazon Simple Notification Service?
- Creare un argomento in Amazon SNS
- Invio di messaggi Amazon SNS a code Amazon SQS
- Iscrizione di una coda a un argomento Amazon SNS
- Invio di messaggi Amazon SNS a endpoint HTTP/HTTPS
- Richiamo di funzioni Lambda mediante notifiche Amazon SNS

Per la struttura dei dati JSON generati per una notifica dell'operazione di approvazione, consulta Formato di dati JSON per le notifiche di approvazione manuali in CodePipeline.

Specifica un URL per la revisione Come parte della configurazione dell'operazione di approvazione, puoi specificare un URL da esaminare. L'URL potrebbe essere un collegamento a un'applicazione Web che desideri venga testata dagli approvatori o una pagina con ulteriori informazioni sulla richiesta di approvazione. L'URL è incluso nella notifica pubblicata nell'argomento Amazon SNS. Gli approvatori possono utilizzare la console o l'interfaccia a riga di comando per visualizzarla.

Inserisci commenti per approvatori Quando crei un'operazione di approvazione, puoi anche aggiungere commenti che vengono visualizzati a coloro che ricevono le notifiche o che visualizzano l'operazione nella console o nella risposta dell'interfaccia a riga di comando.

Nessuna opzione di configurazione Puoi anche scegliere di non configurare alcuna di queste tre opzioni. Queste potrebbero non essere necessarie se, ad esempio, puoi inviare direttamente una notifica per segnalare che l'operazione è pronta per la revisione o se semplicemente desideri che l'esecuzione della pipeline venga interrotta finché non decidi di approvare tu stesso l'operazione.

Panoramica della configurazione e del flusso di lavoro per le azioni di approvazione in CodePipeline

Di seguito viene visualizzata una panoramica per la configurazione e l'utilizzo di approvazioni manuali.

1. Concedi le autorizzazioni IAM necessarie per approvare o rifiutare le azioni di approvazione a uno o più ruoli IAM nella tua organizzazione.
2. (Facoltativo) Se utilizzi le notifiche di Amazon SNS, assicurati che il ruolo di servizio che utilizzi nelle tue CodePipeline operazioni sia autorizzato ad accedere alle risorse Amazon SNS.
3. (Facoltativo) Se utilizzi le notifiche di Amazon SNS, crei un argomento Amazon SNS e aggiungi uno o più abbonati o endpoint.
4. Quando si utilizza la AWS CLI per creare la pipeline o dopo aver utilizzato la CLI o la console per creare la pipeline, si aggiunge un'azione di approvazione a una fase della pipeline.

   Se utilizzi le notifiche, includi l'Amazon Resource Name (ARN) dell'argomento Amazon SNS nella configurazione dell'azione. (Un ARN è un identificatore univoco per una risorsa Amazon. Gli ARN per gli argomenti di Amazon SNS sono strutturati *come* `arn:aws:sns:us-east` - 2:80398 ESEMPIO:. MyApprovalTopic Per ulteriori informazioni, consulta Amazon Resource Names (ARNs) e Servizio AWS namespace in.) Riferimenti generali di Amazon Web Services
5. La pipeline si interrompe quando raggiunge l'operazione di approvazione. Se un argomento ARN di Amazon SNS è stato incluso nella configurazione dell'azione, viene pubblicata una notifica sull'argomento Amazon SNS e viene inviato un messaggio a tutti gli abbonati all'argomento o agli endpoint sottoscritti, con un link per esaminare l'azione di approvazione nella console.
6. Un approvatore esamina l'URL di destinazione ed esamina gli eventuali commenti.
7. Utilizzando la console, l'interfaccia a riga di comando o l'SDK, l'approvatore fornisce un commento di riepilogo e invia una risposta:
   - Approvato: l'esecuzione della pipeline riprende.
   - Rifiutato: lo stato della fase viene modificato in "Non riuscita" e l'esecuzione della pipeline non riprende.

Se nessuna risposta viene inviata entro sette giorni, l'operazione viene contrassegnata come "Non riuscita".

Concedi le autorizzazioni di approvazione a un utente IAM in CodePipeline

Prima che gli utenti IAM della tua organizzazione possano approvare o rifiutare le azioni di approvazione, devono disporre delle autorizzazioni per accedere alle pipeline e aggiornare lo stato delle azioni di approvazione. Puoi concedere l'autorizzazione per accedere a tutte le pipeline e le azioni di approvazione del tuo account allegando la policy `AWSCodePipelineApproverAccess` gestita a un utente, ruolo o gruppo IAM; oppure puoi concedere autorizzazioni limitate specificando le singole risorse a cui può accedere un utente, un ruolo o un gruppo IAM.

> **Note**
>
> Le autorizzazioni descritte in questo argomento concedono un accesso molto limitato. Affinché un utente, ruolo o gruppo possa fare altro oltre che approvare o rifiutare operazioni di approvazione, puoi collegare altre policy gestite. Per informazioni sulle politiche gestite disponibili per, consulta. CodePipeline AWS politiche gestite per AWS CodePipeline

Concessione dell'autorizzazione di approvazione a tutte le pipeline e operazioni di approvazione

Per gli utenti che devono eseguire azioni di approvazione in CodePipeline, utilizza la policy `AWSCodePipelineApproverAccess` gestita.

Per fornire l'accesso, aggiungi autorizzazioni ai tuoi utenti, gruppi o ruoli:

- Utenti e gruppi in AWS IAM Identity Center:

  Crea un set di autorizzazioni. Segui le istruzioni riportate nella pagina Create a permission set (Creazione di un set di autorizzazioni) nella Guida per l'utente di AWS IAM Identity Center .

- Utenti gestiti in IAM tramite un provider di identità:

  Crea un ruolo per la federazione delle identità. Segui le istruzioni riportate nella pagina Creating a role for a third-party identity provider (federation) (Creazione di un ruolo per un provider di identità di terze parti [federazione]) nella Guida per l'utente di IAM.

- Utenti IAM:
  - Crea un ruolo che l'utente possa assumere. Per istruzioni, consulta la pagina Creating a role for an IAM user (Creazione di un ruolo per un utente IAM) nella Guida per l'utente di IAM.
  - (Non consigliato) Collega una policy direttamente a un utente o aggiungi un utente a un gruppo di utenti. Segui le istruzioni riportate nella pagina Aggiunta di autorizzazioni a un utente (console) nella Guida per l'utente di IAM.

Specifica dell'autorizzazione di approvazione per operazioni di approvazione e pipeline specifiche

Per gli utenti che devono eseguire azioni di approvazione in CodePipeline, utilizza la seguente politica personalizzata. Nella politica seguente, specifica le singole risorse a cui un utente può accedere. Ad esempio, la seguente politica concede agli utenti l'autorità di approvare o rifiutare solo l'azione indicata `MyApprovalAction` nella `MyFirstPipeline` pipeline nella regione degli Stati Uniti orientali (Ohio) (us-east-2):

> **Note**
>
> L'`codepipeline:ListPipelines`autorizzazione è richiesta solo se gli utenti IAM devono accedere alla CodePipeline dashboard per visualizzare questo elenco di pipeline. Se l'accesso alla console non è richiesto, puoi omettere `codepipeline:ListPipelines`.

Come utilizzare l'editor di policy JSON per creare una policy

1. Accedi AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nel riquadro di navigazione a sinistra, seleziona Policies (Policy).

   Se è la prima volta che selezioni Policy, verrà visualizzata la pagina Benvenuto nelle policy gestite. Seleziona Inizia.
3. Nella parte superiore della pagina, scegli Crea policy.
4. Nella sezione Editor di policy, scegli l'opzione JSON.
5. Inserisci il documento di policy JSON seguente:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:ListPipelines"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:GetPipeline",
                "codepipeline:GetPipelineState",
                "codepipeline:GetPipelineExecution"
            ],
            "Resource": "arn:aws:codepipeline:us-
east-2:80398EXAMPLE:MyFirstPipeline"
        },
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:PutApprovalResult"
            ],
            "Resource": "arn:aws:codepipeline:us-
east-2:80398EXAMPLE:MyFirstPipeline/MyApprovalStage/MyApprovalAction"
        }
    ]
}
```

6. Seleziona Successivo.

> **Note**
>
> È possibile alternare le opzioni dell'editor Visivo e JSON in qualsiasi momento. Se tuttavia si apportano modifiche o si seleziona Successivo nell'editor Visivo, IAM potrebbe ristrutturare la policy in modo da ottimizzarla per l'editor visivo. Per ulteriori informazioni, consulta Modifica della struttura delle policy nella Guida per l'utente di IAM.

7. Nella pagina Rivedi e crea, inserisci un valore in Nome policy e Descrizione (facoltativo) per la policy in fase di creazione. Rivedi Autorizzazioni definite in questa policy per visualizzare le autorizzazioni concesse dalla policy.
8. Seleziona Crea policy per salvare la nuova policy.

Concedi le autorizzazioni Amazon SNS per un ruolo di servizio CodePipeline

Se prevedi di utilizzare Amazon SNS per pubblicare notifiche su argomenti quando le azioni di approvazione richiedono una revisione, al ruolo di servizio che utilizzi nelle tue CodePipeline operazioni deve essere concessa l'autorizzazione per accedere alle risorse di Amazon SNS. Puoi utilizzare la console IAM per aggiungere questa autorizzazione al tuo ruolo di servizio.

Nella politica seguente, specifica la politica per la pubblicazione con SNS. Per la seguente politica, puoi assegnarle un nome. `SNSPublish` Utilizza la seguente politica associandola al tuo ruolo di servizio.

> **Important**
>
> Assicurati di aver effettuato l'accesso AWS Management Console con le stesse informazioni sull'account che hai utilizzato. Guida introduttiva con CodePipeline

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sns:Publish",
            "Resource": "*"
        }
    ]
}
```

Come utilizzare l'editor di policy JSON per creare una policy

1. Accedi AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.

2. Nel riquadro di navigazione a sinistra, seleziona Policies (Policy).

   Se è la prima volta che selezioni Policy, verrà visualizzata la pagina Benvenuto nelle policy gestite. Seleziona Inizia.

3. Nella parte superiore della pagina, scegli Crea policy.
4. Nella sezione Editor di policy, scegli l'opzione JSON.
5. Immettere o incollare un documento di policy JSON. Per dettagli sul linguaggio della policy IAM, consulta la Documentazione di riferimento delle policy JSON IAM.
6. Risolvi eventuali avvisi di sicurezza, errori o avvisi generali generati durante la convalida delle policy, quindi scegli Next (Successivo).

   > **Note**
   >
   > È possibile alternare le opzioni dell'editor Visivo e JSON in qualsiasi momento. Se tuttavia si apportano modifiche o si seleziona Successivo nell'editor Visivo, IAM potrebbe ristrutturare la policy in modo da ottimizzarla per l'editor visivo. Per ulteriori informazioni, consulta Modifica della struttura delle policy nella Guida per l'utente di IAM.

7. (Facoltativo) Quando crei o modifichi una policy in AWS Management Console, puoi generare un modello di policy JSON o YAML da utilizzare nei modelli. AWS CloudFormation

   Per fare ciò, nell'editor delle politiche scegli Azioni, quindi scegli Genera modello. CloudFormation Per saperne di più AWS CloudFormation, consulta il riferimento al tipo di AWS Identity and Access Management risorsa nella Guida AWS CloudFormation per l'utente.

8. Una volta terminata l'aggiunta delle autorizzazioni alla policy, scegli Successivo.
9. Nella pagina Rivedi e crea, immettere un valore in Nome policy e Descrizione (facoltativo) per la policy in fase di creazione. Rivedi Autorizzazioni definite in questa policy per visualizzare le autorizzazioni concesse dalla policy.
10. (Facoltativo) Aggiungere metadati alla policy collegando i tag come coppie chiave-valore. Per ulteriori informazioni sull'utilizzo di tag in IAM, consulta la sezione Applicazione di tag alle risorse IAM nella Guida per l'utente di IAM.
11. Seleziona Crea policy per salvare la nuova policy.

Aggiungi un'azione di approvazione manuale a una pipeline in CodePipeline

Puoi aggiungere un'azione di approvazione a una fase di una CodePipeline pipeline nel punto in cui desideri che la pipeline si interrompa in modo che qualcuno possa approvare o rifiutare manualmente l'azione.

> **Note**
>
> Le operazioni di approvazione non possono essere aggiunte nelle fasi Origine. Le fasi Origine possono contenere solo operazioni di origine.

Se desideri utilizzare Amazon SNS per inviare notifiche quando un'azione di approvazione è pronta per la revisione, devi prima completare i seguenti prerequisiti:

- Concedi l'autorizzazione al tuo ruolo CodePipeline di servizio per accedere alle risorse Amazon SNS. Per informazioni, consulta Concedi le autorizzazioni Amazon SNS per un ruolo di servizio CodePipeline.
- Concedi l'autorizzazione a una o più identità IAM della tua organizzazione per aggiornare lo stato di un'azione di approvazione. Per informazioni, consulta Concedi le autorizzazioni di approvazione a un utente IAM in CodePipeline.

In questo esempio, crei una nuova fase di approvazione e aggiungi un'azione di approvazione manuale alla fase. È inoltre possibile aggiungere un'azione di approvazione manuale a una fase esistente che contiene altre azioni.

Aggiungi un'azione di approvazione manuale a una CodePipeline pipeline (console)

Puoi utilizzare la CodePipeline console per aggiungere un'azione di approvazione a una CodePipeline pipeline esistente. È necessario utilizzare la AWS CLI se si desidera aggiungere azioni di approvazione quando si crea una nuova pipeline.

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. In Name (Nome), seleziona la pipeline.
3. Nella pagina dei dettagli della pipeline, scegliere Edit (Modifica).
4. Se desideri aggiungere un'operazione di approvazione a una nuova fase, scegli +Add stage (+ Aggiungi fase) nel punto della pipeline in cui desideri aggiungere una richiesta di approvazione,

quindi immetti un nome per la fase. Nella pagina Add stage (Aggiungi fase), in Stage name (Nome fase), inserire il nuovo nome della fase. Ad esempio, aggiungere una nuova fase e denominarla `Manual_Approval`.

Se desideri aggiungere un'operazione di approvazione a una fase esistente, scegli Edit stage (Modifica fase).

5. Nella fase in cui si desidera aggiungere l'operazione di approvazione, scegliere + Add action group (+Aggiungi gruppo di operazioni).
6. Nella pagina Edit action (Modifica operazione), esegui le seguenti operazioni:
   1. In Action name (Nome operazione), immetti un nome per identificare l'operazione.
   2. In Action provider (Provider operazione), in Approval (Approvazione), scegli Manual approval (Approvazione manuale).
   3. (Facoltativo) In SNS topic ARN (ARN argomento SNS), scegli il nome dell'argomento da utilizzare per inviare notifiche per l'operazione di approvazione.
   4. (Facoltativo) In URL for review (URL per revisione), immetti l'URL della pagina o dell'applicazione che deve essere esaminata dall'approvatore. Gli approvatori possono accedere a questo URL tramite un collegamento incluso nella vista della console della pipeline.
   5. (Facoltativo) In Comments (Commenti), immetti eventuali altre informazioni che desideri condividere con il revisore.
   6. Selezionare Salva.

Aggiungere un'azione di approvazione manuale a una CodePipeline pipeline (CLI)

Puoi utilizzare l'interfaccia a riga di comando per aggiungere un'operazione di approvazione a una pipeline esistente o quando crei una pipeline. A questo scopo, includi un'operazione di approvazione, con il tipo di approvazione manuale, in una fase che stai creando o modificando.

Per ulteriori informazioni sulla creazione e la modifica di pipeline, consulta Creare una pipeline in CodePipeline e Modificare una tubazione in CodePipeline.

Per aggiungere una fase a una pipeline che include solo un'operazione di approvazione, includi quanto segue durante la creazione o l'aggiornamento della pipeline.

> **Note**
>
> La sezione `configuration` è facoltativa. Viene visualizzata solo una parte, non l'intera struttura. Per ulteriori informazioni, consulta CodePipeline riferimento alla struttura della tubazione.

```
{
    "name": "MyApprovalStage",
    "actions": [
        {
            "name": "MyApprovalAction",
            "actionTypeId": {
                "category": "Approval",
                "owner": "AWS",
                "version": "1",
                "provider": "Manual"
            },
            "inputArtifacts": [],
            "outputArtifacts": [],
            "configuration": {
                "NotificationArn": "arn:aws:sns:us-
east-2:80398EXAMPLE:MyApprovalTopic",
                "ExternalEntityLink": "http://example.com",
                "CustomData": "The latest changes include feedback from Bob."},
            "runOrder": 1
        }
    ]
}
```

Se l'operazione di approvazione si trova in una fase con altre operazioni, l'aspetto della sezione del file JSON contenente la fase potrebbe invece essere simile all'esempio seguente.

> **Note**
>
> La sezione `configuration` è facoltativa. Viene visualizzata solo una parte, non l'intera struttura. Per ulteriori informazioni, consulta CodePipeline riferimento alla struttura della tubazione.

```
,
{
    "name": "Production",
    "actions": [
        {
            "inputArtifacts": [],
            "name": "MyApprovalAction",
            "actionTypeId": {
                "category": "Approval",
                "owner": "AWS",
                "version": "1",
                "provider": "Manual"
            },
            "outputArtifacts": [],
            "configuration": {
                "NotificationArn": "arn:aws:sns:us-
east-2:80398EXAMPLE:MyApprovalTopic",
                "ExternalEntityLink": "http://example.com",
                "CustomData": "The latest changes include feedback from Bob."
            },
            "runOrder": 1
        },
        {
            "inputArtifacts": [
                {
                    "name": "MyApp"
                }
            ],
            "name": "MyDeploymentAction",
            "actionTypeId": {
                "category": "Deploy",
                "owner": "AWS",
                "version": "1",
                "provider": "CodeDeploy"
            },
            "outputArtifacts": [],
            "configuration": {
                "ApplicationName": "MyDemoApplication",
                "DeploymentGroupName": "MyProductionFleet"
            },
            "runOrder": 2
        }
    ]
```

```
}
```

Approva o rifiuta un'azione di approvazione in CodePipeline

Quando una pipeline include un'operazione di approvazione, l'esecuzione della pipeline si interrompe nel momento in cui l'operazione è stata aggiunta. La pipeline non riprende a meno che l'operazione non venga approvata manualmente. Se un approvatore rifiuta l'operazione o se nessuna risposta di approvazione viene ricevuta entro sette giorni dall'interruzione della pipeline per l'operazione di approvazione, lo stato della pipeline diventa "Non riuscita".

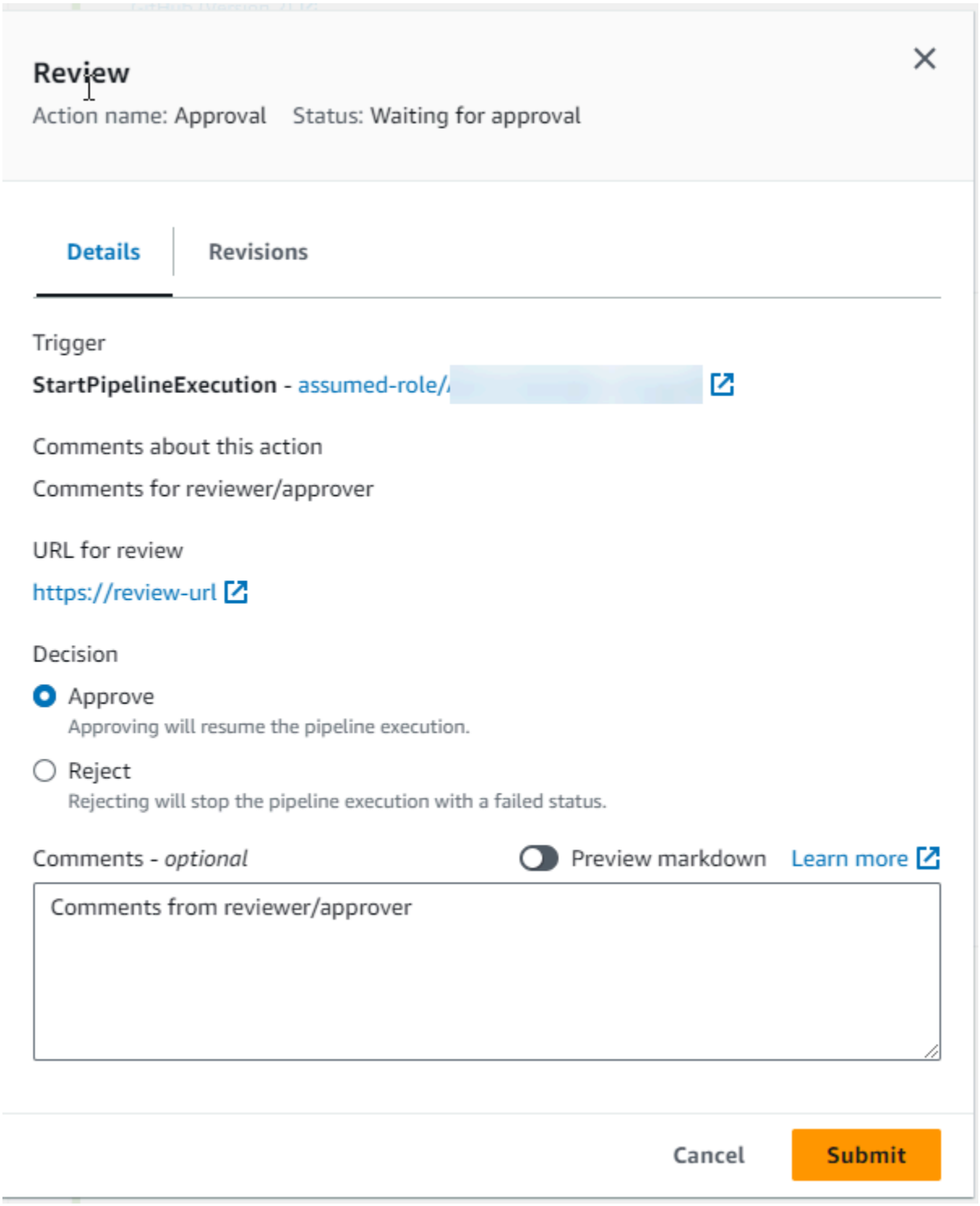
Se la persona che ha aggiunto l'azione di approvazione alla pipeline ha configurato le notifiche, potresti ricevere un'email con le informazioni sulla pipeline e lo stato di approvazione.

Approvazione o rifiuto di un'operazione di approvazione (console)

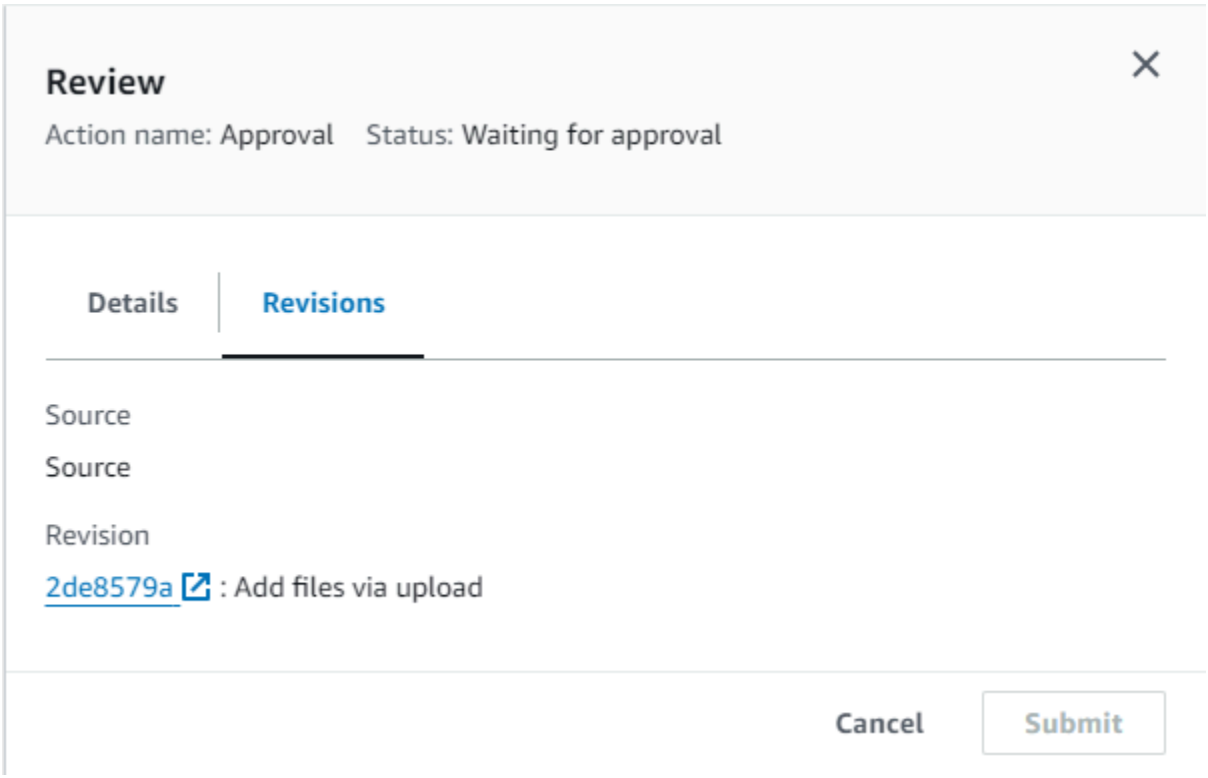
Se ricevi una notifica che include un collegamento diretto a un'operazione di approvazione, scegli il collegamento Approve or reject (Approva o rifiuta), accedi alla console e quindi continua con il passaggio 7. In caso contrario, segui tutti i passaggi.

1. Apri la CodePipeline console all'indirizzo https://console.aws.amazon.com/codepipeline/.
2. Nella pagina All Pipelines (Tutte le pipeline), selezionare il nome della pipeline.
3. Individuare la fase con l'operazione di approvazione. Scegli Rivedi.

   Viene visualizzata la finestra di dialogo Revisione. La scheda Dettagli mostra il contenuto e i commenti della recensione.



La scheda Revisioni mostra le revisioni di origine per l'esecuzione.



4. Nella scheda Dettagli, visualizza i commenti e l'URL, se presenti. Il messaggio contiene anche l'URL dei contenuti da rivedere, se uno è stato incluso.
5. Se è stato fornito un URL, scegli il link URL per la revisione nell'azione per aprire la pagina web di destinazione, quindi esamina il contenuto.
6. Nella finestra Revisione, inserisci i commenti di recensione, ad esempio il motivo per cui stai approvando o rifiutando l'azione, quindi scegli Approva o Rifiuta.
7. Scegli Invia.

Approvazione o rifiuto di una richiesta di approvazione (CLI)

Per utilizzare l'interfaccia a riga di comando per rispondere a un'operazione di approvazione, utilizza innanzitutto il comando get-pipeline-state per recuperare il token associato all'ultima esecuzione dell'operazione di approvazione.

1. In un terminale (Linux, macOS o Unix) o nel prompt dei comandi (Windows), esegui il get-pipeline-statecomando sulla pipeline che contiene l'azione di approvazione. Ad esempio, per una pipeline denominata *MyFirstPipeline*, inserisci quanto segue:

```
aws codepipeline get-pipeline-state --name MyFirstPipeline
```

2. Nella risposta al comando, individuare il valore `token`, che appare in `latestExecution` nella sezione `actionStates` per l'operazione di approvazione, come mostrato qui:

```
{
    "created": 1467929497.204,
    "pipelineName": "MyFirstPipeline",
    "pipelineVersion": 1,
    "stageStates": [
        {
            "actionStates": [
                {
                    "actionName": "MyApprovalAction",
                    "currentRevision": {
                        "created": 1467929497.204,
                        "revisionChangeId": "CEM7d6Tp7zfelUSLCPPwo234xEXAMPLE",
                        "revisionId": "HYGp7zmwbCPPwo23xCMdTeqIlEXAMPLE"
                    },
                    "latestExecution": {
                        "lastUpdatedBy": "identity",
                        "summary": "The new design needs to be reviewed before
 release.",
                        "token": "1a2b3c4d-573f-4ea7-a67E-XAMPLETOKEN"
                    }
                }
//More content might appear here
}
```

3. In un editor di testo normale, creare un file in cui viene aggiunto quanto segue, in formato JSON:

   - Il nome della pipeline contenente l'operazione di approvazione.
   - Il nome della fase contenente l'operazione di approvazione.
   - Il nome dell'operazione di approvazione.
   - Il valore del token raccolto nella fase precedente.
   - La risposta all'operazione (Approvata o Rifiutata). La risposta deve essere maiuscola.
   - I commenti di riepilogo.

   *MyFirstPipeline*Nell'esempio precedente, il file dovrebbe avere il seguente aspetto:

```
{
  "pipelineName": "MyFirstPipeline",
```

```
  "stageName": "MyApprovalStage",
  "actionName": "MyApprovalAction",
  "token": "1a2b3c4d-573f-4ea7-a67E-XAMPLETOKEN",
  "result": {
    "status": "Approved",
    "summary": "The new design looks good. Ready to release to customers."
  }
}
```

4. Salvare il file con un nome come **`approvalstage-approved.json`**.
5. Esegui il put-approval-resultcomando, specificando il nome del file JSON di approvazione, simile al seguente:

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

```
aws codepipeline put-approval-result --cli-input-json file://approvalstage-approved.json
```

Formato di dati JSON per le notifiche di approvazione manuali in CodePipeline

Per le azioni di approvazione che utilizzano le notifiche di Amazon SNS, i dati JSON relativi all'azione vengono creati e pubblicati su Amazon SNS quando la pipeline si interrompe. Puoi utilizzare l'output JSON per inviare messaggi alle code di Amazon SQS o richiamare funzioni in. AWS Lambda

> **Note**
>
> In questa guida non viene spiegato come configurare le notifiche tramite JSON. Per informazioni, consulta Invio di messaggi Amazon SNS a code Amazon SQS e richiamo di funzioni Lambda utilizzando Amazon SNS Notifications nella Amazon SNS Developer Guide.

L'esempio seguente mostra la struttura dell'output JSON disponibile con le approvazioni. CodePipeline

```
{
    "region": "us-east-2",
    "consoleLink": "https://console.aws.amazon.com/codepipeline/home?region=us-east-2#/
view/MyFirstPipeline",
    "approval": {
        "pipelineName": "MyFirstPipeline",
        "stageName": "MyApprovalStage",
        "actionName": "MyApprovalAction",
        "token": "1a2b3c4d-573f-4ea7-a67E-XAMPLETOKEN",
        "expires": "2016-07-07T20:22Z",
        "externalEntityLink": "http://example.com",
        "approvalReviewLink": "https://console.aws.amazon.com/codepipeline/
home?region=us-east-2#/view/MyFirstPipeline/MyApprovalStage/MyApprovalAction/
approve/1a2b3c4d-573f-4ea7-a67E-XAMPLETOKEN",
        "customData": "Review the latest changes and approve or reject within seven
 days."
    }
}
```

Aggiungere un'azione interregionale in CodePipeline

AWS CodePipeline include una serie di azioni che consentono di configurare, testare e distribuire risorse per il processo di rilascio automatico. Puoi aggiungere azioni alla tua pipeline che si trovano in una AWS regione diversa dalla tua pipeline. Quando un Servizio AWS è il provider di un'azione e questo tipo di azione/tipo di provider si trova in una AWS regione diversa dalla pipeline, si tratta di un'azione interregionale.

> **Note**
>
> Le azioni interregionali sono supportate e possono essere create solo nelle regioni in cui AWS è supportata. CodePipeline Per un elenco delle AWS regioni supportate per CodePipeline, consultaQuote in AWS CodePipeline.

Puoi utilizzare la console o AWS CloudFormation aggiungere azioni interregionali nelle pipeline. AWS CLI

> **Note**
>
> Alcuni tipi di azioni CodePipeline potrebbero essere disponibili solo in determinate AWS regioni. Tieni inoltre presente che potrebbero esserci AWS regioni in cui è disponibile un tipo di azione, ma non è disponibile un AWS provider specifico per quel tipo di azione.

Quando crei o modifichi una pipeline, devi disporre di un bucket di artefatti nella regione della pipeline e di un bucket di artefatti per ogni regione in cui prevedi di eseguire un'operazione. Per ulteriori informazioni sul parametro `ArtifactStores`, vedi CodePipeline riferimento alla struttura della tubazione.

> **Note**
>
> CodePipeline gestisce la copia di artefatti da una AWS regione all'altra durante l'esecuzione di azioni interregionali.

Se si utilizza la console per creare una pipeline o azioni interregionali, i bucket di artefatti predefiniti vengono configurati nelle regioni in cui sono presenti le azioni. CodePipeline Quando utilizzi AWS CLI AWS CloudFormation, o un SDK per creare una pipeline o azioni interregionali, fornisci il bucket di artefatti per ogni regione in cui sono presenti azioni.

> **Note**
>
> È necessario creare il bucket di artefatti e la chiave di crittografia nella stessa area dell'azione AWS Interregionale e nello stesso account della pipeline.

Non è possibile creare operazioni tra regioni per i seguenti tipi di operazione:

- Operazioni di origine
- Operazioni di terze parti
- Operazioni personalizzate

> **Note**
>
> Quando si utilizza Cross-region Lambda invoke action CodePipeline in, lo stato dell'esecuzione lambda che utilizza PutJobSuccessResultPutJobFailureResulte deve essere inviato alla AWS regione in cui è presente la funzione Lambda e non alla regione in cui esiste. CodePipeline

Quando una pipeline include un'azione interregionale come parte di una fase, CodePipeline replica solo gli artefatti di input dell'azione Cross-region dalla regione della pipeline alla regione dell'azione.

> **Note**
>
> La regione della pipeline e la regione in cui sono gestite le risorse di rilevamento delle modifiche CloudWatch agli eventi rimangono invariate. La regione in cui è ospitata la pipeline non cambia.

Gestione di operazioni tra regioni in una pipeline (console)

Puoi utilizzare la CodePipeline console per aggiungere un'azione interregionale a una pipeline esistente. Per creare una nuova pipeline con operazioni tra regioni utilizzando la procedura guidata di creazione pipeline, consulta Creazione di una pipeline (console).

Nella console, crei un'operazione tra regioni in una fase della pipeline scegliendo il provider dell'operazione e il campo Regione che elenca le risorse create in quella regione per quel provider. Quando aggiungi un'azione interregionale, CodePipeline utilizza un bucket di artefatti separato nella regione dell'azione. Per ulteriori informazioni sui bucket di artefatti tra più regioni, consulta CodePipeline riferimento alla struttura della tubazione.

Aggiunta di un'operazione tra regioni a una fase della pipeline (console)

Utilizza la console per aggiungere un'operazione tra regioni a una pipeline.

> **Note**
>
> Se la pipeline è in esecuzione quando le modifiche vengono salvate, l'esecuzione non viene portata a termine.

Per aggiungere un'operazione tra regioni

1. Accedi alla console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Seleziona la pipeline, quindi scegli Edit (Modifica).
3. Nella parte inferiore del diagramma, scegli + Add stage (+ Aggiungi fase) se stai aggiungendo una nuova fase oppure scegli Edit stage (Modifica fase) se vuoi aggiungere l'operazione a una fase esistente.
4. In Edit: <Stage> (Modifica: <Fase>), scegli + Add action group (+ Aggiungi gruppo di operazioni) per aggiungere un'operazione seriale. In alternativa, scegli + Add action (+ Aggiungi operazione) per aggiungere un'operazione parallela.
5. Nella pagina Edit action (Modifica operazione):
   a. In Nome operazione immetti un nome per l'operazione tra regioni.
   b. In Azione provider (Provider operazione), scegli il provider dell'operazione.
   c. In Regione, scegli la AWS regione in cui hai creato o intendi creare la risorsa per l'azione. Quando la regione viene selezionata, le risorse disponibili per la regione sono elencate per la selezione. Il campo Regione indica dove vengono create le AWS risorse per questo tipo di azione e tipo di provider. Questo campo viene visualizzato solo per le azioni in cui il fornitore dell'azione è un Servizio AWS. Per impostazione predefinita, il campo Regione è lo Regione AWS stesso della pipeline.
   d. In Input artifacts (Artefatti di input) scegli l'input appropriato dalla fase precedente. Ad esempio, se la fase precedente è una fase di origine, scegli. SourceArtifact
   e. Completa tutti i campi obbligatori per il provider dell'operazione che stai configurando.
   f. In Output artifacts (Artefatti di output) scegli l'output appropriato per la prossima fase. Ad esempio, se la fase successiva è una fase di distribuzione, scegli BuildArtifact.
   g. Selezionare Salva.
6. In Edit: <Stage> (Modifica: <Fase>), scegli Done (Fatto).
7. Selezionare Salva.

Modifica di un'operazione tra regioni in una fase della pipeline (console)

Utilizza la console per modificare un'operazione tra regioni esistente in una pipeline.

> **Note**
>
> Se la pipeline è in esecuzione quando le modifiche vengono salvate, l'esecuzione non viene portata a termine.

Per modificare un'operazione tra regioni

1. Accedi alla console all'indirizzo https://console.aws.amazon.com/codesuite/codepipeline/home.
2. Seleziona la pipeline, quindi scegli Edit (Modifica).
3. Scegli Edit stage (Modifica fase).
4. In Edit: <Stage> (Modifica: <Fase>), scegli l'icona per modificare un'operazione esistente.
5. Nella pagina Edit action (Modifica operazione), apporta modifiche ai campi in base alle esigenze.
6. In Edit: <Stage> (Modifica: <Fase>), scegli Done (Fatto).
7. Selezionare Salva.

Eliminazione di un'operazione tra regioni da una fase della pipeline (console)

Utilizza la console per eliminare un'operazione tra regioni esistente da una pipeline.

> **Note**
>
> Se la pipeline è in esecuzione quando le modifiche vengono salvate, l'esecuzione non viene portata a termine.

Per eliminare un'operazione tra regioni

1. Accedi alla console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Seleziona la pipeline, quindi scegli Edit (Modifica).
3. Scegli Edit stage (Modifica fase).
4. In Edit: <Stage> (Modifica: <Fase>), scegli l'icona per eliminare un'operazione esistente.
5. In Edit: <Stage> (Modifica: <Fase>), scegli Done (Fatto).
6. Selezionare Salva.

Aggiunta di un'operazione tra regioni a una pipeline (CLI)

È possibile utilizzare il AWS CLI per aggiungere un'azione interregionale a una pipeline esistente.

Per creare un'azione interregionale in una fase di pipeline con AWS CLI, aggiungete l'azione di configurazione insieme a un campo opzionale. `region` Devi avere inoltre già creato un bucket di artefatti nella regione dell'operazione. Invece di fornire il parametro `artifactStore` della pipeline della singola regione, utilizza il parametro `artifactStores` per includere un elenco del bucket di artefatti di ogni regione.

> **Note**
>
> In questa procedura guidata e nei relativi esempi, *`RegionA`* è la regione in cui la pipeline viene creata. Ha accesso al bucket *`RegionA`* Amazon S3 utilizzato per archiviare gli artefatti della pipeline e al ruolo di servizio utilizzato da. CodePipeline *`RegionB`* è la regione in cui vengono creati l'applicazione, CodeDeploy il gruppo di distribuzione e il ruolo di servizio utilizzati CodeDeploy da.

Prerequisiti

Devi aver creato i seguenti elementi:

- Una pipeline in *`RegionA`*.
- *`Un bucket di artefatti Amazon S3 in regionB.`*
- *`Le risorse per la tua azione, ad esempio CodeDeploy l'applicazione e il gruppo di distribuzione per un'azione di distribuzione tra regioni, in RegionB.`*

Aggiunta di un'operazione tra regioni a una pipeline (CLI)

Usa AWS CLI per aggiungere un'azione interregionale a una pipeline.

Per aggiungere un'operazione tra regioni

1. Per una pipeline in *`RegionA`*, esegui il comando get-pipeline per copiare la struttura della pipeline in un file JSON. Ad esempio, per una pipeline denominata `MyFirstPipeline`, esegui il seguente comando:

```
aws codepipeline get-pipeline --name MyFirstPipeline >pipeline.json
```

Questo comando non restituisce alcun valore, ma nella directory in cui è stato eseguito dovrebbe comparire il file creato.

2. Aggiungi il campo `region` per aggiungere una nuova fase con l'operazione tra regioni che include la regione e le risorse per l'operazione. L'esempio JSON seguente aggiunge una fase di distribuzione con un'azione di distribuzione tra regioni in cui si trova il provider, in una nuova regione. CodeDeploy `us-east-1`

```
{
                "name": "Deploy",
                "actions": [
                    {
                        "inputArtifacts": [
                            {
                                "name": "SourceArtifact"
                            }
                        ],
                        "name": "Deploy",
                        "region": "RegionB",
                        "actionTypeId": {
                            "category": "Deploy",
                            "owner": "AWS",
                            "version": "1",
                            "provider": "CodeDeploy"
                        },
                        "outputArtifacts": [],
                        "configuration": {
                            "ApplicationName": "name",
                            "DeploymentGroupName": "name"
                        },
                        "runOrder": 1
                    }
```

3. Nella struttura della pipeline, rimuovi il campo `artifactStore` e aggiungi la mappa `artifactStores` per la nuova operazione tra regioni. La mappatura deve includere una voce per ogni AWS regione in cui sono presenti azioni. Per ogni voce della mappatura, le risorse devono trovarsi nella rispettiva AWS regione. Nell'esempio seguente, `ID-A` è l'ID della chiave di crittografia per la *RegionA* e `ID-B` è la chiave di crittografia per la *RegionB*.

```
"artifactStores":{
   "RegionA":{
      "encryptionKey":{
         "id":"ID-A",
         "type":"KMS"
      },
      "location":"Location1",
      "type":"S3"
   },
   "RegionB":{
      "encryptionKey":{
         "id":"ID-B",
         "type":"KMS"
      },
      "location":"Location2",
      "type":"S3"
   }
}
```

L'esempio di JSON seguente mostra il bucket us-west-2 come `my-storage-bucket` e aggiunge il nuovo bucket us-east-1 denominato `my-storage-bucket-us-east-1`.

```
        "artifactStores": {
            "us-west-2": {
                "type": "S3",
                "location": "my-storage-bucket"
            },
            "us-east-1": {
                "type": "S3",
                "location": "my-storage-bucket-us-east-1"
            }
        },
```

4. Se stai utilizzando la struttura della pipeline recuperata tramite il comando get-pipeline, rimuovi le righe `metadata` dal file JSON. In caso contrario, il comando update-pipeline non è in grado di utilizzarlo. Rimuovi le righe `"metadata": { }` e i campi `"created"`, `"pipelineARN"` e `"updated"`.

   Ad esempio, rimuovere dalla struttura le seguenti righe:

```
"metadata": {
```

```
  "pipelineArn": "arn:aws:codepipeline:region:account-ID:pipeline-name",
  "created": "date",
  "updated": "date"
  }
```

Salvare il file.

5. Per applicare le modifiche, eseguire il comando update-pipeline, specificando il file JSON della pipeline:

> **Important**
>
> Assicurarsi di includere `file://` prima del nome del file. Questo è obbligatorio in questo comando.

```
aws codepipeline update-pipeline --cli-input-json file://pipeline.json
```

Questo comando restituisce l'intera struttura della pipeline modificata. L'output è simile a quello riportato di seguito.

```
{
    "pipeline": {
        "version": 4,
        "roleArn": "ARN",
        "stages": [
            {
                "name": "Source",
                "actions": [
                    {
                        "inputArtifacts": [],
                        "name": "Source",
                        "actionTypeId": {
                            "category": "Source",
                            "owner": "AWS",
                            "version": "1",
                            "provider": "CodeCommit"
                        },
                        "outputArtifacts": [
                            {
                                "name": "SourceArtifact"
```

```
                        }
                    ],
                    "configuration": {
                        "PollForSourceChanges": "false",
                        "BranchName": "main",
                        "RepositoryName": "MyTestRepo"
                    },
                    "runOrder": 1
                }
            ]
        },
        {
            "name": "Deploy",
            "actions": [
                {
                    "inputArtifacts": [
                        {
                            "name": "SourceArtifact"
                        }
                    ],
                    "name": "Deploy",
                    "region": "us-east-1",
                    "actionTypeId": {
                        "category": "Deploy",
                        "owner": "AWS",
                        "version": "1",
                        "provider": "CodeDeploy"
                    },
                    "outputArtifacts": [],
                    "configuration": {
                        "ApplicationName": "name",
                        "DeploymentGroupName": "name"
                    },
                    "runOrder": 1
                }
            ]
        }
    ],
    "name": "AnyCompanyPipeline",
    "artifactStores": {
        "us-west-2": {
            "type": "S3",
            "location": "my-storage-bucket"
        },
```

```
            "us-east-1": {
                "type": "S3",
                "location": "my-storage-bucket-us-east-1"
            }
        }
    }
}
```

> **Note**
>
> Il comando update-pipeline arresta la pipeline. Se è in corso di elaborazione una versione durante l'esecuzione del comando update-pipeline, tale elaborazione viene arrestata. Per elaborare tale versione utilizzando la pipeline aggiornata, devi avviare manualmente la pipeline. Utilizza il comando **`start-pipeline-execution`** per avviare manualmente la pipeline.

6. Dopo aver aggiornato la pipeline, l'azione Interregione viene visualizzata nella console.

Aggiunta di un'operazione tra regioni a una pipeline (AWS CloudFormation)

È possibile utilizzare AWS CloudFormation per aggiungere un'azione interregionale a una pipeline esistente.

Per aggiungere un'azione interregionale con AWS CloudFormation

1. Aggiungi il parametro `Region` alla risorsa `ActionDeclaration` nel modello, come illustrato in questo esempio:

```
ActionDeclaration:
              Type: Object
              Properties:
                ActionTypeId:
                  Type: ActionTypeId
                  Required: true
                Configuration:
                  Type: Map
                InputArtifacts:
                  Type: Array
                  ItemType:
                    Type: InputArtifact
                Name:
                  Type: String
                  Required: true
                OutputArtifacts:
                  Type: Array
                  ItemType:
                    Type: OutputArtifact
                RoleArn:
                  Type: String
                RunOrder:
                  Type: Integer
                Region:
                  Type: String
```

2. In `Mappings`, aggiungi la mappa della regione come illustrato in questo esempio, relativo a una mappatura denominata `SecondRegionMap` che mappa i valori per le chiavi `RegionA` e `RegionB`. Nella risorsa `Pipeline`, nel campo `artifactStore`, aggiungi la mappa `artifactStores` per la nuova operazione tra regioni come segue:

```
Mappings:
  SecondRegionMap:
    RegionA:
      SecondRegion: "RegionB"
    RegionB:
      SecondRegion: "RegionA"

...

          Properties:
            ArtifactStores:
```

```
            -
              Region: RegionB
              ArtifactStore:
                Type: "S3"
                Location: test-cross-region-artifact-store-bucket-RegionB
            -
              Region: RegionA
              ArtifactStore:
                Type: "S3"
                Location: test-cross-region-artifact-store-bucket-RegionA
```

Il seguente esempio YAML mostra il bucket della *RegionA* come `us-west-2` e aggiunge il nuovo bucket della *RegionB*, `eu-central-1`:

```
Mappings:
  SecondRegionMap:
    us-west-2:
      SecondRegion: "eu-central-1"
    eu-central-1:
      SecondRegion: "us-west-2"

...

          Properties:
            ArtifactStores:
              -
                Region: eu-central-1
                ArtifactStore:
                  Type: "S3"
                  Location: test-cross-region-artifact-store-bucket-eu-central-1
              -
                Region: us-west-2
                ArtifactStore:
                  Type: "S3"
                  Location: test-cross-region-artifact-store-bucket-us-west-2
```

3. Salva il modello aggiornato nel computer locale e quindi apri la console AWS CloudFormation .
4. Seleziona lo stack e scegli Create Change Set for Current Stack (Crea set di modifiche per lo stack corrente).
5. Carica il modello e quindi visualizza le modifiche elencate in AWS CloudFormation. Queste sono le modifiche da apportare allo stack. Le nuove risorse dovrebbero essere visibili nell'elenco.

6. Scegliere Execute (Esegui).

Utilizzo delle variabili

Alcune azioni nelle variabili di CodePipeline generazione. Per utilizzare le variabili:

- Si assegna uno spazio dei nomi a un'azione per rendere disponibili le variabili che produce per una configurazione di azione downstream.
- È possibile configurare l'azione downstream per utilizzare le variabili generate dall'azione.

  È possibile visualizzare i dettagli per ogni esecuzione di azione per visualizzare i valori per ogni variabile di output generata dall'azione in fase di esecuzione.

Per vedere step-by-step esempi di utilizzo delle variabili:

- Per un tutorial con un'azione Lambda che utilizza le variabili di un'azione upstream (CodeCommit) e genera variabili di output, consulta. Tutorial: Utilizzo delle variabili con le azioni di richiamo Lambda
- Per un tutorial con un' AWS CloudFormation azione che fa riferimento alle variabili di output impilate di un'azione upstream CloudFormation , consulta. Tutorial: crea una pipeline che utilizza le variabili delle azioni di AWS CloudFormation distribuzione
- Per un esempio di azione di approvazione manuale con testo del messaggio che fa riferimento a variabili di output che si CodeCommit risolvono nell'ID di commit e nel messaggio di commit, vedi. Esempio: utilizzo delle variabili nelle approvazioni manuali
- Per un esempio di CodeBuild azione con una variabile di ambiente che si risolve nel nome del GitHub ramo, vedi. Esempio: utilizzare una BranchName variabile con variabili di CodeBuild ambiente
- CodeBuild le azioni producono come variabili tutte le variabili di ambiente che sono state esportate come parte della build. Per ulteriori informazioni, consulta CodeBuild azioni (variabili di output). Per un elenco delle variabili di ambiente in cui è possibile utilizzare CodeBuild, consulta Variabili di ambiente negli ambienti di compilazione nella Guida per l'AWS CodeBuild utente.

Argomenti

- Configurazione delle operazioni per le variabili
- Visualizzazione delle variabili di output

- Esempio: utilizzo delle variabili nelle approvazioni manuali
- Esempio: utilizzare una BranchName variabile con variabili di CodeBuild ambiente

Configurazione delle operazioni per le variabili

Quando si aggiunge un'azione alla pipeline, è possibile assegnargli uno spazio dei nomi e configurarlo per utilizzare le variabili delle azioni precedenti.

Configurazione delle operazioni con le variabili (console)

Questo esempio crea una pipeline con un'azione di CodeCommit origine e un'azione di CodeBuild compilazione. L' CodeBuild azione è configurata per utilizzare le variabili prodotte dall' CodeCommit azione.

Se lo spazio dei nomi non è specificato, le variabili non sono disponibili per riferimento nella configurazione dell'azione. Quando si utilizza la console per creare una pipeline, lo spazio dei nomi per ogni azione viene generato automaticamente.

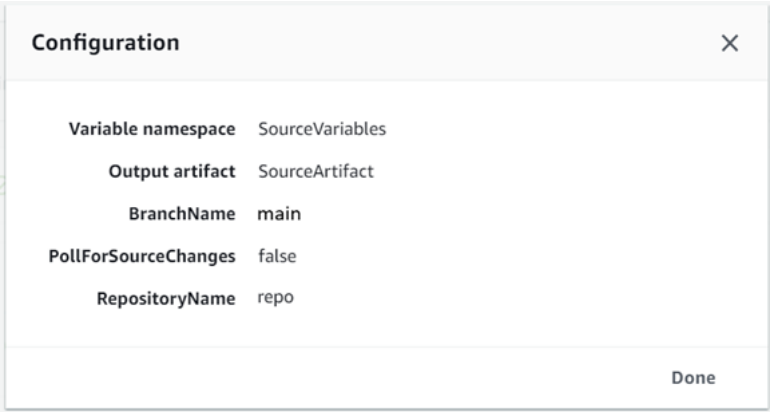
Per creare una pipeline con variabili

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Scegliere Create pipeline (Crea pipeline). Immettere un nome per il dispositivo, quindi scegliere Next (Avanti).
3. In Source, in Provider, scegli CodeCommit. Scegli il CodeCommit repository e il ramo per l'azione di origine, quindi scegli Avanti.
4. In Build, in Provider, scegli CodeBuild. Scegli il nome di un progetto di CodeBuild build esistente o scegli Crea progetto. In Crea progetto di compilazione, crea un progetto di compilazione, quindi scegli Torna a CodePipeline.

   In variabili di ambiente, selezionare Add environment variable (Aggiungi variabile di ambiente). Ad esempio, inserisci l'ID di esecuzione con la sintassi della variabile `#{codepipeline.PipelineExecutionId}` e l'ID di commit con la sintassi `#{SourceVariables.CommitId}` della variabile.

> **Note**
>
> È possibile immettere la sintassi delle variabili in qualsiasi campo di configurazione delle azioni della procedura guidata.

5. Scegli Crea.
6. Dopo aver creato la pipeline, è possibile visualizzare lo spazio dei nomi creato dalla procedura guidata. Nella pipeline, scegliete l'icona per la fase di cui desiderate visualizzare lo spazio dei nomi. In questo esempio, viene visualizzato lo spazio dei nomi generato automaticamente dell'azione di origine `SourceVariables`.



Per modificare lo spazio dei nomi per un'azione esistente

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.
2. Scegliere la pipeline che si desidera modificare, quindi scegliere Edit (Modifica). Nella fase di origine, scegli Edit stage (Modifica fase). Aggiungi l' CodeCommit azione.
3. In Modifica azione, visualizzare il campo Spazio dei nomi variabile. Se l'azione esistente è stata creata in precedenza o senza utilizzare la procedura guidata, è necessario aggiungere uno spazio dei nomi. Nello spazio dei nomi variabile, immettere un nome dello spazio dei nomi e quindi scegliere Salva.

Per visualizzare le variabili di output

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

2. Dopo aver creato e eseguito correttamente la pipeline, è possibile visualizzare le variabili nella pagina Dettagli esecuzione azione . Per informazioni, consulta Visualizzazione delle variabili (console).

Configurazione delle operazioni per le variabili (CLI)

Quando si utilizza il comando create-pipeline per creare una pipeline o il comando update-pipeline per modificare una pipeline, è possibile fare riferimento o utilizzare variabili nella configurazione di un'azione.

Se lo spazio dei nomi non è specificato, le variabili prodotte dall'azione non sono disponibili per essere referenziate in qualsiasi configurazione di azione downstream.

Per configurare un'azione con uno spazio dei nomi

1. Seguire i passaggi Creare una pipeline in CodePipeline per creare una pipeline utilizzando l'interfaccia della riga di comando. Avviare un file di input per fornire il comando create-pipeline con il parametro `--cli-input-json`. Nella struttura della pipeline aggiungere il parametro `namespace` e specificare un nome, ad esempio `SourceVariables`.

```
. . .
{
          "inputArtifacts": [],
          "name": "Source",
          "region": "us-west-2",
          "namespace": "SourceVariables",
          "actionTypeId": {
            "category": "Source",
            "owner": "AWS",
            "version": "1",
            "provider": "CodeCommit"
          },
          "outputArtifacts": [

. . .
```

2. Salvare il file con un nome come **`MyPipeline.json`**.
3. In un terminale (Linux, macOS o Unix) o dal prompt dei comandi (Windows), esegui il create-pipelinecomando e crea la pipeline.

   Chiama il file creato quando esegui il comando create-pipeline. Per esempio:

```
aws codepipeline create-pipeline --cli-input-json file://MyPipeline.json
```

Per configurare le azioni downstream per utilizzare le variabili

1. Modificare un file di input per fornire il comando update-pipeline con il parametro `--cli-input-json`. Nell'azione downstream, aggiungere la variabile alla configurazione per tale azione. Una variabile è costituita da uno spazio dei nomi e una chiave, separati da un punto. Ad esempio, per aggiungere variabili per l'ID di esecuzione della pipeline e l'ID di commit di origine, specificare lo spazio dei nomi `codepipeline` per la variabile `#{codepipeline.PipelineExecutionId}`. Specificare lo spazio dei nomi `SourceVariables` per la variabile `#{SourceVariables.CommitId}`.

```
{
    "name": "Build",
    "actions": [
        {
            "outputArtifacts": [
                {
                    "name": "BuildArtifacts"
                }
            ],
            "name": "Build",
            "configuration": {
                "EnvironmentVariables": "[{\"name\":\"Execution_ID\",\"value
\":\"#{codepipeline.PipelineExecutionId}\",\"type\":\"PLAINTEXT\"},{\"name\":
\"Commit_ID\",\"value\":\"#{SourceVariables.CommitId}\",\"type\":\"PLAINTEXT\"}]",
                "ProjectName": "env-var-test"
            },
            "inputArtifacts": [
                {
                    "name": "SourceArtifact"
                }
            ],
            "region": "us-west-2",
            "actionTypeId": {
                "provider": "CodeBuild",
                "category": "Build",
                "version": "1",
                "owner": "AWS"
            },
```

```
            "runOrder": 1
        }
    ]
},
```

2. Salvare il file con un nome come **`MyPipeline.json`**.
3. In un terminale (Linux, macOS o Unix) o dal prompt dei comandi (Windows), esegui il create-pipelinecomando e crea la pipeline.

   Chiama il file creato quando esegui il comando create-pipeline. Per esempio:

```
aws codepipeline create-pipeline --cli-input-json file://MyPipeline.json
```

Visualizzazione delle variabili di output

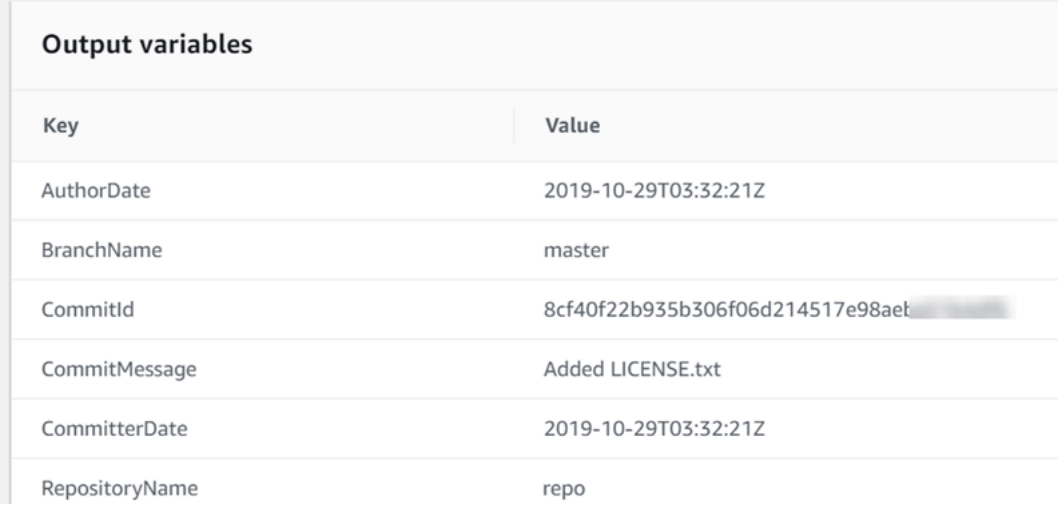
È possibile visualizzare i dettagli di esecuzione dell'azione per visualizzare le variabili per tale azione, specifiche per ogni esecuzione.

Visualizzazione delle variabili (console)

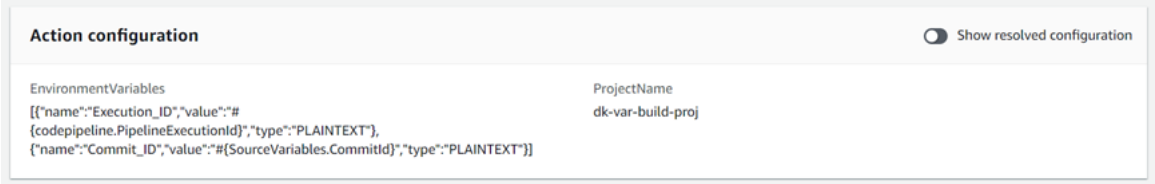
È possibile utilizzare la console per visualizzare le variabili per un'azione.

1. Accedi AWS Management Console e apri la console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home. CodePipeline

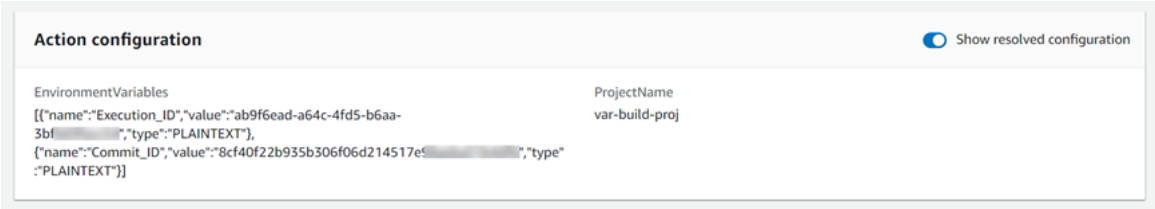
   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account.
2. In Name (Nome), scegli il nome della pipeline.
3. Scegli View history (Visualizza cronologia).
4. Dopo che la pipeline viene eseguita correttamente, è possibile visualizzare le variabili prodotte dall'azione di origine. Scegli View history (Visualizza cronologia). Scegli Source nell'elenco delle azioni per l'esecuzione della pipeline per visualizzare i dettagli di esecuzione dell' CodeCommit azione. Nella schermata dei dettagli dell'azione, visualizzare le variabili in Variabili di output.



5. Dopo che la pipeline viene eseguita correttamente, è possibile visualizzare le variabili consumate dall'azione di compilazione. Scegli View history (Visualizza cronologia). Nell'elenco delle azioni per l'esecuzione della pipeline, scegliete Crea per visualizzare i dettagli di esecuzione dell' CodeBuild azione. Nella pagina dettaglio azione, visualizzare le variabili in Configurazione azione. Viene visualizzato lo spazio dei nomi generato automaticamente.



Per impostazione predefinita, la configurazione dell'azione visualizza la sintassi della variabile. È possibile scegliere Mostra configurazione risolta per attivare o disattivare l'elenco per visualizzare i valori prodotti durante l'esecuzione dell'azione.



Visualizzazione delle variabili (CLI)

È possibile utilizzare il comando list-action-executions per visualizzare le variabili per un'azione.

1. Utilizza il seguente comando:

```
aws codepipeline list-action-executions
```

L'output mostra il parametro `outputVariables` come mostrato qui.

```
"outputVariables": {
                    "BranchName": "main",
                    "CommitMessage": "Updated files for test",
                    "AuthorDate": "2019-11-08T22:24:34Z",
                    "CommitId": "d99b0083cc10EXAMPLE",
                    "CommitterDate": "2019-11-08T22:24:34Z",
                    "RepositoryName": "variables-repo"
                },
```

2. Utilizza il seguente comando:

```
aws codepipeline get-pipeline --name <pipeline-name>
```

Nella configurazione dell' CodeBuild azione, puoi visualizzare le variabili:

```
{
    "name": "Build",
    "actions": [
        {
            "outputArtifacts": [
                {
                    "name": "BuildArtifact"
                }
            ],
            "name": "Build",
            "configuration": {
                "EnvironmentVariables": "[{\"name\":\"Execution_ID\",\"value
\":\"#{codepipeline.PipelineExecutionId}\",\"type\":\"PLAINTEXT\"},{\"name\":
\"Commit_ID\",\"value\":\"#{SourceVariables.CommitId}\",\"type\":\"PLAINTEXT\"}]",
                "ProjectName": "env-var-test"
            },
            "inputArtifacts": [
                {
                    "name": "SourceArtifact"
                }
            ],
```

```
            "region": "us-west-2",
            "actionTypeId": {
                "provider": "CodeBuild",
                "category": "Build",
                "version": "1",
                "owner": "AWS"
            },
            "runOrder": 1
        }
    ]
},
```

Esempio: utilizzo delle variabili nelle approvazioni manuali

Quando si specifica uno spazio dei nomi per un'operazione e tale operazione produce variabili di output, è possibile aggiungere un'approvazione manuale che visualizza le variabili nel messaggio di approvazione. In questo esempio viene illustrato come aggiungere la sintassi delle variabili a un messaggio di approvazione manuale.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account. Scegliere la pipeline a cui si desidera aggiungere l'approvazione.
2. Per modificare la pipeline, scegliere Edit (Modifica). Aggiungere un'approvazione manuale dopo l'operazione di origine. In Action name (Nome operazione), immettere il nome dell'operazione di approvazione.
3. In Action provider (Provider operazione), scegliere Manual approval (Approvazione manuale).
4. In URL for review, aggiungi la sintassi della variabile for `CommitId` al tuo CodeCommit URL. Verificare di utilizzare lo spazio dei nomi assegnato all'operazione di origine. Ad esempio, la sintassi della variabile per un' CodeCommitazione con lo spazio dei nomi `SourceVariables` predefinito è. `#{SourceVariables.CommitId}`

   In Commenti, in`CommitMessage`, inserisci il messaggio di commit:

```
Please approve this change. Commit message: #{SourceVariables.CommitMessage}
```

5. Dopo che la pipeline viene eseguita correttamente, è possibile visualizzare i valori delle variabili nel messaggio di approvazione.

Esempio: utilizzare una BranchName variabile con variabili di CodeBuild ambiente

Quando aggiungi un' CodeBuild azione alla tua pipeline, puoi utilizzare le variabili di CodeBuild ambiente per fare riferimento a una variabile di `BranchName` output proveniente da un'azione di origine upstream. Con una variabile di output proveniente da un'azione in CodePipeline, puoi creare variabili di CodeBuild ambiente personalizzate da utilizzare nei comandi di compilazione.

Questo esempio mostra come aggiungere la sintassi della variabile di output da un'azione di GitHub origine a una variabile di CodeBuild ambiente. La sintassi della variabile di output in questo esempio rappresenta la variabile di output dell'azione di GitHub origine per. `BranchName` Dopo che l'azione è stata eseguita correttamente, la variabile si risolve in modo da mostrare il nome del GitHub ramo.

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

   Vengono visualizzati i nomi di tutte le pipeline associate al tuo AWS account. Scegliere la pipeline a cui si desidera aggiungere l'approvazione.
2. Per modificare la pipeline, scegliere Edit (Modifica). Nella fase che contiene la tua CodeBuild azione, scegli Modifica fase.
3. Scegli l'icona per modificare l' CodeBuild azione.
4. Nella pagina Modifica azione, in Variabili di ambiente, inserisci quanto segue:

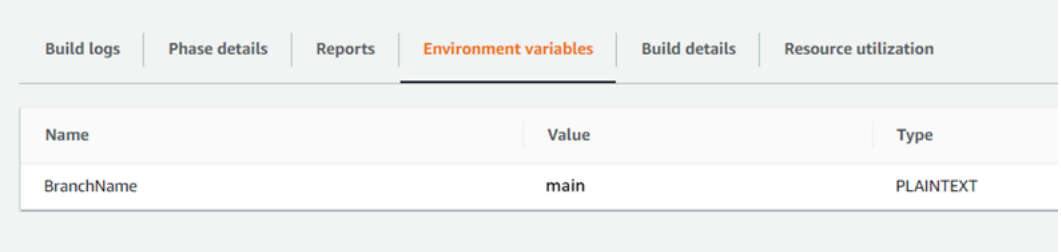
   - In Nome, inserisci un nome per la variabile di ambiente.
   - In Value, inserite la sintassi della variabile per la variabile di output della pipeline, che include lo spazio dei nomi assegnato all'azione di origine. Ad esempio, la sintassi della variabile di output per un' GitHubazione con lo spazio dei nomi predefinito è. `SourceVariables` `#{SourceVariables.BranchName}`
   - In Tipo, scegliete Testo normale.
5. Dopo che la pipeline è stata eseguita correttamente, puoi vedere come la variabile di output risolta è il valore nella variabile di ambiente. Seleziona una delle seguenti opzioni:

   - CodePipeline console: scegli la pipeline, quindi scegli Cronologia. Scegli l'esecuzione della pipeline più recente.

- In Cronologia, scegli il selettore per Sorgente. Questa è l'azione sorgente che genera le variabili GitHub di output. Scegli Visualizza dettagli di esecuzione. In Variabili di output, visualizza l'elenco delle variabili di output generate da questa azione.
- In Cronologia, scegliete il selettore per Build. Questa è l'azione di compilazione che specifica le variabili di CodeBuild ambiente per il tuo progetto di compilazione. Scegli Visualizza i dettagli di esecuzione. In Configurazione delle azioni, visualizza le variabili di CodeBuild ambiente. Scegli Mostra configurazione risolta. Il valore della variabile di ambiente è la variabile `BranchName` di output risolta dall'azione di GitHub origine. In questo esempio, il valore risolto è`main`.

  Per ulteriori informazioni, consulta Visualizzazione delle variabili (console).

- CodeBuild console: scegli il tuo progetto di build e scegli il link per l'esecuzione della build. In Variabili di ambiente, la variabile di output risolta è il valore della variabile di CodeBuild ambiente. In questo esempio, la variabile di ambiente Name is `BranchName` e Value è la variabile di `BranchName` output risolta dall'azione di GitHub origine. In questo esempio, il valore risolto è`main`.



Lavorare con le transizioni di fase in CodePipeline

Le transizioni sono collegamenti tra le fasi della pipeline che possono essere disabilitate o abilitate. (impostazione predefinita). Quando riabiliti una transizione disabilitata, la revisione più recente viene eseguita nelle fasi rimanenti della pipeline, a meno che non siano passati più di 30 giorni. L'esecuzione delle pipeline non riprende per una transizione che è stata disabilitata da oltre 30 giorni, a meno che non sia rilevata una nuova modifica o che la pipeline non sia riavviata manualmente.

È possibile utilizzare la AWS CodePipeline console o il AWS CLI per disabilitare o abilitare le transizioni tra le fasi di una pipeline.

> **Note**
>
> Puoi utilizzare un'operazione di approvazione per sospendere l'esecuzione di una pipeline finché la prosecuzione non viene approvata manualmente. Per ulteriori informazioni, consulta Gestisci le azioni di approvazione in CodePipeline.

Argomenti

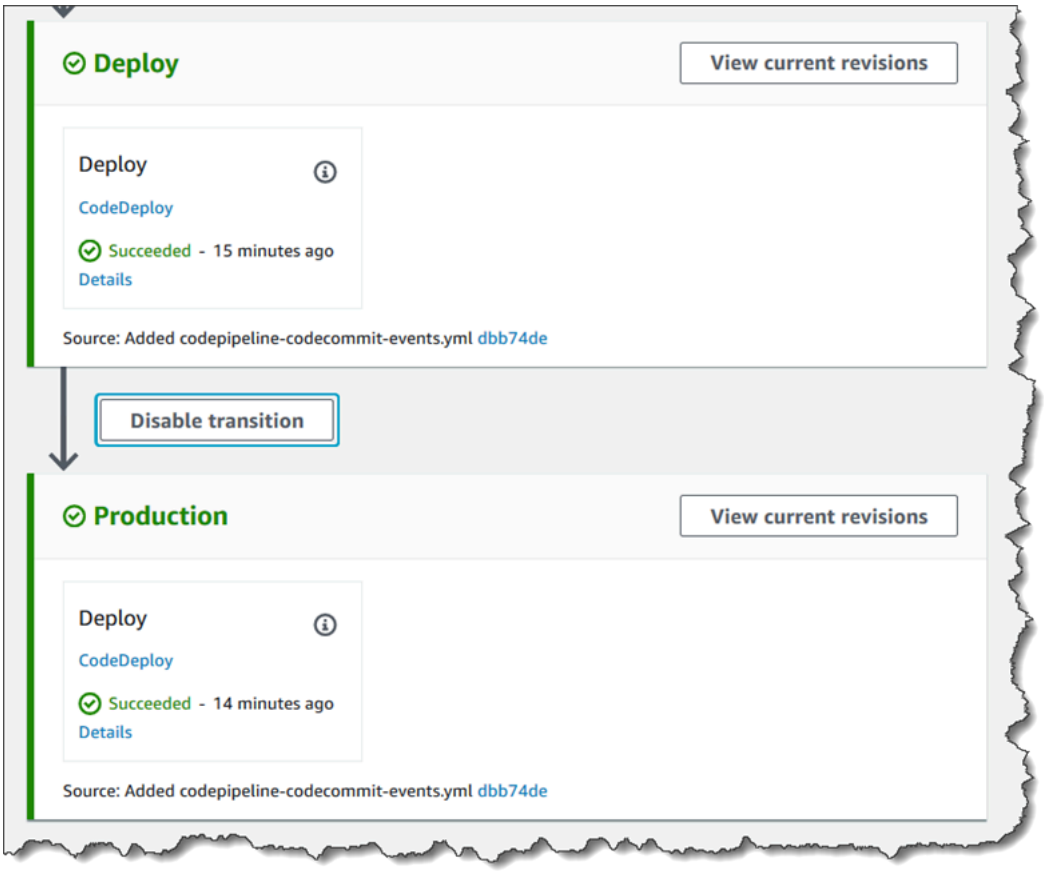
Disabilitazione o abilitazione delle transizioni (console)

Per disabilitare o abilitare le transizioni in una pipeline

1. Accedi AWS Management Console e apri la CodePipeline console all'indirizzo http://console.aws.amazon.com/codesuite/codepipeline/home.

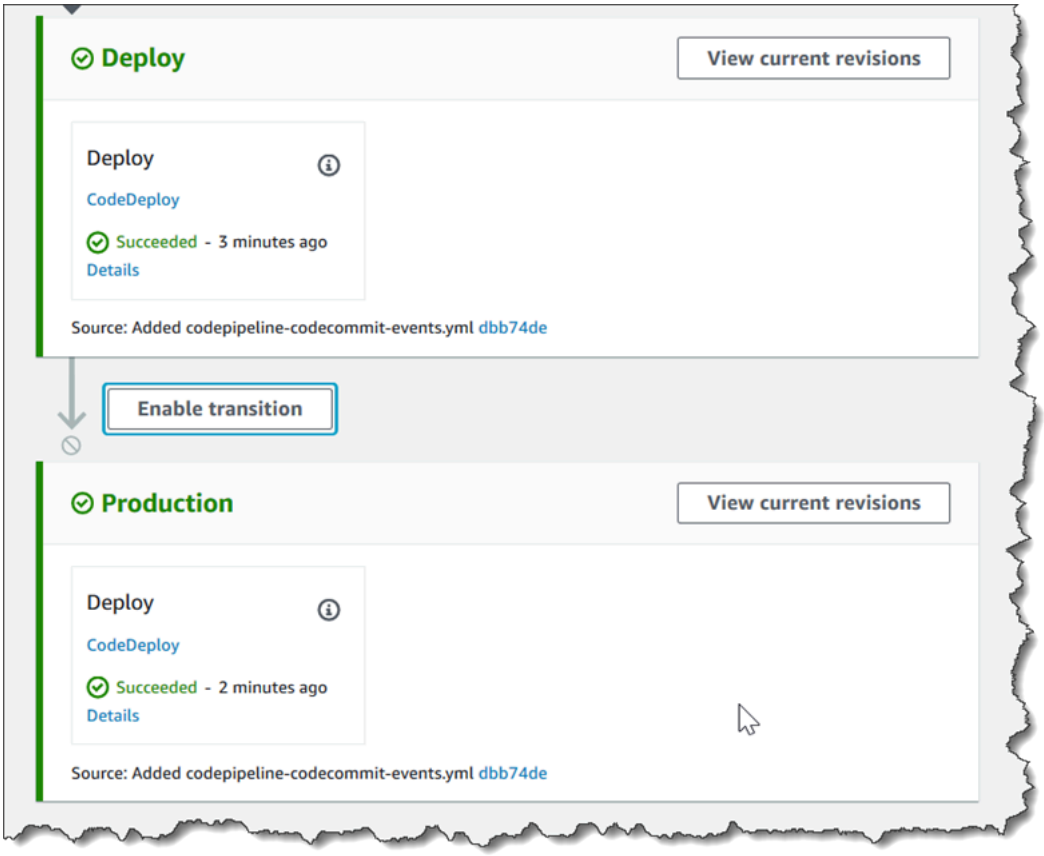
   Vengono mostrati i nomi di tutte le pipeline associate al tuo account AWS .
2. In Name (Nome), scegliere il nome della pipeline per la quale si desidera abilitare o disabilitare le transizioni. Questa azione apre una visualizzazione dettagliata della pipeline. comprese le transizioni tra le fasi della pipeline.
3. Individuare la freccia dopo l'ultima fase da eseguire, quindi scegliere il pulsante accanto a essa. Ad esempio nella seguente pipeline, se si desidera che vengano eseguite le operazioni

della fase Staging (Gestione temporanea) ma non quelle della fase Production (Produzione), è possibile scegliere il pulsante Disable transition (Disabilita transizione) posto tra le due fasi:



4. Nella finestra di dialogo Disable transition (Disabilita transizione), specificare il motivo della disabilitazione della transizione, quindi scegliere Disable (Disabilita).

   Il pulsante cambia per evidenziare che le transizioni tra la fase sopra la freccia e la fase successiva sono disabilitate. Tutte le revisioni già in elaborazione nelle fasi successive alla transizione disabilitata continueranno l'esecuzione nella pipeline, ma eventuali revisioni successive non vanno oltre la transizione disabilitata.



5. Scegliere il pulsante Enable transition (Abilita transizione) accanto alla freccia. Nella finestra di dialogo Enable transition (Abilita transazione), scegliere Enable (Abilita). La pipeline abilita immediatamente la transizione tra le due fasi. Se dopo la disabilitazione della transizione sono state eseguite revisioni nelle fasi precedenti, dopo pochi istanti la pipeline inizia a eseguire la revisione più recente nelle fasi successive alla transizione precedentemente disabilitata. La pipeline esegue la revisione in tutte le fasi rimanenti.

> **Note**
>
> Potrebbero essere necessari alcuni secondi prima che le modifiche vengano visualizzate nella CodePipeline console dopo aver abilitato la transizione.

Disabilitazione o abilitazione delle transizioni (CLI)

Per disabilitare una transizione tra fasi utilizzando il AWS CLI, esegui il disable-stage-transition comando. Per abilitare una transizione disabilitata, eseguire il comando enable-stage-transition.

Per disabilitare una transizione

1. Apri un terminale (Linux, macOS o Unix) o il prompt dei comandi (Windows) e usa AWS CLI per eseguire il disable-stage-transitioncomando, specificando il nome della pipeline, il nome della fase in cui desideri disabilitare le transizioni, il tipo di transizione e il motivo per cui stai disabilitando le transizioni verso quella fase. A differenza dell'utilizzo della console, è necessario specificare se si desidera disabilitare le transizioni verso la fase (in ingresso) o le transizioni che hanno origine dalla fase dopo il completamento di tutte le operazioni (in uscita).

   Ad esempio, per disabilitare la transizione a una fase denominata *Staging* in una pipeline denominata *MyFirstPipeline*, digitate un comando simile al seguente:

   ```
   aws codepipeline disable-stage-transition --pipeline-name MyFirstPipeline --stage-
   name Staging --transition-type Inbound --reason "My Reason"
   ```

   Il comando non restituisce alcun risultato.
2. Per verificare che la transizione sia stata disabilitata, visualizzate la pipeline nella CodePipeline console o eseguite il comando. get-pipeline-state Per ulteriori informazioni, consulta Visualizza le pipeline (console) e Visualizzazione dei dettagli e della cronologia della pipeline (CLI).

Per abilitare una transizione

1. Apri un terminale (Linux, macOS o Unix) o il prompt dei comandi (Windows) e usa AWS CLI per eseguire il enable-stage-transitioncomando, specificando il nome della pipeline, il nome dello stadio in cui desideri abilitare le transizioni e il tipo di transizione.

   Ad esempio, per abilitare la transizione a una fase denominata *Staging* in una pipeline denominata *MyFirstPipeline*, è necessario digitare un comando simile al seguente:

   ```
   aws codepipeline enable-stage-transition --pipeline-name MyFirstPipeline --stage-
   name Staging  --transition-type Inbound
   ```

   Il comando non restituisce alcun risultato.
2. Per verificare che la transizione sia stata disabilitata, visualizzate la pipeline nella CodePipeline console o eseguite il comando. get-pipeline-state Per ulteriori informazioni, consultare Visualizza le pipeline (console) e Visualizzazione dei dettagli e della cronologia della pipeline (CLI).

Monitoraggio di pipeline

Il monitoraggio è una parte importante per mantenere l'affidabilità, la disponibilità e le prestazioni di AWS CodePipeline. È necessario raccogliere i dati di monitoraggio da tutte le parti della AWS soluzione in modo da poter eseguire più facilmente il debug di un errore multipunto, se si verifica. Prima di iniziare il monitoraggio, è opportuno creare un piano di monitoraggio che risponde alle seguenti domande:

- Quali sono gli obiettivi del monitoraggio?
- Quali risorse verranno monitorate?
- Con quale frequenza eseguirai il monitoraggio di queste risorse?
- Quali sono gli strumenti di monitoraggio disponibili?
- Chi eseguirà i processi di monitoraggio?
- Chi deve ricevere una notifica se si verifica un problema?

È possibile utilizzare i seguenti strumenti per monitorare le CodePipeline pipeline e le relative risorse:

- EventBridge eventi del bus di eventi: puoi monitorare CodePipeline gli eventi in EventBridge, il che rileva i cambiamenti nello stato di esecuzione della pipeline, della fase o dell'azione. EventBridge indirizza tali dati verso obiettivi come Amazon AWS Lambda Simple Notification Service. EventBridge gli eventi sono gli stessi che appaiono in Amazon CloudWatch Events.
- Notifiche per gli eventi della pipeline nella console Developer Tools: puoi monitorare CodePipeline gli eventi con le notifiche configurate nella console e quindi creare un argomento e un abbonamento ad Amazon Simple Notification Service. Per ulteriori informazioni, consulta Cosa sono le notifiche nella Guida per l'utente della Developer Tools Console.
- AWS CloudTrail— CloudTrail Da utilizzare per acquisire le chiamate API effettuate da o per conto di CodePipeline nel proprio AWS account e inviare i file di registro a un bucket Amazon S3. Puoi scegliere di CloudWatch pubblicare le notifiche di Amazon SNS quando vengono consegnati nuovi file di log in modo da poter agire rapidamente.
- Console e CLI: è possibile utilizzare la CodePipeline console e la CLI per visualizzare i dettagli sullo stato di una pipeline o sull'esecuzione di una particolare pipeline.

Argomenti

- Riferimento per il bucket segnaposto eventi
- Registrazione delle chiamate CodePipeline API con AWS CloudTrail

Monitoraggio CodePipeline degli eventi

Puoi monitorare CodePipeline gli eventi in EventBridge, il che fornisce un flusso di dati in tempo reale dalle tue applicazioni, applicazioni software-as-a-service (SaaS) e. Servizi AWS EventBridge indirizza tali dati verso obiettivi come Amazon AWS Lambda Simple Notification Service. Questi eventi sono gli stessi che compaiono in Amazon CloudWatch Events, che fornisce un flusso quasi in tempo reale di eventi di sistema che descrivono i cambiamenti nelle AWS risorse. Per ulteriori informazioni, consulta What Is Amazon EventBridge? nella Amazon EventBridge User Guide.

> **Note**
>
> Amazon EventBridge è il modo preferito per gestire i tuoi eventi. Amazon CloudWatch Events e Amazon EventBridge sono lo stesso servizio e la stessa API di base, ma EventBridge offrono più funzionalità. Le modifiche che apporti in CloudWatch Events o EventBridge verranno visualizzate in ogni console.

Gli eventi sono composti da regole. Una regola viene configurata scegliendo quanto segue:

- Modello di evento. Ogni regola è espressa come un pattern di eventi con l'origine e il tipo di eventi da monitorare e le destinazioni degli eventi. Per monitorare gli eventi, si crea una regola con il servizio che si sta monitorando come origine dell'evento, ad esempio CodePipeline. Ad esempio, è possibile creare una regola con un pattern di eventi da utilizzare CodePipeline come fonte di eventi per attivare la regola in caso di cambiamenti nello stato di una pipeline, di una fase o di un'azione.
- Destinazioni. La nuova regola riceve un servizio selezionato come la destinazione dell'evento. Potresti voler configurare un servizio di destinazione per inviare notifiche, acquisire informazioni sullo stato, intraprendere azioni correttive, avviare eventi o intraprendere altre azioni. Quando aggiungi il tuo target, devi anche concedere le autorizzazioni per consentirgli EventBridge di richiamare il servizio di destinazione selezionato.

Ogni tipo di evento di modifica dello stato di esecuzione genera notifiche con contenuto del messaggio specifico, dove:

- La `version` voce iniziale mostra il numero di versione dell'evento.
- La voce `version` nella pipeline `detail` mostra il numero di versione della struttura della pipeline.
- La voce `execution-id` nella pipeline `detail` mostra l'ID di esecuzione per l'esecuzione della pipeline che ha causato la modifica dello stato. Fate riferimento alla chiamata GetPipelineExecution API nell'AWS CodePipeline API Reference.
- La `pipeline-execution-attempt` voce mostra il numero di tentativi, o nuovi tentativi, per l'ID di esecuzione specifico.

CodePipeline segnala un evento EventBridge ogni volta che lo stato di una risorsa nell'utente Account AWS cambia. Gli eventi vengono emessi su at-least-once base garantita per le seguenti risorse:

- Esecuzioni pipeline
- Esecuzioni sceniche
- Esecuzioni di operazioni

Gli eventi vengono emessi EventBridge con il modello e lo schema degli eventi descritti sopra. Per gli eventi elaborati, ad esempio gli eventi ricevuti tramite notifiche configurate nella console Developer Tools, il messaggio relativo all'evento include campi relativi al modello di evento con alcune variazioni. Ad esempio, il `detail-type` campo viene convertito in`detailType`. Per ulteriori informazioni, consulta la chiamata PutEvents API nell'Amazon EventBridge API Reference.

Gli esempi seguenti mostrano eventi per CodePipeline. Ove possibile, ogni esempio mostra lo schema per un evento emesso insieme allo schema per un evento elaborato.

Argomenti

Tipi di dettaglio

Quando configurate gli eventi da monitorare, potete scegliere il tipo di dettaglio per l'evento.

Puoi configurare le notifiche da inviare quando lo stato cambia per:

- Pipeline specificate o tutte le pipeline. Esegui il controllo utilizzando `"detail-type": "CodePipeline Pipeline Execution State Change"`.
- Fasi specificate o tutte le fasi, all'interno di una pipeline specificata o tutte le pipeline. Esegui il controllo utilizzando `"detail-type": "CodePipeline Stage Execution State Change"`.
- Operazioni specificate o tutte le operazioni, all'interno di una determinata fase o di tutte le fasi, all'interno di una pipeline specificata o di tutte le pipeline. Esegui il controllo utilizzando `"detail-type": "CodePipeline Action Execution State Change"`.

> **Note**
>
> Gli eventi emessi da EventBridge contengono il `detail-type` parametro, che viene convertito in `detailType` quando gli eventi vengono elaborati.

Tipo di dettaglio	Stato	Descrizione
CodePipeline Modifica dello stato di esecuzione della pipeline	CANCELED (ANNULLATO)	L'esecuzione della pipeline è stata annullata perché la struttura della pipeline è stata aggiornata.
	Non riuscito	L'esecuzione della pipeline non è stata completata.
	RIPRESA	È stato eseguito un nuovo tentativo di esecuzione della pipeline non riuscita in risposta alla chiamata API `RetryStageExecution` .
	AVVIATA	L'esecuzione della pipeline è attualmente in corso.
	STOPPED	Il processo di arresto è completo e l'esecuzione della pipeline viene interrotta.
	STOPPING	L'esecuzione della pipeline si interrompe a causa di una richiesta di arrestare e attendere o interrompere e abbandonare l'esecuzione della pipeline.
	RIUSCITA	L'esecuzione della pipeline è stata completata.

Tipo di dettaglio	Stato	Descrizione
	SOSTITUITA	Mentre l'esecuzione di questa pipeline era in attesa del completamento della fase successiva, una nuova esecuzione della pipeline è avanzata e ha continuato nella pipeline.
CodePipeline Modifica dello stato di esecuzione della fase	CANCELED (ANNULLATO)	La fase è stata annullata perché la struttura della pipeline è stata aggiornata.
	Non riuscito	La fase non è stata completata.
	RIPRESA	È stato eseguito un nuovo tentativo di esecuzione della fase non riuscita in risposta alla chiamata API `RetryStageExecution` .
	AVVIATA	La fase è attualmente in esecuzione.
	STOPPED	Il processo di arresto è completo e l'esecuzione dello stage viene interrotta.
	STOPPING	L'esecuzione dello stage si interrompe a causa di una richiesta di arrestare e attendere o interrompere e abbandonare l'esecuzione della pipeline.
	RIUSCITA	La fase è stata completata.
CodePipeline Modifica dello stato di esecuzione dell'azione	ABBANDONATO	L'azione è abbandonata a causa di una richiesta di arrestare e abbandonare l'esecuzione della pipeline.
	CANCELED (ANNULLATO)	L'operazione è stata annullata perché la struttura della pipeline è stata aggiornata.
	Non riuscito	Per le operazioni di approvazione, lo stato NON RIUSCITA significa che l'operazione è stata rifiutata dal revisore o che non è riuscita a causa di un errore di configurazione dell'operazione.
	AVVIATA	L'operazione è attualmente in esecuzione.

Tipo di dettaglio	Stato	Descrizione
	RIUSCITA	L'operazione è stata completata.

eventi a livello di pipeline

Gli eventi a livello di pipeline vengono emessi quando si verifica un cambiamento di stato per l'esecuzione di una pipeline.

Argomenti

Evento Pipeline STARTED

Quando inizia l'esecuzione di una pipeline, emette un evento che invia notifiche con il seguente contenuto. Questo esempio riguarda la pipeline denominata `"myPipeline"` nella regione. `us-east-1` Il `id` campo rappresenta l'ID dell'evento e il `account` campo rappresenta l'ID dell'account in cui viene creata la pipeline.

Emitted event

```
{
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Pipeline Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2020-01-24T22:03:07Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:codepipeline:us-east-1:123456789012:myPipeline"
    ],
```

```
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "execution-trigger": {
            "trigger-type": "StartPipelineExecution",
            "trigger-detail": "arn:aws:sts::123456789012:assumed-role/Admin/my-user"
        },
        "state": "STARTED",
        "version": 1.0,
        "pipeline-execution-attempt": 1.0
    }
}
```

Processed event

```
{
    "account": "123456789012",
    "detailType": "CodePipeline Pipeline Execution State Change",
    "region": "us-east-1",
    "source": "aws.codepipeline",
    "time": "2021-06-24T00:44:50Z",
    "notificationRuleArn": "arn:aws:codestar-notifications:us-
east-1:123456789012:notificationrule/a69c62c21EXAMPLE",
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "execution-trigger": {
            "trigger-type": "StartPipelineExecution",
            "trigger-detail": "arn:aws:sts::123456789012:assumed-role/Admin/my-user"
        },
        "state": "STARTED",
        "version": 1.0,
        "pipeline-execution-attempt": 1.0
    },
    "resources": [
        "arn:aws:codepipeline:us-east-1:123456789012:myPipeline"
    ],
    "additionalAttributes": {}
}
```

Evento Pipeline STOPPING

Quando l'esecuzione di una pipeline si interrompe, emette un evento che invia notifiche con il seguente contenuto. Questo esempio riguarda la pipeline denominata `myPipeline` nella regione. `us-west-2`

```
{
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Pipeline Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2020-01-24T22:02:20Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "state": "STOPPING",
        "version": 3.0,
        "pipeline-execution-attempt": 1.0
        "stop-execution-comments": "Stopping the pipeline for an update"
    }
}
```

Evento Pipeline SUCCEEDED

Quando l'esecuzione di una pipeline ha esito positivo, emette un evento che invia notifiche con il seguente contenuto. Questo esempio riguarda la pipeline denominata `myPipeline` nella regione. `us-east-1`

Emitted event

```
{
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Pipeline Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
```

```
    "time": "2020-01-24T22:03:44Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:codepipeline:us-east-1:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "state": "SUCCEEDED",
        "version": 3.0,
        "pipeline-execution-attempt": 1.0
    }
}
```

Processed event

```
{
    "account": "123456789012",
    "detailType": "CodePipeline Pipeline Execution State Change",
    "region": "us-east-1",
    "source": "aws.codepipeline",
    "time": "2021-06-30T22:13:51Z",
    "notificationRuleArn": "arn:aws:codestar-notifications:us-
west-2:123456789012:notificationrule/a69c62c21EXAMPLE",
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "state": "SUCCEEDED",
        "version": 1.0,
        "pipeline-execution-attempt": 1.0
    },
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "additionalAttributes": {}
}
```

Pipeline RIUSCITA (esempio con tag Git)

Quando l'esecuzione di una pipeline ha una fase che è stata ritentata e ha avuto successo, emette un evento che invia notifiche con il seguente contenuto. Questo esempio è per la pipeline denominata `myPipeline` nella `eu-central-1` regione in cui `execution-trigger` è configurata per i tag Git.

> **Note**
>
> Il `execution-trigger` campo avrà una delle due opzioni `tag-name` oppure`branch-name`, a seconda del tipo di evento che ha attivato la pipeline.

```
{
    "version": "0",
    "id": "b128b002-09fd-4574-4eba-27152726c777",
    "detail-type": "CodePipeline Pipeline Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2023-10-26T13:50:53Z",
    "region": "eu-central-1",
    "resources": [
        "arn:aws:codepipeline:eu-central-1:123456789012:BuildFromTag"
    ],
    "detail": {
        "pipeline": "BuildFromTag",
        "execution-id": "e17b5773-cc0d-4db2-9ad7-594c73888de8",
        "start-time": "2023-10-26T13:49:39.208Z",
        "execution-trigger": {
            "author-display-name": "Mary Major",
            "full-repository-name": "mmajor/sample-project",
            "provider-type": "GitLab",
            "author-email": "email_address",
            "commit-message": "Update file README.md",
            "author-date": "2023-08-16T21:08:08Z",
            "tag-name": "gitlab-v4.2.1",
            "commit-id": "commit_ID",
            "connection-arn": "arn:aws:codestar-connections:eu-
central-1:123456789012:connection/0f5b706a-1a1d-46c5-86b6-f177321bcfb2",
            "author-id": "Mary Major"
        },
        "state": "SUCCEEDED",
        "version": 32.0,
```

```
        "pipeline-execution-attempt": 1.0
    }
}
```

Evento Pipeline FAILED

Quando l'esecuzione di una pipeline fallisce, emette un evento che invia notifiche con il seguente contenuto. Questo esempio riguarda la pipeline denominata `"myPipeline"` nella regione. `us-west-2`

Emitted event

```
{
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Pipeline Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2020-01-31T18:55:43Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "state": "FAILED",
        "version": 4.0,
        "pipeline-execution-attempt": 1.0
    }
}
```

Processed event

```
{
    "account": "123456789012",
    "detailType": "CodePipeline Pipeline Execution State Change",
    "region": "us-west-2",
    "source": "aws.codepipeline",
    "time": "2021-06-24T00:46:16Z",
    "notificationRuleArn": "arn:aws:codestar-notifications:us-
west-2:123456789012:notificationrule/a69c62c21EXAMPLE",
```

```
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "state": "FAILED",
        "version": 1.0,
        "pipeline-execution-attempt": 1.0
    },
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "additionalAttributes": {
        "failedActionCount": 1,
        "failedActions": [
            {
                "action": "Deploy",
                "additionalInformation": "Deployment <ID> failed"
            }
        ],
        "failedStage": "Deploy"
    }
```

Pipeline FAILED (esempio con tag Git)

A meno che non fallisca nella fase di origine, per una pipeline configurata con trigger, emette un evento che invia notifiche con il seguente contenuto. Questo esempio è per la pipeline denominata `myPipeline` nella `eu-central-1` regione in cui `execution-trigger` è configurata per i tag Git.

> **Note**
>
> Il `execution-trigger` campo avrà una delle due opzioni `tag-name` oppure`branch-name`, a seconda del tipo di evento che ha attivato la pipeline.

Emitted event

```
{
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Pipeline Execution State Change",
    "source": "aws.codepipeline",
```

```
    "account": "123456789012",
    "time": "2020-01-31T18:55:43Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "execution-trigger": {
            "author-display-name": "Mary Major",
            "full-repository-name": "mmajor/sample-project",
            "provider-type": "GitLab",
            "author-email": "email_address",
            "commit-message": "Update file README.md",
            "author-date": "2023-08-16T21:08:08Z",
            "tag-name": "gitlab-v4.2.1",
            "commit-id": "commit_ID",
            "connection-arn": "arn:aws:codestar-connections:eu-
central-1:123456789012:connection/0f5b706a-1a1d-46c5-86b6-f177321bcfb2",
            "author-id": "Mary Major"
        },
        "state": "FAILED",
        "version": 4.0,
        "pipeline-execution-attempt": 1.0
    }
}
```

Processed event

```
{
    "account": "123456789012",
    "detailType": "CodePipeline Pipeline Execution State Change",
    "region": "us-west-2",
    "source": "aws.codepipeline",
    "time": "2021-06-24T00:46:16Z",
    "notificationRuleArn": "arn:aws:codestar-notifications:us-
west-2:123456789012:notificationrule/a69c62c21EXAMPLE",
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
```

```
        "execution-trigger": {
            "author-display-name": "Mary Major",
            "full-repository-name": "mmajor/sample-project",
            "provider-type": "GitLab",
            "author-email": "email_address",
            "commit-message": "Update file README.md",
            "author-date": "2023-08-16T21:08:08Z",
            "tag-name": "gitlab-v4.2.1",
            "commit-id": "commit_ID",
            "connection-arn": "arn:aws:codestar-connections:eu-
central-1:123456789012:connection/0f5b706a-1a1d-46c5-86b6-f177321bcfb2",
            "author-id": "Mary Major"
        },
        "state": "FAILED",
        "version": 1.0,
        "pipeline-execution-attempt": 1.0
    },
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "additionalAttributes": {
        "failedActionCount": 1,
        "failedActions": [
            {
                "action": "Deploy",
                "additionalInformation": "Deployment <ID> failed"
            }
        ],
        "failedStage": "Deploy"
    }
```

Eventi a livello di fase

Gli eventi a livello di fase vengono emessi quando si verifica un cambio di stato per l'esecuzione di una fase.

Argomenti

- Evento Stage STARTED
- Evento Stage STOPPING
- Evento Stage STOPED

- Fase RIPRESA dopo l'evento di riprova

Evento Stage STARTED

Quando inizia l'esecuzione di una fase, emette un evento che invia notifiche con il seguente contenuto. Questo esempio è per la pipeline denominata "myPipeline" nella us-east-1 regione, per lo stage. Prod

Emitted event

```
{
    "version": "0",
    "id": 01234567-EXAMPLE,
    "detail-type": "CodePipeline Stage Execution State Change",
    "source": "aws.codepipeline",
    "account": 123456789012,
    "time": "2020-01-24T22:03:07Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:codepipeline:us-east-1:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "version": 1.0,
        "execution-id": 12345678-1234-5678-abcd-12345678abcd,
        "start-time": "2023-10-26T13:49:39.208Z",
        "stage": "Prod",
        "state": "STARTED",
        "pipeline-execution-attempt": 1.0
    }
}
```

Processed event

```
{
    "account": "123456789012",
    "detailType": "CodePipeline Stage Execution State Change",
    "region": "us-east-1",
    "source": "aws.codepipeline",
    "time": "2021-06-24T00:45:40Z",
    "notificationRuleArn": "arn:aws:codestar-notifications:us-
west-2:123456789012:notificationrule/a69c62c21EXAMPLE",
```

```
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "stage": "Source",
        "state": "STARTED",
        "version": 1.0,
        "pipeline-execution-attempt": 0.0
    },
    "resources": [
        "arn:aws:codepipeline:us-east-1:123456789012:myPipeline"
    ],
    "additionalAttributes": {
        "sourceActions": [
            {
                "sourceActionName": "Source",
                "sourceActionProvider": "CodeCommit",
                "sourceActionVariables": {
                    "BranchName": "main",
                    "CommitId": "<ID>",
                    "RepositoryName": "my-repo"
                }
            }
        ]
    }
}
```

Evento Stage STOPPING

Quando l'esecuzione di una fase si interrompe, emette un evento che invia notifiche con il seguente contenuto. Questo esempio è per la pipeline denominata `myPipeline` nella `us-west-2` regione, per lo stage. `Deploy`

```
{
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Stage Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2020-01-24T22:02:20Z",
    "region": "us-west-2",
    "resources": [
```

```
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "stage": "Deploy",
        "state": "STOPPING",
        "version": 3.0,
        "pipeline-execution-attempt": 1.0
    }
}
```

Evento Stage STOPED

Quando l'esecuzione di una fase viene interrotta, emette un evento che invia notifiche con il seguente contenuto. Questo esempio è per la pipeline denominata `myPipeline` nella `us-west-2` regione, per lo stage. `Deploy`

```
{
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Stage Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2020-01-31T18:21:39Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:49:39.208Z",
        "stage": "Deploy",
        "state": "STOPPED",
        "version": 3.0,
        "pipeline-execution-attempt": 1.0
    }
}
```

Fase RIPRESA dopo l'evento di riprova

Quando l'esecuzione di una fase viene ripresa e una fase è stata ritentata, emette un evento che invia notifiche con il seguente contenuto.

Quando una fase è stata ritentata, il `stage-last-retry-attempt-time` campo viene visualizzato, come mostrato nell'esempio. Il campo viene visualizzato in tutti gli eventi dello stage se è stato eseguito un nuovo tentativo.

> **Note**
>
> Il `stage-last-retry-attempt-time` campo sarà presente in tutti gli eventi della fase successivi dopo che una fase sarà stata riprovata.

```
{
    "version": "0",
    "id": "38656bcd-a798-5f92-c738-02a71be484e1",
    "detail-type": "CodePipeline Stage Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2023-10-26T14:14:56Z",
    "region": "eu-central-1",
    "resources": [
        "arn:aws:codepipeline:eu-central-1:123456789012:BuildFromTag"
    ],
    "detail": {
        "pipeline": "BuildFromTag",
        "execution-id": "05dafb6a-5a56-4951-a858-968795364846",
        "stage-last-retry-attempt-time": "2023-10-26T14:14:56.305Z",
        "stage": "Build",
        "state": "RESUMED",
        "version": 32.0,
        "pipeline-execution-attempt": 2.0
    }
}
```

Eventi a livello di azione

Gli eventi a livello di azione vengono emessi quando si verifica un cambiamento di stato per l'esecuzione di un'azione.

Argomenti

Evento Action STARTED

Quando inizia l'esecuzione di un'azione, emette un evento che invia notifiche con il seguente contenuto. Questo esempio riguarda la pipeline denominata `myPipeline` nella `us-east-1` regione, per l'azione di distribuzione. `myAction`

Emitted event

```
{
    "version": "0",
    "id": 01234567-EXAMPLE,
    "detail-type": "CodePipeline Action Execution State Change",
    "source": "aws.codepipeline",
    "account": 123456789012,
    "time": "2020-01-24T22:03:07Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:codepipeline:us-east-1:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": 12345678-1234-5678-abcd-12345678abcd,
        "start-time": "2023-10-26T13:51:09.981Z",
        "stage": "Prod",
        "action-execution-id": "47f821c5-a902-44b2-ae61-b878d31ecd21",
        "action": "myAction",
        "state": "STARTED",
        "type": {
            "owner": "AWS",
            "category": "Deploy",
            "provider": "CodeDeploy",
            "version": "1"
        },
        "version": 2.0
        "pipeline-execution-attempt": 1.0
```

```
        "input-artifacts": [
            {
                "name": "SourceArtifact",
                "s3location": {
                    "bucket": "codepipeline-us-east-1-BUCKETEXAMPLE",
                    "key": "myPipeline/SourceArti/KEYEXAMPLE"
                }
            }
        ]
    }
}
```

Processed event

```
{
    "account": "123456789012",
    "detailType": "CodePipeline Action Execution State Change",
    "region": "us-west-2",
    "source": "aws.codepipeline",
    "time": "2021-06-24T00:45:44Z",
    "notificationRuleArn": "arn:aws:codestar-notifications:us-
west-2:123456789012:notificationrule/a69c62c21EXAMPLE",
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:51:09.981Z",
        "stage": "Deploy",
        "action-execution-id": "47f821c5-a902-44b2-ae61-b878d31ecd21",
        "action": "Deploy",
        "input-artifacts": [
            {
                "name": "SourceArtifact",
                "s3location": {
                    "bucket": "codepipeline-us-east-1-EXAMPLE",
                    "key": "myPipeline/SourceArti/EXAMPLE"
                }
            }
        ],
        "state": "STARTED",
        "region": "us-east-1",
        "type": {
            "owner": "AWS",
            "provider": "CodeDeploy",
```

```
            "category": "Deploy",
            "version": "1"
        },
        "version": 1.0,
        "pipeline-execution-attempt": 1.0
    },
    "resources": [
        "arn:aws:codepipeline:us-east-1:123456789012:myPipeline"
    ],
    "additionalAttributes": {}
}
```

Evento Azione RIUSCITA

Quando l'esecuzione di un'azione ha esito positivo, emette un evento che invia notifiche con il seguente contenuto. Questo esempio riguarda la pipeline denominata `"myPipeline"` nella `us-west-2` regione, per l'azione di origine. `"Source"` Per questo tipo di evento, sono disponibili due `region` campi diversi. Il `region` campo evento specifica la regione per l'evento della pipeline. Il `region` campo sotto la `detail` sezione specifica la regione per l'azione.

Emitted event

```
{
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Action Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2020-01-24T22:03:11Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:51:09.981Z",
        "stage": "Source",
        "execution-result": {
            "external-execution-url": "https://us-west-2.console.aws.amazon.com/
codecommit/home#/repository/my-repo/commit/8cf40f2EXAMPLE",
            "external-execution-summary": "Added LICENSE.txt",
```

```
            "external-execution-id": "8cf40fEXAMPLE"
        },
        "output-artifacts": [
            {
                "name": "SourceArtifact",
                "s3location": {
                    "bucket": "codepipeline-us-west-2-BUCKETEXAMPLE",
                    "key": "myPipeline/SourceArti/KEYEXAMPLE"
                }
            }
        ],
        "action-execution-id": "47f821c5-a902-44b2-ae61-b878d31ecd21",
        "action": "Source",
        "state": "SUCCEEDED",
        "region": "us-west-2",
        "type": {
            "owner": "AWS",
            "provider": "CodeCommit",
            "category": "Source",
            "version": "1"
        },
        "version": 3.0,
        "pipeline-execution-attempt": 1.0
    }
}
```

Processed event

```
{
    "account": "123456789012",
    "detailType": "CodePipeline Action Execution State Change",
    "region": "us-west-2",
    "source": "aws.codepipeline",
    "time": "2021-06-24T00:45:44Z",
    "notificationRuleArn": "arn:aws:codestar-notifications:us-
west-2:ACCOUNT:notificationrule/a69c62c21EXAMPLE",
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "arn:aws:codepipeline:us-west-2:123456789012:myPipeline",
        "start-time": "2023-10-26T13:51:09.981Z",
        "stage": "Source",
        "execution-result": {
```

```
            "external-execution-url": "https://us-west-2.console.aws.amazon.com/
codecommit/home#/repository/my-repo/commit/8cf40f2EXAMPLE",
            "external-execution-summary": "Edited index.html",
            "external-execution-id": "36ab3ab7EXAMPLE"
        },
        "output-artifacts": [
            {
                "name": "SourceArtifact",
                "s3location": {
                    "bucket": "codepipeline-us-west-2-EXAMPLE",
                    "key": "myPipeline/SourceArti/EXAMPLE"
                }
            }
        ],
        "action-execution-id": "47f821c5-a902-44b2-ae61-b878d31ecd21",
        "action": "Source",
        "state": "SUCCEEDED",
        "region": "us-west-2",
        "type": {
            "owner": "AWS",
            "provider": "CodeCommit",
            "category": "Source",
            "version": "1"
        },
        "version": 1.0,
        "pipeline-execution-attempt": 1.0
    },
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "additionalAttributes": {}
}
```

Azione (evento FALLITO)

Quando l'esecuzione di un'azione fallisce, emette un evento che invia notifiche con il seguente contenuto. Questo esempio riguarda la pipeline denominata `"myPipeline"` nella `us-west-2` regione, per l'azione. `"Deploy"`

Emitted event

```
{
```

```
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Action Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2020-01-31T18:55:43Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "start-time": "2023-10-26T13:51:09.981Z",
        "stage": "Deploy",
        "execution-result": {
            "external-execution-url": "https://us-west-2.console.aws.amazon.com/
codedeploy/home?#/deployments/<ID>",
            "external-execution-summary": "Deployment <ID> failed",
            "external-execution-id": "<ID>",
            "error-code": "JobFailed"
        },
        "action-execution-id": "47f821c5-a902-44b2-ae61-b878d31ecd21",
        "action": "Deploy",
        "state": "FAILED",
        "region": "us-west-2",
        "type": {
            "owner": "AWS",
            "provider": "CodeDeploy",
            "category": "Deploy",
            "version": "1"
        },
        "version": 4.0,
        "pipeline-execution-attempt": 1.0
    }
}
```

Processed event

```
{
    "account": "123456789012",
    "detailType": "CodePipeline Action Execution State Change",
    "region": "us-west-2",
```

```
    "source": "aws.codepipeline",
    "time": "2021-06-24T00:46:16Z",
    "notificationRuleArn": "arn:aws:codestar-notifications:us-
west-2:123456789012:notificationrule/a69c62c21EXAMPLE",
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "stage": "Deploy",
        "execution-result": {
            "external-execution-url": "https://console.aws.amazon.com/codedeploy/
home?region=us-west-2#/deployments/<ID>",
            "external-execution-summary": "Deployment <ID> failed",
            "external-execution-id": "<ID>",
            "error-code": "JobFailed"
        },
        "action-execution-id": "47f821c5-a902-44b2-ae61-b878d31ecd21",
        "action": "Deploy",
        "state": "FAILED",
        "region": "us-west-2",
        "type": {
            "owner": "AWS",
            "provider": "CodeDeploy",
            "category": "Deploy",
            "version": "1"
        },
        "version": 13.0,
        "pipeline-execution-attempt": 1.0
    },
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "additionalAttributes": {
        "additionalInformation": "Deployment <ID> failed"
    }
}
```

Azione: evento ABANDON

Quando l'esecuzione di un'azione viene abbandonata, emette un evento che invia notifiche con il seguente contenuto. Questo esempio riguarda la pipeline denominata `"myPipeline"` nella `us-west-2` regione, per l'azione. `"Deploy"`

```
{
    "version": "0",
    "id": "01234567-EXAMPLE",
    "detail-type": "CodePipeline Action Execution State Change",
    "source": "aws.codepipeline",
    "account": "123456789012",
    "time": "2020-01-31T18:21:39Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:codepipeline:us-west-2:123456789012:myPipeline"
    ],
    "detail": {
        "pipeline": "myPipeline",
        "execution-id": "12345678-1234-5678-abcd-12345678abcd",
        "stage": "Deploy",
        "action-execution-id": "47f821c5-a902-44b2-ae61-b878d31ecd21",
        "action": "Deploy",
        "state": "ABANDONED",
        "region": "us-west-2",
        "type": {
            "owner": "AWS",
            "provider": "CodeDeploy",
            "category": "Deploy",
            "version": "1"
        },
        "version": 3.0,
        "pipeline-execution-attempt": 1.0
    }
}
```

Crea una regola che invia una notifica su un evento Pipeline

Una regola rileva determinati eventi e poi li indirizza verso AWS obiettivi scelti dall'utente. È possibile creare una regola che esegue un' AWS azione automaticamente quando si verifica un'altra AWS azione o una regola che esegue un' AWS azione regolarmente secondo una pianificazione prestabilita.

Argomenti

- Invia una notifica quando lo stato della pipeline cambia (console)
- Invia una notifica quando lo stato della pipeline cambia (CLI)

Invia una notifica quando lo stato della pipeline cambia (console)

Questi passaggi mostrano come utilizzare la EventBridge console per creare una regola per inviare notifiche di modifiche. CodePipeline

Per creare una EventBridge regola che abbia come target la tua pipeline con una fonte Amazon S3

1. Apri la EventBridge console Amazon all'indirizzo https://console.aws.amazon.com/events/.
2. Nel pannello di navigazione, scegli Regole. Lascia selezionato il bus predefinito o scegli un bus per eventi. Scegli Crea regola.
3. In Nome, inserisci un nome per la regola.
4. In Tipo di regola, scegli Regola con un modello di evento. Seleziona Successivo.
5. In Schema di evento, scegli AWS servizi.
6. Dall'elenco a discesa Event Type (Tipo di evento), scegliere il livello di modifica dello stato per la notifica.
   - Per una regola che si applica agli eventi a livello di pipeline, scegliete CodePipelinePipeline Execution State Change.
   - Per una regola che si applica agli eventi a livello di fase, scegliete Stage Execution State Change. CodePipeline
   - Per una regola che si applica agli eventi a livello di azione, scegliete CodePipelineAction Execution State Change.
7. Specificare le modifiche di stato cui si applica la regola:
   - Per una regola valida per tutte le modifiche di stato, scegliere Any state (Qualsiasi stato).
   - Per una regola valida solo per alcune modifiche di stato, scegliere Specific state(s) (Stati specifici), quindi scegliere uno o più valori di stato dall'elenco.
8. Per modelli di eventi più dettagliati di quelli consentiti dai selettori, puoi anche utilizzare l'opzione Modifica modello nella finestra Schema di evento per designare un modello di evento in formato JSON.

> **Note**
>
> Se non diversamente specificato, il modello eventi viene creato per tutte le pipeline/fasi/operazioni e stati.

Per modelli di eventi più dettagliati, è possibile copiare e incollare i seguenti modelli di eventi di esempio nella finestra Event pattern.

- Example

  Utilizzare questo modello eventi di esempio per acquisire le operazioni di distribuzione e di compilazione non riuscite in tutte le pipeline.

```
{
"source": [
    "aws.codepipeline"
  ],
  "detail-type": [
    "CodePipeline Action Execution State Change"
  ],
  "detail": {
    "state": [
      "FAILED"
    ],
    "type": {
      "category": ["Deploy", "Build"]
    }
  }
}
```

- Example

  Utilizzare questo modello eventi di esempio per acquisire tutte le operazioni di approvazione rifiutate o non riuscite in tutte le pipeline.

```
{
 "source": [
    "aws.codepipeline"
  ],
  "detail-type": [
    "CodePipeline Action Execution State Change"
  ],
  "detail": {
    "state": [
      "FAILED"
    ],
```

```
    "type": {
      "category": ["Approval"]
    }
  }
}
```

- Example

  Utilizzare questo modello eventi di esempio per acquisire tutti gli eventi dalle pipeline specificate.

```
{
"source": [
    "aws.codepipeline"
  ],
  "detail-type": [
    "CodePipeline Pipeline Execution State Change",
    "CodePipeline Action Execution State Change",
    "CodePipeline Stage Execution State Change"
  ],
  "detail": {
    "pipeline": ["myPipeline", "my2ndPipeline"]
  }
}
```

9. Seleziona Successivo.
10. In Tipi di Target, scegli AWS service.
11. In Seleziona un obiettivo, scegli CodePipeline. In Pipeline ARN, immettete l'ARN della pipeline per la pipeline da avviare in base a questa regola.

> **Note**
>
> Per ottenere l'ARN della pipeline, esegui il comando get-pipeline. L'ARN della pipeline viene visualizzato nell'output. Il formato è il seguente:
> *arn:aws:codepipeline: regione: account: nome-pipeline*
> ARN della pipeline di esempio:
> arn:aws:codepipeline:us-east- 2:80398 ESEMPIO: MyFirstPipeline

12. Per creare o specificare un ruolo di servizio IAM che conceda le EventBridge autorizzazioni per richiamare il target associato alla regola (in questo caso, l'obiettivo è): EventBridge CodePipeline

- Scegli Crea un nuovo ruolo per questa risorsa specifica per creare un ruolo di servizio che ti dia EventBridge le autorizzazioni per avviare le esecuzioni della pipeline.
- Scegli Usa il ruolo esistente per inserire un ruolo di servizio che ti dia EventBridge le autorizzazioni per avviare le esecuzioni della pipeline.

13. Seleziona Successivo.
14. Nella pagina Tag, scegli Avanti.
15. Nella pagina Rivedi e crea, esamina la configurazione della regola. Se la regola ti soddisfa, scegli Create rule (Crea regola).

Invia una notifica quando lo stato della pipeline cambia (CLI)

Questi passaggi mostrano come utilizzare la CLI per creare una regola CloudWatch Events per inviare notifiche di modifiche. CodePipeline

Per utilizzare la AWS CLI per creare una regola, chiamate il put-rule comando, specificando:

- Un nome che identifica in modo univoco la regola che stai creando. Questo nome deve essere univoco in tutte le pipeline create CodePipeline associate al tuo AWS account.
- Il modello eventi per i campi di origine e di dettaglio utilizzati dalla regola. Per ulteriori informazioni, consulta Amazon EventBridge e Event Patterns.

Per creare una EventBridge regola con CodePipeline come origine dell'evento

1. Chiamare il comando put-rule per creare una regola specificando il modello eventi. (Consulta le tabelle precedenti per gli stati validi.)

   Il seguente comando di esempio utilizza --event-pattern per creare una regola chiamata “`MyPipelineStateChanges`” che emette l' CloudWatch evento quando l'esecuzione di una pipeline fallisce per la pipeline denominata «myPipeline».

   ```
   aws events put-rule --name "MyPipelineStateChanges" --event-pattern "{\"source\":
   [\"aws.codepipeline\"],\"detail-type\":[\"CodePipeline Pipeline Execution State
    Change\"],\"detail\":{\"pipeline\":[\"myPipeline\"],\"state\":[\"FAILED\"]}}"
   ```

2. Richiamate il put-targets comando e includete i seguenti parametri:

   - Il parametro `--rule` viene utilizzato con il `rule_name` che hai creato utilizzando put-rule.

- Il `--targets` parametro viene utilizzato con l'elenco `Id` delle destinazioni nell'elenco delle destinazioni e l'ARNargomento Amazon SNS.

Il comando di esempio seguente specifica che per la regola denominata `MyPipelineStateChanges`, la destinazione `Id` è composta dal numero uno, per indicare che in un elenco di destinazioni per la regola questa è la destinazione 1. Il comando sample specifica anche un esempio ARN per l'argomento Amazon SNS.

```
aws events put-targets --rule MyPipelineStateChanges --targets
 Id=1,Arn=arn:aws:sns:us-west-2:11111EXAMPLE:MyNotificationTopic
```

3. Aggiungi le autorizzazioni per EventBridge utilizzare il servizio di destinazione designato per richiamare la notifica. Per ulteriori informazioni, consulta Utilizzo delle politiche basate sulle risorse per Amazon. EventBridge

Riferimento per il bucket segnaposto eventi

Questa sezione è solo per riferimento. Per informazioni sulla creazione di una pipeline con risorse di rilevamento eventi, consulta Azioni all'origine e metodi di rilevamento delle modifiche.

Crea azioni fornite da Amazon S3 e CodeCommit utilizza risorse di rilevamento delle modifiche basate sugli eventi per attivare la pipeline quando viene apportata una modifica al bucket o al repository di origine. Queste risorse sono le regole CloudWatch Events configurate per rispondere agli eventi nella sorgente della pipeline, come una modifica del codice al repository. CodeCommit Quando usi CloudWatch Events per una fonte Amazon S3, devi attivarla per CloudTrail registrare gli eventi. CloudTrail richiede un bucket S3 a cui inviare i digest. Puoi accedere ai file di registro per le tue risorse CloudWatch Events dal bucket personalizzato, ma non puoi accedere ai dati dal bucket segnaposto.

- Se hai usato la CLI o AWS CloudFormation per configurare le risorse CloudWatch Events, puoi trovare CloudTrail i tuoi file nel bucket che hai specificato quando hai configurato la pipeline.
- Se hai usato la console per configurare la pipeline con una sorgente S3, la console utilizza un bucket CloudTrail segnaposto quando crea le risorse Events per te. CloudWatch CloudTrail i digest vengono archiviati nel bucket segnaposto in cui viene creata la pipeline. Regione AWS

Puoi modificare la configurazione se desideri utilizzare un bucket diverso dal bucket segnaposto.

> **Note**
>
> I dati scritti nei bucket CloudTrail segnaposto scadono automaticamente dopo un giorno e non vengono conservati.

Per ulteriori informazioni sulla ricerca e la gestione dei file di CloudTrail registro, consulta Acquisizione e visualizzazione dei file di registro. CloudTrail

Argomenti

Nomi di bucket segnaposto eventi per regione

Questa tabella elenca i nomi dei bucket segnaposto S3 che contengono file di log che tengono traccia degli eventi di rilevamento delle modifiche per le pipeline con azioni sorgente di Amazon S3.

Nome Regione	Nome del bucket segnaposto	Identificatore della regione
Stati Uniti orientali (Ohio)	codepipeline-cloudtrail-placeholder-bucket-usa-est-2	us-east-2
Stati Uniti orientali (Virginia settentrionale)	codepipeline-cloudtrail-placeholder-bucket-usa-est-1	us-east-1
Stati Uniti occidentali (California settentrionale)	codepipeline-cloudtrail-placeholder-bucket-us-west-1	us-west-1
US West (Oregon)	codepipeline-cloudtrail-placeholder-bucket-us-west-2	us-west-2
Canada (Centrale)	codepipeline-cloudtrail-placeholder-bucket-ca-central-1	ca-central-1
Europa (Francoforte)	codepipeline-cloudtrail-placeholder-bucket-eu-central-1	eu-central-1

Nome Regione	Nome del bucket segnaposto	Identificatore della regione
Europa (Irlanda)	codepipeline-cloudtrail-placeholder-bucket-eu-west-1	eu-west-1
Europa (Londra)	codepipeline-cloudtrail-placeholder-bucket-eu-west-2	eu-west-2
Europa (Parigi)	codepipeline-cloudtrail-placeholder-bucket-eu-west-3	eu-west-3
Europa (Stoccolma)	codepipeline-cloudtrail-placeholder-bucket-eu-nord-1	eu-north-1
Asia Pacifico (Hong Kong)	codepipeline-cloudtrail-placeholder-bucket-ap-est-1	ap-east-1
Asia Pacific (Hyderabad)	codepipeline-cloudtrail-placeholder-bucket-ap-sud-2	ap-south-2
Asia Pacifico (Giacarta)	codepipeline-cloudtrail-placeholder-bucket-ap-sud-est-3	ap-southeast-3
Asia Pacifico (Melbourne)	codepipeline-cloudtrail-placeholder-bucket-ap-sud-est - 4	ap-southeast-4
Asia Pacifico (Mumbai)	codepipeline-cloudtrail-placeholder-bucket-ap-sud-1	ap-south-1
Asia Pacifico (Osaka-Locale)	codepipeline-cloudtrail-placeholder-bucket-ap-northeast-3-prod	ap-northeast-3
Asia Pacifico (Tokyo)	codepipeline-cloudtrail-placeholder-bucket-ap-nord-est-1	ap-northeast-1
Asia Pacifico (Seoul)	codepipeline-cloudtrail-placeholder-bucket-ap-nord-est-2	ap-northeast-2

Nome Regione	Nome del bucket segnaposto	Identificatore della regione
Asia Pacifico (Singapore)	codepipeline-cloudtrail-placeholder-bucket-ap-sud-est-1	ap-southeast-1
Asia Pacifico (Sydney)	codepipeline-cloudtrail-placeholder-bucket-ap-sud-est-2	ap-southeast-2
Asia Pacifico (Tokyo)	codepipeline-cloudtrail-placeholder-bucket-ap-nord-est-1	ap-northeast-1
Canada (Centrale)	codepipeline-cloudtrail-placeholder-bucket-ca-central-1	ca-central-1
Europa (Francoforte)	codepipeline-cloudtrail-placeholder-bucket-eu-central-1	eu-central-1
Europa (Irlanda)	codepipeline-cloudtrail-placeholder-bucket-eu-west-1	eu-west-1
Europa (Londra)	codepipeline-cloudtrail-placeholder-bucket-eu-west-2	eu-west-2
Europa (Milano)	codepipeline-cloudtrail-placeholder-bucket-eu-sud-1	eu-south-1
Europa (Parigi)	codepipeline-cloudtrail-placeholder-bucket-eu-west-3	eu-west-3
Europa (Spagna)	codepipeline-cloudtrail-placeholder-bucket-eu-sud-2	eu-south-2
Europa (Stoccolma)	codepipeline-cloudtrail-placeholder-bucket-eu-nord-1	eu-north-1
Europa (Zurigo) *	codepipeline-cloudtrail-placeholder-bucket-eu-central-2	eu-central-2
Israele (Tel Aviv)	codepipeline-cloudtrail-placeholder-bucket-il-central-1	il-central-1

Nome Regione	Nome del bucket segnaposto	Identificatore della regione
Medio Oriente (Bahrein) *	codepipeline-cloudtrail-placeholder-bucket-me-sud-1	me-south-1
Medio Oriente (Emirati Arabi Uniti)	codepipeline-cloudtrail-placeholder-bucket-me-central-1	me-central-1
Sud America (San Paolo)	codepipeline-cloudtrail-placeholder-bucket-sa-est-1	sa-east-1

Registrazione delle chiamate CodePipeline API con AWS CloudTrail

AWS CodePipeline è integrato con AWS CloudTrail, un servizio che fornisce una registrazione delle azioni intraprese da un utente, ruolo o membro Servizio AWS CodePipeline;. CloudTrail acquisisce tutte le chiamate API CodePipeline come eventi. Le chiamate acquisite includono chiamate dalla CodePipeline console e chiamate di codice alle operazioni CodePipeline API. Se crei un trail, puoi abilitare la distribuzione continua di CloudTrail eventi a un bucket Amazon S3, inclusi gli eventi per. CodePipeline Se non configuri un percorso, puoi comunque visualizzare gli eventi più recenti nella CloudTrail console nella cronologia degli eventi. Utilizzando le informazioni raccolte da CloudTrail, puoi determinare a quale richiesta è stata inviata CodePipeline, l'indirizzo IP da cui è stata effettuata la richiesta, chi ha effettuato la richiesta, quando è stata effettuata e dettagli aggiuntivi.

Per ulteriori informazioni CloudTrail, consulta la Guida AWS CloudTrail per l'utente.

CodePipeline informazioni in CloudTrail

CloudTrail è abilitato sul tuo account al Account AWS momento della creazione dell'account. Quando si verifica un'attività in CodePipeline, tale attività viene registrata in un CloudTrail evento insieme ad altri Servizio AWS eventi nella Cronologia degli eventi. Puoi visualizzare, cercare e scaricare gli eventi recenti nel tuo AWS account. Per ulteriori informazioni, consulta Visualizzazione degli eventi con la cronologia degli CloudTrail eventi.

Per una registrazione continua degli eventi del tuo sito Account AWS , inclusi gli eventi di CodePipeline, crea un percorso. Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Per impostazione predefinita, quando crei un percorso nella console, il percorso si applica a

tutte le AWS regioni. Il trail registra gli eventi di tutte le regioni della AWS partizione e consegna i file di log al bucket Amazon S3 specificato. Inoltre, puoi configurarne altri Servizi AWS per analizzare ulteriormente e agire in base ai dati sugli eventi raccolti nei log. CloudTrail Per ulteriori informazioni, consulta gli argomenti seguenti:

- Panoramica della creazione di un trail
- CloudTrail Servizi e integrazioni supportati
- Configurazione delle notifiche Amazon SNS per CloudTrail
- Ricezione di file di CloudTrail registro da più regioni e ricezione di file di CloudTrail registro da più account

Tutte CodePipeline le azioni vengono registrate CloudTrail e documentate nell'CodePipeline API Reference. Ad esempio, le chiamate a `GetPipelineExecution` e `CreatePipeline` le `UpdatePipeline` azioni generano voci nei file di CloudTrail registro.

Ogni evento o voce di log contiene informazioni sull'utente che ha generato la richiesta. Le informazioni di identità consentono di determinare quanto segue:

- Se la richiesta è stata effettuata con credenziali root o AWS Identity and Access Management (IAM).
- Se la richiesta è stata effettuata con le credenziali di sicurezza temporanee per un ruolo o un utente federato.
- Se la richiesta è stata effettuata da un altro Servizio AWS.

Per ulteriori informazioni, vedete l'elemento CloudTrail userIdentity.

Comprendere le CodePipeline voci dei file di registro

Un trail è una configurazione che consente la distribuzione di eventi come file di log in un bucket Amazon S3 specificato dall'utente. CloudTrail i file di registro contengono una o più voci di registro. Un evento rappresenta una singola richiesta proveniente da qualsiasi fonte e include informazioni sull'azione richiesta, la data e l'ora dell'azione, i parametri della richiesta e così via. CloudTrail i file di registro non sono una traccia ordinata dello stack delle chiamate API pubbliche, quindi non vengono visualizzati in un ordine specifico.

L'esempio seguente mostra una voce di CloudTrail registro per un evento di aggiornamento della pipeline, in cui una pipeline denominata MyFirstPipeline è stata modificata dall'utente denominato,

CodePipeline con l'ID account JaneDoe 80398EXAMPLE. L'utente ha modificato il nome della fase di origine di una pipeline da `Source` in `MySourceStage`. Poiché entrambi `requestParameters` gli `responseElements` elementi del CloudTrail log contengono l'intera struttura della pipeline modificata, tali elementi sono stati abbreviati nell'esempio seguente. Enfasi è stata aggiunta alla parte `requestParameters` della pipeline in cui si è verificata la modifica, al numero di versione precedente della pipeline e alla parte `responseElements`, che mostra il numero di versione incrementato di 1. Le parti modificate sono contrassegnate con puntini di sospensione (...) per illustrare dove più dati vengono visualizzati in una voce di log reale.

```
{
 "eventVersion":"1.03",
  "userIdentity": {
   "type":"IAMUser",
   "principalId":"AKIAI44QH8DHBEXAMPLE",
   "arn":"arn:aws:iam::80398EXAMPLE:user/JaneDoe-CodePipeline",
   "accountId":"80398EXAMPLE",
   "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
   "userName":"JaneDoe-CodePipeline",
   "sessionContext": {
   "attributes":{
    "mfaAuthenticated":"false",
    "creationDate":"2015-06-17T14:44:03Z"
    }
   },
 "invokedBy":"signin.amazonaws.com"},
 "eventTime":"2015-06-17T19:12:20Z",
 "eventSource":"codepipeline.amazonaws.com",
 "eventName":"UpdatePipeline",
 "awsRegion":"us-east-2",
 "sourceIPAddress":"192.0.2.64",
 "userAgent":"signin.amazonaws.com",
 "requestParameters":{
   "pipeline":{
   "version":1,
   "roleArn":"arn:aws:iam::80398EXAMPLE:role/CodePipeline_Service_Role",
   "name":"MyFirstPipeline",
   "stages":[
     {
     "actions":[
      {
          "name":"MySourceStage",
        "actionType":{
```

```
        "owner":"AWS",
        "version":"1",
        "category":"Source",
        "provider":"S3"
     },
    "inputArtifacts":[],
    "outputArtifacts":[
     {"name":"MyApp"}
     ],
    "runOrder":1,
    "configuration":{
     "S3Bucket":"awscodepipeline-demobucket-example-date",
     "S3ObjectKey":"sampleapp_linux.zip"
      }
      }
     ],
       "name":"Source"
     },
     (...)
                },
 "responseElements":{
   "pipeline":{
     "version":2,
     (...)
           },
     "requestID":"2c4af5c9-7ce8-EXAMPLE",
     "eventID":""c53dbd42-This-Is-An-Example"",
     "eventType":"AwsApiCall",
     "recipientAccountId":"80398EXAMPLE"
     }
      ]
}
```

Risoluzione dei problemi CodePipeline

Le informazioni seguenti possono risultare utili per risolvere i problemi comuni di AWS CodePipeline.

Argomenti

- Errore di connessione durante la connessione a GitHub: «Si è verificato un problema, assicurati che i cookie siano abilitati nel tuo browser» o «Il proprietario di un'organizzazione deve installare l' GitHub app»
- Le pipeline con modalità di esecuzione modificata in modalità QUEUED o PARALLEL non funzionano quando viene raggiunto il limite di esecuzione
- Le tubazioni in modalità PARALLEL hanno una definizione di pipeline obsoleta se modificate quando si passa alla modalità QUEUED o SUPERSEDED
- Le pipeline modificate dalla modalità PARALLEL mostreranno una modalità di esecuzione precedente
- Le pipeline con connessioni che utilizzano il filtraggio dei trigger in base ai percorsi dei file potrebbero non iniziare alla creazione del ramo
- Le pipeline con connessioni che utilizzano il filtro a trigger in base ai percorsi dei file potrebbero non avviarsi quando viene raggiunto il limite di file
- CodeCommit oppure le revisioni dei sorgenti S3 in modalità PARALLEL potrebbero non corrispondere all'evento EventBridge
- Hai bisogno di assistenza per un problema diverso?

Errore della pipeline: una pipeline configurata con AWS Elastic Beanstalk restituisce il messaggio di errore: "Distribuzione non riuscita. Il ruolo fornito non dispone di autorizzazioni sufficienti: Servizio:» AmazonElasticLoadBalancing

Problema: il ruolo di servizio per CodePipeline non dispone di autorizzazioni sufficienti per AWS Elastic Beanstalk, incluse, a titolo esemplificativo, alcune operazioni in Elastic Load Balancing. Il ruolo di servizio per CodePipeline è stato aggiornato il 6 agosto 2015 per risolvere questo problema. I clienti che hanno creato il ruolo di servizio prima di questa data devono modificare l'istruzione della policy per il ruolo del servizio in modo da aggiungere le autorizzazioni necessarie.

Possibili correzioni: la soluzione più semplice consiste nel modificare l'istruzione della policy per il ruolo del servizio, come descritto ampiamente in Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline.

Dopo aver applicato la policy modificata, segui i passaggi indicati per Avvio manuale di una pipeline rieseguire manualmente tutte le pipeline che utilizzano Elastic Beanstalk.

In base alle esigenze di sicurezza, puoi modificare le autorizzazioni anche in altri modi.

Errore di distribuzione: una pipeline configurata con un'azione di AWS Elastic Beanstalk distribuzione si blocca invece di fallire se manca l'autorizzazione "" DescribeEvents

Problema: il ruolo di servizio per CodePipeline deve includere l'"`elasticbeanstalk:DescribeEvents`"azione per tutte le pipeline che utilizzano. AWS Elastic Beanstalk Senza questa autorizzazione, le azioni di AWS Elastic Beanstalk distribuzione si bloccano senza fallire o indicare un errore. Se questa azione non rientra nel tuo ruolo di servizio, CodePipeline significa che non hai le autorizzazioni per eseguire la fase di distribuzione della pipeline per tuo conto. AWS Elastic Beanstalk

Possibili correzioni: rivedi il tuo CodePipeline ruolo di servizio. Se l'operazione `"elasticbeanstalk:DescribeEvents"` è mancante, utilizza le istruzioni in Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline per aggiungerla usando la funzione Edit Policy (Modifica policy) nella console IAM.

Dopo aver applicato la policy modificata, segui i passaggi indicati per Avvio manuale di una pipeline rieseguire manualmente tutte le pipeline che utilizzano Elastic Beanstalk.

Errore di pipeline: un'azione di origine restituisce il messaggio di autorizzazioni insufficienti: «Impossibile accedere al repository. CodeCommit **`repository-name`** Verifica che il ruolo IAM della pipeline abbia le autorizzazioni sufficienti per accedere al repository."

Problema: il ruolo di servizio per CodePipeline non dispone di autorizzazioni sufficienti CodeCommit e probabilmente è stato creato prima dell'aggiunta del supporto per l'utilizzo dei CodeCommit repository il 18 aprile 2016. I clienti che hanno creato il ruolo di servizio prima di questa data devono modificare l'istruzione della policy per il ruolo del servizio in modo da aggiungere le autorizzazioni necessarie.

Possibili correzioni: aggiungi le autorizzazioni richieste per la politica del tuo CodeCommit ruolo di CodePipeline servizio. Per ulteriori informazioni, consulta Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline.

Errore della pipeline: un'operazione di test o compilazione Jenkins viene eseguita per lungo tempo e poi restituisce l'esito negativo a causa di mancanza di credenziali o autorizzazioni

Problema: se il server Jenkins è installato su un'istanza Amazon EC2, è possibile che l'istanza non sia stata creata con un ruolo di istanza con le autorizzazioni richieste per. CodePipeline Se utilizzi un utente IAM su un server Jenkins, un'istanza locale o un'istanza Amazon EC2 creata senza il ruolo IAM richiesto, l'utente IAM non dispone delle autorizzazioni richieste oppure il server Jenkins non può accedere a tali credenziali tramite il profilo configurato sul server.

Possibili correzioni: assicurati che il ruolo dell'istanza Amazon EC2 o l'utente IAM siano configurati con `AWSCodePipelineCustomActionAccess` la policy gestita o con le autorizzazioni equivalenti. Per ulteriori informazioni, consulta AWS politiche gestite per AWS CodePipeline.

Se utilizzi un utente IAM, assicurati che il AWS profilo configurato sull'istanza utilizzi l'utente IAM configurato con le autorizzazioni corrette. Potrebbe essere necessario fornire le credenziali utente IAM che hai configurato per l'integrazione tra Jenkins e CodePipeline direttamente nell'interfaccia utente di Jenkins. Questa non è una best practice consigliata. Se è necessario, verifica che il server Jenkins sia protetto e utilizzi HTTPS anziché HTTP.

Errore di pipeline: una pipeline creata in una AWS regione utilizzando un bucket creato in un'altra AWS regione restituisce un "" con il codice "» InternalError JobFailed

Problema: il download di un elemento archiviato in un bucket Amazon S3 non riesce se la pipeline e il bucket vengono creati in regioni diverse. AWS

Possibili correzioni: assicurati che il bucket Amazon S3 in cui è archiviato l'artefatto si trovi nella AWS stessa regione della pipeline che hai creato.

Errore di distribuzione: un file ZIP che contiene un file WAR viene distribuito correttamente AWS Elastic Beanstalk, ma l'URL dell'applicazione riporta un errore 404 not found

Problema: un file WAR viene distribuito in un ambiente AWS Elastic Beanstalk , ma l'URL dell'applicazione restituisce un errore 404 Non trovato.

Possibili correzioni: AWS Elastic Beanstalk può decomprimere un file ZIP, ma non un file WAR contenuto in un file ZIP. Invece di specificare un file WAR nel tuo file `buildspec.yml`, specifica una cartella con i contenuti da distribuire. Per esempio:

```
version: 0.2

phases:
  post_build:
    commands:
      - mvn package
      - mv target/my-web-app ./
artifacts:
  files:
    - my-web-app/**/*
 discard-paths: yes
```

Per un esempio, consulta Esempio di AWS Elastic Beanstalk per CodeBuild.

I nomi della cartella degli artefatti della pipeline sembrano troncati

Problema: quando si visualizzano i nomi degli artefatti della pipeline in CodePipeline, i nomi sembrano troncati. Questo può far sì che i nomi sembrino simili o sembrino non contenere più tutto il nome della pipeline.

Spiegazione: CodePipeline tronca i nomi degli artefatti per garantire che il percorso completo di Amazon S3 non superi i limiti di dimensione delle policy quando CodePipeline genera credenziali temporanee per i lavoratori.

Anche se il nome dell'artefatto sembra troncato, viene mappato al bucket degli artefatti in modo da non CodePipeline risentire degli artefatti con nomi troncati. La pipeline può funzionare normalmente. Questo non è un problema con la cartella o gli artefatti. I nomi di pipeline hanno un limite di 100

caratteri. Anche se il nome della cartella degli artefatti potrebbe sembrare accorciato, è ancora univoco per la pipeline.

Aggiungi le autorizzazioni per le connessioni a Bitbucket, Enterprise Server o.com CodeBuild GitClone GitHub GitHub GitLab

Quando si utilizza una AWS CodeStar connessione in un'azione di origine e in un' CodeBuild azione, ci sono due modi in cui l'elemento di input può essere passato alla build:

- Impostazione predefinita: l'azione source produce un file zip che contiene il codice che CodeBuild viene scaricato.
- Git clone: il codice sorgente può essere scaricato direttamente nell'ambiente di compilazione.

  La modalità Git clone permette di interagire con il codice sorgente come repository Git funzionante. Per utilizzare questa modalità, è necessario concedere CodeBuild all'ambiente le autorizzazioni necessarie per utilizzare la connessione.

Per aggiungere autorizzazioni alla politica relativa al ruolo CodeBuild di servizio, è necessario creare una politica gestita dal cliente da allegare al proprio ruolo di servizio. CodeBuild La procedura seguente crea una policy in cui l'autorizzazione `UseConnection` è specificata nel campo `action` e l'ARN della connessione è specificato nel campo `Resource`.

Per utilizzare la console per aggiungere le autorizzazioni UseConnection

1. Per trovare l'ARN della connessione per la pipeline, aprire la pipeline e fare clic sull'icona (i) nell'operazione di origine. L'ARN della connessione viene aggiunto alla politica del ruolo CodeBuild di servizio.

   Un esempio di ARN di connessione è:

   ```
   arn:aws:codeconnections:eu-central-1:123456789123:connection/
   sample-1908-4932-9ecc-2ddacee15095
   ```

2. Per trovare il tuo ruolo di CodeBuild servizio, apri il progetto di build utilizzato nella tua pipeline e vai alla scheda Dettagli della build.
3. Scegliere il collegamento Service role (Ruolo del servizio). Si apre la console IAM, in cui è possibile aggiungere una nuova policy che permette l'accesso alla connessione.

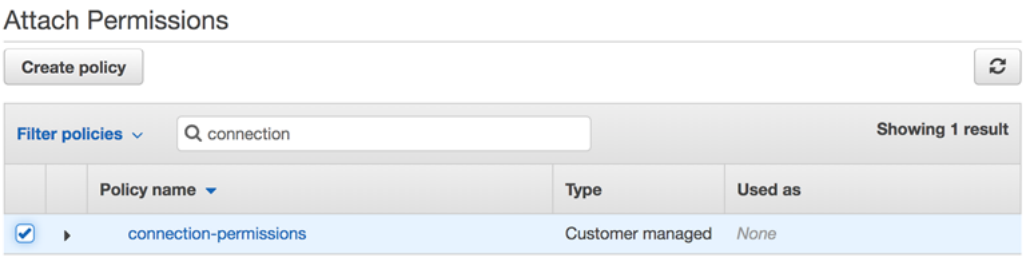
4. Nella console IAM, scegliere Attach policies (Collega policy), quindi selezionare Create policy (Crea policy).

   Utilizzare il seguente esempio di modello di policy. Aggiungere l'ARN della connessione nel campo `Resource`, come mostrato in questo esempio:

   ```
   {
       "Version": "2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": "codestar-connections:UseConnection",
               "Resource": "insert connection ARN here"
           }
       ]
   }
   ```

   Nella scheda JSON incollare la policy.

5. Scegli Verifica policy. Immettere un nome per la policy (ad esempio **`connection-permissions`**), quindi scegliere Create policy (Crea policy).
6. Tornare alla pagina in cui si stavano collegando le autorizzazioni, aggiornare l'elenco delle policy e selezionare la policy appena creata. Scegli Collega policy.



Aggiungi le CodeBuild GitClone autorizzazioni per le azioni di origine CodeCommit

Quando la pipeline ha un'azione CodeCommit sorgente, ci sono due modi per passare l'artefatto di input alla build:

- Predefinito: l'azione source produce un file zip che contiene il codice che viene scaricato. CodeBuild

- Clone completo: il codice sorgente può essere scaricato direttamente nell'ambiente di compilazione.

  L'opzione Full clone consente di interagire con il codice sorgente come un repository Git funzionante. Per utilizzare questa modalità, è necessario aggiungere le autorizzazioni per consentire all' CodeBuildambiente di estrarre dal repository.

Per aggiungere autorizzazioni alla politica relativa al ruolo CodeBuild di servizio, è necessario creare una politica gestita dal cliente da allegare al proprio ruolo di servizio. CodeBuild I passaggi seguenti creano una politica che specifica l'`codecommit:GitPull`autorizzazione nel campo. `action`

Per utilizzare la console per aggiungere le GitPull autorizzazioni

1. Per trovare il tuo ruolo di CodeBuild servizio, apri il progetto di build utilizzato nella tua pipeline e vai alla scheda Dettagli della build.
2. Scegliere il collegamento Service role (Ruolo del servizio). Si apre la console IAM in cui puoi aggiungere una nuova policy che concede l'accesso al tuo repository.
3. Nella console IAM, scegliere Attach policies (Collega policy), quindi selezionare Create policy (Crea policy).
4. Nella scheda JSON, incolla la seguente policy di esempio.

   ```
   {
       "Action": [
           "codecommit:GitPull"
       ],
       "Resource": "*",
       "Effect": "Allow"
   },
   ```

5. Scegli Verifica policy. Immettere un nome per la policy (ad esempio **`codecommit-gitpull`**), quindi scegliere Create policy (Crea policy).
6. Tornare alla pagina in cui si stavano collegando le autorizzazioni, aggiornare l'elenco delle policy e selezionare la policy appena creata. Scegli Collega policy.

<source artifact name>Errore di pipeline: una distribuzione con l'azione CodeDeployTo ECS restituisce un messaggio di errore: «Eccezione durante il tentativo di leggere il file degli artefatti di definizione dell'attività da:»

Problema

Il file di definizione dell'attività è un elemento necessario per l'azione di CodePipeline distribuzione su Amazon ECS tramite CodeDeploy (l'azione). `CodeDeployToECS` La dimensione massima del file ZIP dell'artefatto nell'azione di distribuzione è di 3 MB`CodeDeployToECS`. Il seguente messaggio di errore viene restituito quando il file non viene trovato o la dimensione dell'artefatto supera i 3 MB:

Eccezione durante il tentativo di leggere il file artefatto della definizione dell'attività da: <source artifact name>

Possibili correzioni: assicuratevi che il file di definizione dell'attività sia incluso come artefatto. Se il file esiste già, assicurati che la dimensione compressa sia inferiore a 3 MB.

GitHub azione sorgente della versione 1: l'elenco dei repository mostra diversi repository

Problema

Dopo aver autorizzato con successo un'azione della GitHub versione 1 nella CodePipeline console, puoi scegliere da un elenco dei tuoi GitHub repository. Se l'elenco non include i repository che ti aspettavi di vedere, puoi risolvere i problemi relativi all'account utilizzato per l'autorizzazione.

Possibili correzioni: l'elenco dei repository fornito nella CodePipeline console si basa sull' GitHub organizzazione a cui appartiene l'account autorizzato. Verifica che l'account con cui stai utilizzando per l'autorizzazione GitHub sia l'account associato all' GitHub organizzazione in cui è stato creato il repository.

GitHub azione di origine della versione 2: impossibile completare la connessione per un repository

Problema

Poiché una connessione a un GitHub repository utilizza il AWS Connector for GitHub, per creare la connessione sono necessarie le autorizzazioni del proprietario dell'organizzazione o delle autorizzazioni di amministratore per accedere al repository.

Possibili correzioni: per informazioni sui livelli di autorizzazione per un GitHub repository, consulta https://docs.github.com/en/ free-pro-team @latest /github/ -/-organization. setting-up-and-managing organizations-and-teams permission-levels-for-an

Errore Amazon S3: al ruolo di CodePipeline servizio <ARN>viene negato l'accesso a S3 per il bucket S3 < > BucketName

Problema

Mentre è in corso, l' CodeCommit azione in CodePipeline verifica l'esistenza del bucket di artefatti della pipeline. Se l'azione non è autorizzata a verificare, si verifica un `AccessDenied` errore in Amazon S3 e il seguente messaggio di errore viene visualizzato in: CodePipeline

CodePipeline *`il ruolo di servizio «arn:aws:iam:: accountID:role/service-role/ roleID" sta negando l'accesso a S3 per il bucket S3 "» BucketName`*

CloudTrail I log dell'azione registrano anche l'errore. `AccessDenied`

Correzioni possibili: procedi come segue:

- Per la politica associata al tuo ruolo CodePipeline di servizio, `s3:ListBucket` aggiungila all'elenco delle azioni della tua politica. Per istruzioni su come visualizzare la politica relativa ai ruoli di servizio, consultaVisualizza l'ARN della pipeline e l'ARN del ruolo di servizio (console). Modifica la dichiarazione politica per il tuo ruolo di servizio come descritto inAggiunta delle autorizzazioni dal ruolo di servizio CodePipeline.
- Per la policy basata sulle risorse allegata al bucket di artifatti Amazon S3 per la tua pipeline, chiamata anche artifact bucket policy, aggiungi un'istruzione per consentire l'utilizzo dell'autorizzazione da parte del tuo ruolo di servizio. `s3:ListBucket` CodePipeline

  Per aggiungere la tua policy al bucket di artefatti

  1. Segui i passaggi Visualizza l'ARN della pipeline e l'ARN del ruolo di servizio (console) per scegliere il tuo bucket di artefatti nella pagina Impostazioni della pipeline, quindi visualizzalo nella console Amazon S3.
  2. Seleziona Autorizzazioni.

3. In Bucket Policy (Policy del bucket) scegliere Edit (Modifica).
4. Nel campo di testo Policy, inserisci una nuova bucket policy o modifica la policy esistente come mostrato nell'esempio seguente. La bucket policy è un file JSON, quindi devi inserire un codice JSON valido.

   L'esempio seguente mostra un'istruzione bucket policy per un bucket di artefatti in cui l'ID del ruolo di esempio per il ruolo di servizio è AROAEXAMPLEID.

```
{
    "Effect": "Allow",
    "Principal": "*",
    "Action": "s3:ListBucket",
    "Resource": "arn:aws:s3:::BucketName",
    "Condition": {
        "StringLike": {
            "aws:userid": "AROAEXAMPLEID:*"
        }
    }
}
```

   L'esempio seguente mostra la stessa dichiarazione di policy del bucket dopo l'aggiunta dell'autorizzazione.

```
{
    "Version": "2012-10-17",
    "Id": "SSEAndSSLPolicy",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890",
            "Condition": {
                "StringLike": {
                    "aws:userid": "AROAEXAMPLEID:*"
                }
            }
        },
        {
            "Sid": "DenyUnEncryptedObjectUploads",
```

```
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "aws:kms"
                }
            }
        },
        {
            "Sid": "DenyInsecureConnections",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": false
                }
            }
        }
    ]
}
```

Per ulteriori informazioni, consulta le fasi in https://aws.amazon.com/blogs/security/writing-iam-policies-how-to-grant-access-to-an-amazon-s3-bucket/.

5. Selezionare Salva.

Dopo aver applicato la policy modificata, segui i passaggi indicati per Avvio manuale di una pipeline rieseguire manualmente la pipeline.

Le pipeline con Amazon S3, Amazon ECR CodeCommit o una fonte non si avviano più automaticamente

Problema

Dopo aver apportato una modifica alle impostazioni di configurazione per un'azione che utilizza le regole CloudWatch degli eventi (EventBridgeo Events) per il rilevamento delle modifiche, la console potrebbe non rilevare una modifica laddove gli identificatori del trigger di origine sono simili e hanno

caratteri iniziali identici. Poiché la nuova regola di evento non viene creata dalla console, la pipeline non si avvia più automaticamente.

Un esempio di modifica minore alla fine del nome del parametro per CodeCommit potrebbe essere la modifica del nome del CodeCommit ramo `MyTestBranch-1` in`MyTestBranch-2`. Poiché la modifica si trova alla fine del nome del ramo, la regola di evento per l'azione di origine potrebbe non aggiornare o creare una regola per le nuove impostazioni di origine.

Ciò si applica alle azioni di origine che utilizzano gli eventi CWE per il rilevamento delle modifiche come segue:

Azione all'origine	Parametri/ identificatori di attivazione (console)
Amazon ECR	Nome del repository Tag di immagine
Amazon S3	Bucket Chiave oggetto S3
CodeCommit	Nome del repository Nome del ramo

Possibili soluzioni.

Esegui una di queste operazioni:

- Modificate le impostazioni di configurazione CodeCommit /S3/ECR in modo da apportare modifiche alla parte iniziale del valore del parametro.

  Esempio: modifica il nome della filiale in. `release-branch` `2nd-release-branch` Evita di cambiare alla fine del nome, ad esempio`release-branch-2`.
- Modificate le impostazioni di configurazione CodeCommit /S3/ECR per ogni pipeline.

  Esempio: cambia il nome della filiale in. `myRepo/myBranch` `myDeployRepo/myDeployBranch` Evita di cambiare alla fine del nome, ad esempio`myRepo/myBranch2`.
- Invece della console, utilizza la CLI o AWS CloudFormation per creare e aggiornare le regole degli eventi di rilevamento delle modifiche. Per istruzioni sulla creazione di regole di evento

per un'azione sorgente S3, consulta. Azioni di origine di Amazon S3 e con EventBridge AWS CloudTrail Per istruzioni sulla creazione di regole di evento per un'azione Amazon ECR, consulta Azioni e risorse relative ai sorgenti di Amazon ECR EventBridge . Per istruzioni sulla creazione di regole di evento per un' CodeCommit azione, consulta CodeCommit azioni di origine e EventBridge.

Dopo aver modificato la configurazione dell'azione nella console, accetta le risorse aggiornate per il rilevamento delle modifiche create dalla console.

Errore di connessione durante la connessione a GitHub: «Si è verificato un problema, assicurati che i cookie siano abilitati nel tuo browser» o «Il proprietario di un'organizzazione deve installare l' GitHub app»

Problema

Per creare la connessione per un'azione di GitHub origine in CodePipeline, devi essere il proprietario GitHub dell'organizzazione. Per i repository che non appartengono a un'organizzazione, è necessario esserne il proprietario. Quando una connessione viene creata da qualcuno diverso dal proprietario dell'organizzazione, viene creata una richiesta per il proprietario dell'organizzazione e viene visualizzato uno dei seguenti errori:

A problem occurred, make sure cookies are enabled in your browser (Si è verificato un problema, assicurarsi che i cookie siano abilitati nel browser)

O

Il proprietario dell'organizzazione deve installare l' GitHub app

Possibili correzioni: per i repository di un' GitHuborganizzazione, il proprietario dell'organizzazione deve creare la connessione al GitHub repository. Per i repository che non appartengono a un'organizzazione, è necessario esserne il proprietario.

Le pipeline con modalità di esecuzione modificata in modalità QUEUED o PARALLEL non funzionano quando viene raggiunto il limite di esecuzione

Problema: il numero massimo di esecuzioni simultanee per una pipeline in modalità QUEUED è di 50 esecuzioni. Quando viene raggiunto questo limite, la pipeline fallisce senza un messaggio di stato.

Possibili correzioni: quando modificate la definizione della pipeline per la modalità di esecuzione, effettuate la modifica separatamente dalle altre azioni di modifica.

Per ulteriori informazioni sulla modalità di esecuzione QUEUED o PARALLEL, vedere. CodePipeline concetti

Le tubazioni in modalità PARALLEL hanno una definizione di pipeline obsoleta se modificate quando si passa alla modalità QUEUED o SUPERSEDED

Problema: per le pipeline in modalità parallela, quando si modifica la modalità di esecuzione della pipeline su QUEUED o SUPERSEDED, la definizione della pipeline per la modalità PARALLEL non verrà aggiornata. La definizione della pipeline aggiornata durante l'aggiornamento della modalità PARALLEL non viene utilizzata nella modalità SUPERSEDED o QUEUED

Possibili correzioni: per le pipeline in modalità parallela, quando modificate la modalità di esecuzione della pipeline su QUEUED o SUPERSEDED, evitate di aggiornare contemporaneamente la definizione della pipeline.

Per ulteriori informazioni sulla modalità di esecuzione QUEUED o PARALLEL, vedere. CodePipeline concetti

Le pipeline modificate dalla modalità PARALLEL mostreranno una modalità di esecuzione precedente

Problema: per le pipeline in modalità PARALLEL, quando si modifica la modalità di esecuzione della pipeline su QUEUED o SUPERSEDED, lo stato della pipeline non visualizzerà lo stato aggiornato come PARALLEL. Se la pipeline è cambiata da PARALLEL a QUEUED o SUPERSEDED, lo stato

della pipeline in modalità SUPERSEDED o QUEUED sarà l'ultimo stato noto in una di queste modalità. Se la pipeline non è mai stata eseguita in quella modalità prima, lo stato sarà vuoto.

Possibili correzioni: per le pipeline in modalità parallela, quando si modifica la modalità di esecuzione della pipeline su QUEUED o SUPERSEDED, si noti che la visualizzazione della modalità di esecuzione non mostrerà lo stato PARALLEL.

Per ulteriori informazioni sulla modalità di esecuzione QUEUED o PARALLEL, vedere. CodePipeline concetti

Le pipeline con connessioni che utilizzano il filtraggio dei trigger in base ai percorsi dei file potrebbero non iniziare alla creazione del ramo

Descrizione: per le pipeline con azioni di origine che utilizzano connessioni, come un'azione di BitBucket origine, puoi impostare un trigger con una configurazione Git che ti consenta di filtrare in base ai percorsi dei file per avviare la pipeline. In alcuni casi, per le pipeline con trigger filtrati in base ai percorsi dei file, la pipeline potrebbe non avviarsi quando viene creato per la prima volta un ramo con un filtro per il percorso dei file, poiché ciò non consente la CodeConnections connessione per risolvere i file modificati. Quando la configurazione Git per il trigger è impostata per filtrare i percorsi dei file, la pipeline non si avvia quando il ramo con il filtro è appena stato creato nel repository di origine. Per ulteriori informazioni sul filtraggio dei percorsi dei file, vedere. Filtra i trigger nelle richieste push o pull di codice

Risultato: ad esempio, le pipeline CodePipeline che hanno un filtro per il percorso dei file su un ramo «B» non verranno attivate quando viene creato il ramo «B». Se non sono presenti filtri per il percorso dei file, la pipeline verrà comunque avviata.

Le pipeline con connessioni che utilizzano il filtro a trigger in base ai percorsi dei file potrebbero non avviarsi quando viene raggiunto il limite di file

Descrizione: per le pipeline con azioni di origine che utilizzano connessioni, come un'azione di BitBucket origine, puoi impostare un trigger con una configurazione Git che ti consenta di filtrare in base ai percorsi dei file per avviare la pipeline. CodePipeline recupera fino ai primi 100 file; pertanto, quando la configurazione Git per il trigger è impostata per filtrare i percorsi dei file, la pipeline

potrebbe non avviarsi se ci sono più di 100 file. Per ulteriori informazioni sul filtraggio dei percorsi dei file, vedere. Filtra i trigger nelle richieste push o pull di codice

Risultato: ad esempio, se un file diff contiene 150 file, CodePipeline esamina i primi 100 file (senza un ordine particolare) per confrontarli con il filtro del percorso del file specificato. Se il file che corrisponde al filtro del percorso dei file non è tra i 100 file recuperati da CodePipeline, la pipeline non verrà richiamata.

CodeCommit oppure le revisioni dei sorgenti S3 in modalità PARALLEL potrebbero non corrispondere all'evento EventBridge

Descrizione: per le esecuzioni della pipeline in modalità PARALLEL, un'esecuzione potrebbe iniziare con la modifica più recente, ad esempio il commit del CodeCommit repository, che potrebbe non essere la stessa della modifica dell'evento. EventBridge In alcuni casi, in cui potrebbe trascorrere una frazione di secondo tra i commit o i tag di immagine che avviano la pipeline, quando CodePipeline riceve l'evento e avvia l'esecuzione, è stato inserito un altro commit o tag di immagine CodePipeline (ad esempio, l' CodeCommit azione) clonerà il commit HEAD in quel momento.

Risultato: per le pipeline in modalità PARALLEL con una sorgente CodeCommit o S3, indipendentemente dalla modifica che ha innescato l'esecuzione della pipeline, l'azione di origine clonerà sempre l'HEAD nel momento in cui viene avviato. Ad esempio, per una pipeline in modalità PARALLEL, viene inviato un commit, che avvia la pipeline per l'esecuzione 1, e la seconda esecuzione della pipeline utilizza il secondo commit.

Hai bisogno di assistenza per un problema diverso?

Prova a queste altre risorse:

- Contatta il supporto AWS.
- Fai una domanda nel forum. CodePipeline
- Richiedi un aumento delle quote. Per ulteriori informazioni, consulta Quote in AWS CodePipeline.

> **Note**
>
> L'elaborazione delle richieste di aumento delle quote può richiedere fino a due settimane.

Sicurezza in AWS CodePipeline

La sicurezza del cloud AWS è la massima priorità. In qualità di AWS cliente, puoi beneficiare di un data center e di un'architettura di rete progettati per soddisfare i requisiti delle organizzazioni più sensibili alla sicurezza.

La sicurezza è una responsabilità condivisa tra te e te. AWS Il modello di responsabilità condivisa descrive questo come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gira Servizi AWS su Cloud AWS. AWS fornisce inoltre servizi che è possibile utilizzare in modo sicuro. I revisori di terze parti testano e verificano regolarmente l'efficacia della sicurezza come parte dei programmi di conformitàAWS. Per maggiori informazioni sui programmi di conformità applicabili AWS CodePipeline, consulta la sezione Servizio AWS Ambito per programma di conformità.
- Sicurezza nel cloud: la tua responsabilità è determinata dal Servizio AWS materiale che utilizzi. L'utente è anche responsabile di altri fattori, tra cui la riservatezza dei dati, i requisiti dell'azienda e le leggi e le normative applicabili.

Questa documentazione aiuta a capire come applicare il modello di responsabilità condivisa durante l'utilizzo CodePipeline. I seguenti argomenti mostrano come eseguire la configurazione CodePipeline per soddisfare gli obiettivi di sicurezza e conformità. Imparerai anche a utilizzarne altri Servizi AWS che ti aiutano a monitorare e proteggere CodePipeline le tue risorse.

Argomenti

Protezione dei dati in AWS CodePipeline

Il modello di responsabilità AWS condivisa modello di di si applica alla protezione dei dati in AWS CodePipeline. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i Cloud AWS. L'utente è responsabile del controllo dei contenuti ospitati su questa infrastruttura. L'utente è inoltre responsabile della configurazione della protezione e delle attività di gestione per i Servizi AWS utilizzati. Per ulteriori informazioni sulla privacy dei dati, vedi le Domande frequenti sulla privacy dei dati. Per informazioni sulla protezione dei dati in Europa, consulta il post del blog relativo al Modello di responsabilità condivisa AWS e GDPR nel Blog sulla sicurezza AWS .

Ai fini della protezione dei dati, consigliamo di proteggere Account AWS le credenziali e configurare i singoli utenti con AWS IAM Identity Center or AWS Identity and Access Management (IAM). In tal modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere i suoi compiti. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- Usa SSL/TLS per comunicare con le risorse. AWS È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Configura l'API e la registrazione delle attività degli utenti con. AWS CloudTrail
- Utilizza soluzioni di AWS crittografia, insieme a tutti i controlli di sicurezza predefiniti all'interno Servizi AWS.
- Utilizza i servizi di sicurezza gestiti avanzati, come Amazon Macie, che aiutano a individuare e proteggere i dati sensibili archiviati in Amazon S3.
- Se hai bisogno di moduli crittografici convalidati FIPS 140-2 per l'accesso AWS tramite un'interfaccia a riga di comando o un'API, utilizza un endpoint FIPS. Per ulteriori informazioni sugli endpoint FIPS disponibili, consulta il Federal Information Processing Standard (FIPS) 140-2.

Ti consigliamo vivamente di non inserire mai informazioni riservate o sensibili, ad esempio gli indirizzi e-mail dei clienti, nei tag o nei campi di testo in formato libero, ad esempio nel campo Nome. Ciò include quando lavori o Servizi AWS utilizzi la console, l'API CodePipeline o gli SDK. AWS CLI AWS I dati inseriti nei tag o nei campi di testo in formato libero utilizzati per i nomi possono essere utilizzati per i la fatturazione o i log di diagnostica. Quando fornisci un URL a un server esterno, ti suggeriamo vivamente di non includere informazioni sulle credenziali nell'URL per convalidare la tua richiesta al server.

Le seguenti best practice di sicurezza riguardano anche la protezione dei dati in CodePipeline:

- Configura la crittografia lato server per gli artefatti archiviati in Amazon S3 per CodePipeline
- Utilizzalo per tenere traccia delle password del database o delle chiavi AWS Secrets Manager API di terze parti

Riservatezza del traffico Internet

Amazon VPC è un software Servizio AWS che puoi utilizzare per avviare AWS risorse in una rete virtuale (cloud privato virtuale) da te definita. CodePipelinesupporta gli endpoint Amazon VPC powered by AWS PrivateLink, una AWS tecnologia che facilita la comunicazione privata tra l' Servizi AWS utilizzo di un'interfaccia di rete elastica e indirizzi IP privati. Ciò significa che puoi connetterti direttamente CodePipeline tramite un endpoint privato nel tuo VPC, mantenendo tutto il traffico all'interno del tuo VPC e della rete. AWS In precedenza, le applicazioni eseguite all'interno di un VPC richiedevano l'accesso a Internet per connettersi. CodePipeline Con un VPC, hai il controllo delle impostazioni di rete, ad esempio:

- Intervallo di indirizzi IP,
- sottoreti,
- Tabelle di routing e
- Gateway di rete.

Per connettere il tuo VPC CodePipeline, definisci un'interfaccia VPC endpoint per. CodePipeline Questo tipo di endpoint ti consente di connettere il tuo Servizi AWS VPC a. L'endpoint fornisce una connettività affidabile e scalabile CodePipeline senza richiedere un gateway Internet, un'istanza NAT (Network Address Translation) o una connessione VPN. Per ulteriori informazioni sulla configurazione di un VPC, consulta la Guida per l'utente di VPC.

Crittografia a riposo

I dati in ingresso vengono crittografati CodePipeline quando sono inattivi utilizzando. AWS KMS keys Gli artefatti del codice vengono archiviati in un bucket S3 di proprietà del cliente e crittografati con la chiave o con una chiave gestita dal Chiave gestita da AWS cliente. Per ulteriori informazioni, consulta Configura la crittografia lato server per gli artefatti archiviati in Amazon S3 per CodePipeline.

Crittografia dei dati in transito

Tutte le service-to-service comunicazioni sono crittografate in transito tramite SSL/TLS.

Gestione delle chiavi di crittografia

Se scegli l'opzione predefinita per la crittografia degli elementi del codice, utilizza il. CodePipeline Chiave gestita da AWS Non è possibile modificarlo o eliminarlo. Chiave gestita da AWS Se utilizzi una chiave gestita dal cliente AWS KMS per crittografare o decrittografare gli artefatti nel bucket S3, puoi modificare o ruotare questa chiave gestita dal cliente secondo necessità.

> **Important**
>
> CodePipeline supporta solo chiavi KMS simmetriche. Non utilizzare una chiave KMS asimmetrica per crittografare i dati nel bucket S3.

Configura la crittografia lato server per gli artefatti archiviati in Amazon S3 per CodePipeline

Esistono due modi per configurare la crittografia lato server per gli artefatti di Amazon S3:

- CodePipeline crea un bucket di artefatti S3 ed è predefinito Chiave gestita da AWS quando crei una pipeline utilizzando la procedura guidata Create Pipeline. Chiave gestita da AWS Viene crittografato insieme ai dati degli oggetti e gestito da. AWS
- È possibile creare e gestire la propria chiave gestita dai clienti.

> **Important**
>
> CodePipeline supporta solo chiavi KMS simmetriche. Non utilizzare una chiave KMS asimmetrica per crittografare i dati nel bucket S3.

Se utilizzi la chiave S3 predefinita, non puoi modificarla o eliminarla. Chiave gestita da AWS Se stai utilizzando una chiave gestita dal cliente AWS KMS per crittografare o decrittografare gli artefatti nel bucket S3, puoi modificare o ruotare questa chiave gestita dal cliente secondo necessità.

Amazon S3 supporta le policy di bucket che puoi utilizzare per la crittografia lato server per tutti gli oggetti archiviati nel bucket. Ad esempio, la seguente policy di bucket rifiuta a chiunque l'autorizzazione al caricamento dell'oggetto (`s3:PutObject`) se la richiesta non include l'intestazione `x-amz-server-side-encryption` che richiede la codifica lato server con SSE-KMS.

```
{
    "Version": "2012-10-17",
    "Id": "SSEAndSSLPolicy",
    "Statement": [
        {
            "Sid": "DenyUnEncryptedObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::codepipeline-us-west-2-89050EXAMPLE/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "aws:kms"
                }
            }
        },
        {
            "Sid": "DenyInsecureConnections",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::codepipeline-us-west-2-89050EXAMPLE/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": "false"
                }
            }
        }
    ]
}
```

Per ulteriori informazioni sulla crittografia lato server e AWS KMS, consulta Protezione dei dati utilizzando la crittografia lato server e Protezione dei dati utilizzando la crittografia lato server con chiavi KMS archiviate in (SSE-KMS). AWS Key Management Service

Per ulteriori informazioni in merito, consulta la Guida per gli sviluppatori. AWS KMSAWS Key Management Service

Argomenti

- Visualizza il tuo Chiave gestita da AWS
- Configura la crittografia lato server per i bucket S3 utilizzando o AWS CloudFormationAWS CLI

Visualizza il tuo Chiave gestita da AWS

Quando utilizzi la procedura guidata Create Pipeline (Crea pipeline) per creare la prima pipeline, un bucket S3 viene automaticamente creato nella stessa regione in cui è stata creata la pipeline. Il bucket viene utilizzato per archiviare gli artefatti pipeline. Quando una pipeline viene eseguita, gli artefatti vengono inseriti nel bucket S3 e recuperati dallo stesso. Per impostazione predefinita, CodePipeline utilizza la crittografia lato server con l' AWS KMS utilizzo di Chiave gestita da AWS per Amazon S3 (`aws/s3`la chiave). Questa Chiave gestita da AWS viene creata e archiviata nel tuo account. AWS Quando gli artefatti vengono recuperati dal bucket S3, CodePipeline utilizza lo stesso processo SSE-KMS per decrittografare l'artefatto.

Per visualizzare informazioni sul tuo Chiave gestita da AWS

1. Accedi a AWS Management Console e apri la AWS KMS console.
2. Se viene visualizzata una pagina di benvenuto, scegli Inizia subito.
3. Nel riquadro di navigazione del servizio, scegli chiavi AWS gestite.
4. Scegli la regione per la tua pipeline. Ad esempio, se la pipeline è stata creata in`us-east-2`, assicurati che il filtro sia impostato su Stati Uniti orientali (Ohio).

   Per ulteriori informazioni sulle regioni e gli endpoint disponibili per CodePipeline, consulta AWS CodePipeline endpoint e quote.
5. Nell'elenco, scegli la chiave con l'alias utilizzato per la tua pipeline (per impostazione predefinita, aws/s3). Vengono visualizzate informazioni di base sulla chiave.

Configura la crittografia lato server per i bucket S3 utilizzando o AWS CloudFormationAWS CLI

Quando utilizzi AWS CloudFormation o AWS CLI crei una pipeline, devi configurare manualmente la crittografia lato server. Utilizza la policy bucket di esempio riportata sopra, quindi crea la tua chiave gestita dal cliente. Puoi anche usare le tue chiavi al posto di. Chiave gestita da AWS Alcuni motivi per scegliere la propria chiave includono:

- Quando desideri ruotare la chiave in base a una pianificazione per soddisfare requisiti di business o di sicurezza dell'organizzazione.

- Quando desideri creare una pipeline che utilizza le risorse associate a un altro account AWS . Ciò richiede l'uso di una chiave gestita dal cliente. Per ulteriori informazioni, consulta Crea una pipeline CodePipeline che utilizzi le risorse di un altro account AWS.

Le best practice di crittografia scoraggiano il riutilizzo esteso delle chiavi di crittografia. Come best practice, ruota la chiave regolarmente. Per creare nuovo materiale crittografico per le AWS KMS chiavi, è possibile creare una chiave gestita dal cliente e quindi modificare le applicazioni o gli alias per utilizzare la nuova chiave gestita dal cliente. In alternativa, puoi abilitare la rotazione automatica delle chiavi per una chiave gestita dal cliente esistente.

Per ruotare la chiave gestita dal cliente, consulta Rotazione delle chiavi.

> **Important**
>
> CodePipeline supporta solo chiavi KMS simmetriche. Non utilizzare una chiave KMS asimmetrica per crittografare i dati nel bucket S3.

Utilizzalo per tenere traccia delle password del database o delle chiavi AWS Secrets Manager API di terze parti

Ti consigliamo di utilizzarle AWS Secrets Manager per ruotare, gestire e recuperare le credenziali del database, le chiavi API e altri segreti durante tutto il loro ciclo di vita. Secrets Manager consente di sostituire le credenziali codificate nel codice (comprese le password) con una chiamata API a Secrets Manager per recuperare il segreto a livello di codice. Per ulteriori informazioni, consulta What Is AWS Secrets Manager? nella Guida per l'utente di AWS Secrets Manager.

Per le pipeline in cui si passano parametri segreti (come le credenziali OAuth) in un AWS CloudFormation modello, è necessario includere riferimenti dinamici nel modello che accedono ai segreti archiviati in Secrets Manager. Per il modello di ID di riferimento e gli esempi, vedere Secrets Manager Secrets nella Guida AWS CloudFormation per l'utente. Per un esempio che utilizza riferimenti dinamici in uno snippet di modello per GitHub webhook in una pipeline, vedi Webhook Resource Configuration.

Consulta anche

Le risorse correlate seguenti possono rivelarsi utili durante l'utilizzo della gestione dei segreti.

- Secrets Manager può ruotare automaticamente le credenziali del database, ad esempio per la rotazione dei segreti di Amazon RDS. Per ulteriori informazioni, consulta Rotating Your AWS Secrets Manager Secrets nella Guida per l'utente di AWS Secrets Manager.
- Per visualizzare le istruzioni per l'aggiunta di riferimenti dinamici di Secrets Manager ai modelli AWS CloudFormation , consulta https://aws.amazon.com/blogs/security/how-to-create-and-retrieve-secrets-managed-in-aws-secrets-manager-using-aws-cloudformation-template/.

Gestione delle identità e degli accessi per l' AWS CodePipeline

AWS Identity and Access Management (IAM) è un software Servizio AWS che aiuta un amministratore a controllare in modo sicuro l'accesso alle AWS risorse. Gli amministratori IAM controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (disporre delle autorizzazioni) a utilizzare le risorse. CodePipeline IAM è uno Servizio AWS strumento che puoi utilizzare senza costi aggiuntivi.

Argomenti

Destinatari

Il modo in cui usi AWS Identity and Access Management (IAM) varia a seconda del lavoro che CodePipeline svolgi.

Utente del servizio: se utilizzi il CodePipeline servizio per svolgere il tuo lavoro, l'amministratore ti fornisce le credenziali e le autorizzazioni necessarie. Man mano che utilizzi più CodePipeline funzionalità per svolgere il tuo lavoro, potresti aver bisogno di autorizzazioni aggiuntive. La comprensione della gestione dell'accesso ti consente di richiedere le autorizzazioni corrette

all'amministratore. Se non riesci ad accedere a una funzionalità in CodePipeline, consultaRisoluzione dei problemi di identità e accesso in AWS CodePipeline.

Amministratore del servizio: se sei responsabile delle CodePipeline risorse della tua azienda, probabilmente hai pieno accesso a CodePipeline. È tuo compito determinare a quali CodePipeline funzionalità e risorse devono accedere gli utenti del servizio. Devi inviare le richieste all'amministratore IAM per cambiare le autorizzazioni degli utenti del servizio. Esamina le informazioni contenute in questa pagina per comprendere i concetti di base relativi a IAM. Per saperne di più su come la tua azienda può utilizzare IAM con CodePipeline, consultaCome AWS CodePipeline funziona con IAM.

Amministratore IAM: se sei un amministratore IAM, potresti voler conoscere i dettagli su come scrivere policy a cui gestire l'accesso CodePipeline. Per visualizzare esempi di policy CodePipeline basate sull'identità che puoi utilizzare in IAM, consulta. Esempi di policy di AWS CodePipeline basate su identità

Autenticazione con identità

L'autenticazione è il modo in cui accedi AWS utilizzando le tue credenziali di identità. Devi essere autenticato (aver effettuato l' Utente root dell'account AWS accesso AWS) come utente IAM o assumendo un ruolo IAM.

Puoi accedere AWS come identità federata utilizzando le credenziali fornite tramite una fonte di identità. AWS IAM Identity Center Gli utenti (IAM Identity Center), l'autenticazione Single Sign-On della tua azienda e le tue credenziali di Google o Facebook sono esempi di identità federate. Se accedi come identità federata, l'amministratore ha configurato in precedenza la federazione delle identità utilizzando i ruoli IAM. Quando accedi AWS utilizzando la federazione, assumi indirettamente un ruolo.

A seconda del tipo di utente, puoi accedere al AWS Management Console o al portale di AWS accesso. Per ulteriori informazioni sull'accesso a AWS, vedi Come accedere al tuo Account AWS nella Guida per l'Accedi ad AWS utente.

Se accedi a AWS livello di codice, AWS fornisce un kit di sviluppo software (SDK) e un'interfaccia a riga di comando (CLI) per firmare crittograficamente le tue richieste utilizzando le tue credenziali. Se non utilizzi AWS strumenti, devi firmare tu stesso le richieste. Per ulteriori informazioni sull'utilizzo del metodo consigliato per firmare autonomamente le richieste, consulta Signing AWS API request nella IAM User Guide.

A prescindere dal metodo di autenticazione utilizzato, potrebbe essere necessario specificare ulteriori informazioni sulla sicurezza. Ad esempio, ti AWS consiglia di utilizzare l'autenticazione a più fattori (MFA) per aumentare la sicurezza del tuo account. Per ulteriori informazioni, consulta Autenticazione a più fattori nella Guida per l'utente di AWS IAM Identity Center e Utilizzo dell'autenticazione a più fattori (MFA) in AWS nella Guida per l'utente di IAM.

Utente root dell'account AWS

Quando ne crei un Account AWS, inizi con un'unica identità di accesso che ha accesso completo a tutte Servizi AWS le risorse dell'account. Questa identità è denominata utente Account AWS root ed è accessibile effettuando l'accesso con l'indirizzo e-mail e la password utilizzati per creare l'account. Si consiglia vivamente di non utilizzare l'utente root per le attività quotidiane. Conservare le credenziali dell'utente root e utilizzarle per eseguire le operazioni che solo l'utente root può eseguire. Per un elenco completo delle attività che richiedono l'accesso come utente root, consulta la sezione Attività che richiedono le credenziali dell'utente root nella Guida per l'utente di IAM.

Utenti e gruppi IAM

Un utente IAM è un'identità interna Account AWS che dispone di autorizzazioni specifiche per una singola persona o applicazione. Ove possibile, consigliamo di fare affidamento a credenziali temporanee invece di creare utenti IAM con credenziali a lungo termine come le password e le chiavi di accesso. Tuttavia, per casi d'uso specifici che richiedono credenziali a lungo termine con utenti IAM, si consiglia di ruotare le chiavi di accesso. Per ulteriori informazioni, consulta la pagina Rotazione periodica delle chiavi di accesso per casi d'uso che richiedono credenziali a lungo termine nella Guida per l'utente di IAM.

Un gruppo IAM è un'identità che specifica un insieme di utenti IAM. Non è possibile eseguire l'accesso come gruppo. È possibile utilizzare gruppi per specificare le autorizzazioni per più utenti alla volta. I gruppi semplificano la gestione delle autorizzazioni per set di utenti di grandi dimensioni. Ad esempio, è possibile avere un gruppo denominato IAMAdmins e concedere a tale gruppo le autorizzazioni per amministrare le risorse IAM.

Gli utenti sono diversi dai ruoli. Un utente è associato in modo univoco a una persona o un'applicazione, mentre un ruolo è destinato a essere assunto da chiunque ne abbia bisogno. Gli utenti dispongono di credenziali a lungo termine permanenti, mentre i ruoli forniscono credenziali temporanee. Per ulteriori informazioni, consulta Quando creare un utente IAM (invece di un ruolo) nella Guida per l'utente di IAM.

Ruoli IAM

Un ruolo IAM è un'identità interna all'utente Account AWS che dispone di autorizzazioni specifiche. È simile a un utente IAM, ma non è associato a una persona specifica. Puoi assumere temporaneamente un ruolo IAM in AWS Management Console cambiando ruolo. Puoi assumere un ruolo chiamando un'operazione AWS CLI o AWS API o utilizzando un URL personalizzato. Per ulteriori informazioni sui metodi per l'utilizzo dei ruoli, consulta Utilizzo di ruoli IAM nella Guida per l'utente di IAM.

I ruoli IAM con credenziali temporanee sono utili nelle seguenti situazioni:

- Accesso utente federato: per assegnare le autorizzazioni a una identità federata, è possibile creare un ruolo e definire le autorizzazioni per il ruolo. Quando un'identità federata viene autenticata, l'identità viene associata al ruolo e ottiene le autorizzazioni da esso definite. Per ulteriori informazioni sulla federazione dei ruoli, consulta Creazione di un ruolo per un provider di identità di terza parte nella Guida per l'utente di IAM. Se utilizzi IAM Identity Center, configura un set di autorizzazioni. IAM Identity Center mette in correlazione il set di autorizzazioni con un ruolo in IAM per controllare a cosa possono accedere le identità dopo l'autenticazione. Per ulteriori informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .
- Autorizzazioni utente IAM temporanee: un utente IAM o un ruolo può assumere un ruolo IAM per ottenere temporaneamente autorizzazioni diverse per un'attività specifica.
- Accesso multi-account: è possibile utilizzare un ruolo IAM per permettere a un utente (un principale affidabile) con un account diverso di accedere alle risorse nell'account. I ruoli sono lo strumento principale per concedere l'accesso multi-account. Tuttavia, con alcuni Servizi AWS, è possibile allegare una policy direttamente a una risorsa (anziché utilizzare un ruolo come proxy). Per informazioni sulle differenze tra ruoli e policy basate su risorse per l'accesso multi-account, consulta Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente di IAM.
- Accesso a più servizi: alcuni Servizi AWS utilizzano le funzionalità di altri Servizi AWS. Ad esempio, quando effettui una chiamata in un servizio, è comune che tale servizio esegua applicazioni in Amazon EC2 o archivi oggetti in Amazon S3. Un servizio può eseguire questa operazione utilizzando le autorizzazioni dell'entità chiamante, utilizzando un ruolo di servizio o utilizzando un ruolo collegato al servizio.
  - Sessioni di accesso inoltrato (FAS): quando utilizzi un utente o un ruolo IAM per eseguire azioni AWS, sei considerato un principale. Quando si utilizzano alcuni servizi, è possibile eseguire un'operazione che attiva un'altra operazione in un servizio diverso. FAS utilizza le autorizzazioni del principale che chiama an Servizio AWS, combinate con la richiesta Servizio AWS per

effettuare richieste ai servizi downstream. Le richieste FAS vengono effettuate solo quando un servizio riceve una richiesta che richiede interazioni con altri Servizi AWS o risorse per essere completata. In questo caso è necessario disporre delle autorizzazioni per eseguire entrambe le azioni. Per i dettagli delle policy relative alle richieste FAS, consulta la pagina Forward access sessions.

- Ruolo di servizio: un ruolo di servizio è un ruolo IAM che un servizio assume per eseguire azioni per tuo conto. Un amministratore IAM può creare, modificare ed eliminare un ruolo di servizio dall'interno di IAM. Per ulteriori informazioni, consulta la sezione Creazione di un ruolo per delegare le autorizzazioni a un Servizio AWS nella Guida per l'utente di IAM.
- Ruolo collegato al servizio: un ruolo collegato al servizio è un tipo di ruolo di servizio collegato a un. Servizio AWS Il servizio può assumere il ruolo per eseguire un'operazione per tuo conto. I ruoli collegati al servizio vengono visualizzati nel tuo account Account AWS e sono di proprietà del servizio. Un amministratore IAM può visualizzare le autorizzazioni per i ruoli collegati ai servizi, ma non modificarle.

- Applicazioni in esecuzione su Amazon EC2: puoi utilizzare un ruolo IAM per gestire le credenziali temporanee per le applicazioni in esecuzione su un'istanza EC2 e che AWS CLI effettuano richieste API. AWS Ciò è preferibile all'archiviazione delle chiavi di accesso nell'istanza EC2. Per assegnare un AWS ruolo a un'istanza EC2 e renderlo disponibile per tutte le sue applicazioni, crei un profilo di istanza collegato all'istanza. Un profilo dell'istanza contiene il ruolo e consente ai programmi in esecuzione sull'istanza EC2 di ottenere le credenziali temporanee. Per ulteriori informazioni, consulta Utilizzo di un ruolo IAM per concedere autorizzazioni ad applicazioni in esecuzione su istanze di Amazon EC2 nella Guida per l'utente di IAM.

Per informazioni sull'utilizzo dei ruoli IAM, consulta Quando creare un ruolo IAM (invece di un utente) nella Guida per l'utente di IAM.

Gestione dell'accesso con policy

Puoi controllare l'accesso AWS creando policy e collegandole a AWS identità o risorse. Una policy è un oggetto AWS che, se associato a un'identità o a una risorsa, ne definisce le autorizzazioni. AWS valuta queste politiche quando un principale (utente, utente root o sessione di ruolo) effettua una richiesta. Le autorizzazioni nelle policy determinano l'approvazione o il rifiuto della richiesta. La maggior parte delle politiche viene archiviata AWS come documenti JSON. Per ulteriori informazioni sulla struttura e sui contenuti dei documenti delle policy JSON, consulta Panoramica delle policy JSON nella Guida per l'utente di IAM.

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. In altre parole, quale principale può eseguire azioni su quali risorse e in quali condizioni.

Per impostazione predefinita, utenti e ruoli non dispongono di autorizzazioni. Per concedere agli utenti l'autorizzazione a eseguire azioni sulle risorse di cui hanno bisogno, un amministratore IAM può creare policy IAM. Successivamente l'amministratore può aggiungere le policy IAM ai ruoli e gli utenti possono assumere i ruoli.

Le policy IAM definiscono le autorizzazioni relative a un'azione, a prescindere dal metodo utilizzato per eseguirla. Ad esempio, supponiamo di disporre di una policy che consente l'azione `iam:GetRole`. Un utente con tale policy può ottenere informazioni sul ruolo dall' AWS Management Console AWS CLI, dall'o dall' AWS API.

Policy basate su identità

Le policy basate su identità sono documenti di policy di autorizzazione JSON che è possibile allegare a un'identità (utente, gruppo di utenti o ruoli IAM). Tali policy definiscono le azioni che utenti e ruoli possono eseguire, su quali risorse e in quali condizioni. Per informazioni su come creare una policy basata su identità, consulta Creazione di policy IAM nella Guida per l'utente di IAM.

Le policy basate su identità possono essere ulteriormente classificate come policy inline o policy gestite. Le policy inline sono integrate direttamente in un singolo utente, gruppo o ruolo. Le politiche gestite sono politiche autonome che puoi allegare a più utenti, gruppi e ruoli nel tuo Account AWS. Le politiche gestite includono politiche AWS gestite e politiche gestite dai clienti. Per informazioni su come scegliere tra una policy gestita o una policy inline, consulta Scelta fra policy gestite e policy inline nella Guida per l'utente di IAM.

Policy basate su risorse

Le policy basate su risorse sono documenti di policy JSON che è possibile collegare a una risorsa. Gli esempi più comuni di policy basate su risorse sono le policy di attendibilità dei ruoli IAM e le policy dei bucket Amazon S3. Nei servizi che supportano policy basate sulle risorse, gli amministratori dei servizi possono utilizzarle per controllare l'accesso a una risorsa specifica. Quando è collegata a una risorsa, una policy definisce le azioni che un principale può eseguire su tale risorsa e a quali condizioni. È necessario specificare un principale in una policy basata sulle risorse. I principali possono includere account, utenti, ruoli, utenti federati o. Servizi AWS

Le policy basate sulle risorse sono policy inline che si trovano in tale servizio. Non puoi utilizzare le policy AWS gestite di IAM in una policy basata sulle risorse.

Altri tipi di policy

AWS supporta tipi di policy aggiuntivi e meno comuni. Questi tipi di policy possono impostare il numero massimo di autorizzazioni concesse dai tipi di policy più comuni.

- Limiti delle autorizzazioni: un limite delle autorizzazioni è una funzione avanzata nella quale si imposta il numero massimo di autorizzazioni che una policy basata su identità può concedere a un'entità IAM (utente o ruolo IAM). È possibile impostare un limite delle autorizzazioni per un'entità. Le autorizzazioni risultanti sono l'intersezione delle policy basate su identità dell'entità e i relativi limiti delle autorizzazioni. Le policy basate su risorse che specificano l'utente o il ruolo nel campo `Principal` sono condizionate dal limite delle autorizzazioni. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni sui limiti delle autorizzazioni, consulta Limiti delle autorizzazioni per le entità IAM nella Guida per l'utente di IAM.
- Politiche di controllo dei servizi (SCP): le SCP sono politiche JSON che specificano le autorizzazioni massime per un'organizzazione o un'unità organizzativa (OU) in. AWS Organizations AWS Organizations è un servizio per il raggruppamento e la gestione centralizzata di più Account AWS di proprietà dell'azienda. Se abiliti tutte le funzionalità in un'organizzazione, puoi applicare le policy di controllo dei servizi (SCP) a uno o tutti i tuoi account. L'SCP limita le autorizzazioni per le entità presenti negli account dei membri, inclusa ciascuna. Utente root dell'account AWS Per ulteriori informazioni su organizzazioni e policy SCP, consulta la pagina sulle Policy di controllo dei servizi nella Guida per l'utente di AWS Organizations .
- Policy di sessione: le policy di sessione sono policy avanzate che vengono trasmesse come parametro quando si crea in modo programmatico una sessione temporanea per un ruolo o un utente federato. Le autorizzazioni della sessione risultante sono l'intersezione delle policy basate su identità del ruolo o dell'utente e le policy di sessione. Le autorizzazioni possono anche provenire da una policy basata su risorse. Un rifiuto esplicito in una qualsiasi di queste policy sostituisce l'autorizzazione. Per ulteriori informazioni, consulta Policy di sessione nella Guida per l'utente di IAM.

Come AWS CodePipeline funziona con IAM

Prima di utilizzare IAM per gestire l'accesso a CodePipeline, è necessario comprendere con quali funzionalità IAM è disponibile l'uso CodePipeline. Per avere una panoramica di alto livello su come CodePipeline e su altri Servizi AWS aspetti del funzionamento di IAM, consulta Servizi AWS la sezione dedicata alla compatibilità con IAM nella IAM User Guide.

Argomenti

- Policy CodePipeline basate su identità
- CodePipeline politiche basate sulle risorse
- Autorizzazione basata su tag CodePipeline
- CodePipeline Ruoli IAM

Policy CodePipeline basate su identità

Con le policy basate su identità di IAM, è possibile specificare quali azioni e risorse sono consentite o rifiutate, nonché le condizioni in base alle quali le azioni sono consentite o rifiutate. CodePipeline supporta azioni, risorse e chiavi di condizione specifiche. Per informazioni su tutti gli elementi utilizzati in una policy JSON, consulta Documentazione di riferimento degli elementi delle policy JSON IAM nella Guida per l'utente IAM.

Azioni

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire azioni su quali risorse, e in quali condizioni.

L'elemento `Action` di una policy JSON descrive le operazioni che è possibile utilizzare per consentire o negare l'accesso a un criterio. Le azioni politiche in genere hanno lo stesso nome dell'operazione AWS API associata. Ci sono alcune eccezioni, ad esempio le azioni di sola autorizzazione che non hanno un'operazione API corrispondente. Esistono anche alcune operazioni che richiedono più operazioni in una policy. Queste operazioni aggiuntive sono denominate operazioni dipendenti.

Includi le operazioni in una policy per concedere le autorizzazioni a eseguire l'operazione associata.

Le azioni politiche CodePipeline utilizzano il seguente prefisso prima dell'azione:`codepipeline:`.

Ad esempio, per concedere a qualcuno l'autorizzazione a visualizzare le pipeline esistenti nell'account, è possibile includere l'operazione `codepipeline:GetPipeline` nella rispettiva policy. Le dichiarazioni politiche devono includere un `NotAction` elemento `Action` or. CodePipeline definisce il proprio set di azioni che descrivono le attività che è possibile eseguire con questo servizio.

Per specificare più azioni in una sola istruzione, separa ciascuna di esse con una virgola come mostrato di seguito:

```
"Action": [
```

```
      "codepipeline:action1",
      "codepipeline:action2"
```

È possibile specificare più azioni tramite caratteri jolly (*). Ad esempio, per specificare tutte le azioni che iniziano con la parola `Get`, includi la seguente azione:

```
"Action": "codepipeline:Get*"
```

Per un elenco di CodePipeline azioni, consulta Actions Defined by AWS CodePipeline nella IAM User Guide.

Risorse

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire operazioni su quali risorse, e in quali condizioni.

L'elemento JSON `Resource` della policy specifica l'oggetto o gli oggetti ai quali si applica l'azione. Le istruzioni devono includere un elemento `Resource` o un elemento `NotResource`. Come best practice, specifica una risorsa utilizzando il suo nome della risorsa Amazon (ARN). Puoi eseguire questa operazione per azioni che supportano un tipo di risorsa specifico, note come autorizzazioni a livello di risorsa.

Per le azioni che non supportano le autorizzazioni a livello di risorsa, ad esempio le operazioni di elenco, utilizza un carattere jolly (*) per indicare che l'istruzione si applica a tutte le risorse.

```
"Resource": "*"
```

CodePipeline risorse e operazioni

Nel CodePipeline, la risorsa principale è una pipeline. In una policy, utilizzi un Amazon Resource Name (ARN) per identificare la risorsa a cui si applica la policy. CodePipeline supporta altre risorse che possono essere utilizzate con la risorsa principale, come fasi, azioni e azioni personalizzate. e vi si può fare riferimento come risorse secondarie. Alle risorse e alle risorse secondarie sono associati Amazon Resource Names (ARN) univoci. Per ulteriori informazioni sugli ARN, consulta Amazon Resource Names (ARN) Servizio AWS e namespace in. Riferimenti generali di Amazon Web Services Per associare l'ARN della pipeline alla pipeline, puoi trovare l'ARN della pipeline in

Impostazioni nella console. Per ulteriori informazioni, consulta Visualizza l'ARN della pipeline e l'ARN del ruolo di servizio (console).

Tipo di risorsa	Formato ARN
Pipeline	*arn:aws:codepipeline: regione: account: nome-pipeline*
Stage	*arn:aws:codepipeline: regione: account: nome-pipeline/nome-fase*
Azione	*arn:aws:codepipeline: regione: account: nome-pipeline/nome-stage/nome-azione*
Operazione personalizzata	arn:aws:codepipeline: region: account:actiontype: *proprietario* /*categoria* /*provider*/*versione*
Tutte CodePipeline le risorse	arn:aws:codepipeline: *
Tutte le CodePipeline risorse di proprietà dell'account specificato nella regione specificata	*arn:aws:codepipeline: region: account: **

> **Note**
>
> La maggior parte dei servizi in AWS uso considera i due punti (:) o una barra (/) come lo stesso carattere negli ARN. Tuttavia, CodePipeline utilizza una corrispondenza esatta nei modelli e nelle regole degli eventi. Assicurati di utilizzare i caratteri ARN corretti durante la creazione di modelli di eventi, in modo che corrispondano alla sintassi ARN nella pipeline che desideri associare.

Nel CodePipeline, ci sono chiamate API che supportano le autorizzazioni a livello di risorsa. Le autorizzazioni a livello di risorsa indicano se una chiamata API è in grado di specificare un ARN della risorsa o se la chiamata API può solo specificare tutte le risorse utilizzando il carattere jolly. CodePipeline riferimento alle autorizzazioniPer una descrizione dettagliata delle autorizzazioni a

livello di risorsa e un elenco delle chiamate API che supportano le autorizzazioni a livello di risorsa, consulta. CodePipeline

Ad esempio, nella dichiarazione puoi indicare una pipeline specifica (*myPipeline*) utilizzando il relativo ARN come segue:

```
"Resource": "arn:aws:codepipeline:us-east-2:111222333444:myPipeline"
```

Puoi anche specificare tutte le pipeline che appartengono a un account specifico utilizzando il carattere jolly (*) come segue:

```
"Resource": "arn:aws:codepipeline:us-east-2:111222333444:*"
```

Per specificare tutte le risorse o se una determinata operazione API non supporta gli ARN, usa il carattere jolly (*) nell'elemento `Resource` come mostrato di seguito:

```
"Resource": "*"
```

> **Note**
>
> Quando crei policy IAM, segui i consigli di sicurezza standard che prevedono la concessione dei privilegi minimi, ovvero la concessione solo delle autorizzazioni necessarie per eseguire un'attività. Se una chiamata API supporta gli ARN, allora supporta anche le autorizzazioni a livello di risorsa e non è necessario utilizzare il carattere jolly (*).

Alcune chiamate CodePipeline API accettano più risorse (ad esempio,). `GetPipeline` Per specificare più risorse in una sola dichiarazione, separa i relativi ARN con una virgola come mostrato di seguito:

```
"Resource": ["arn1", "arn2"]
```

CodePipeline fornisce una serie di operazioni per utilizzare le CodePipeline risorse. Per un elenco di operazioni disponibili, consulta la sezione CodePipeline riferimento alle autorizzazioni.

Chiavi di condizione

Gli amministratori possono utilizzare le policy AWS JSON per specificare chi ha accesso a cosa. Cioè, quale principale può eseguire azioni su quali risorse, e in quali condizioni.

L'elemento `Condition` (o blocco `Condition`) consente di specificare le condizioni in cui un'istruzione è in vigore. L'elemento `Condition` è facoltativo. Puoi compilare espressioni condizionali che utilizzano operatori di condizione, ad esempio uguale a o minore di, per soddisfare la condizione nella policy con i valori nella richiesta.

Se specifichi più elementi `Condition` in un'istruzione o più chiavi in un singolo elemento `Condition`, questi vengono valutati da AWS utilizzando un'operazione `AND` logica. Se si specificano più valori per una singola chiave di condizione, AWS valuta la condizione utilizzando un'operazione logica. `OR` Tutte le condizioni devono essere soddisfatte prima che le autorizzazioni dell'istruzione vengano concesse.

Puoi anche utilizzare variabili segnaposto quando specifichi le condizioni. Ad esempio, puoi autorizzare un utente IAM ad accedere a una risorsa solo se è stata taggata con il relativo nome utente IAM. Per ulteriori informazioni, consulta Elementi delle policy IAM: variabili e tag nella Guida per l'utente di IAM.

AWS supporta chiavi di condizione globali e chiavi di condizione specifiche del servizio. Per visualizzare tutte le chiavi di condizione AWS globali, consulta le chiavi di contesto delle condizioni AWS globali nella Guida per l'utente IAM.

CodePipeline definisce il proprio set di chiavi di condizione e supporta anche l'utilizzo di alcune chiavi di condizione globali. Per visualizzare tutte le chiavi di condizione AWS globali, consulta AWS Global Condition Context Keys nella IAM User Guide.

Tutte le operazioni Amazon EC2 supportano le chiavi di condizione `aws:RequestedRegion` e `ec2:Region`. Per ulteriori informazioni, consultare Esempio: limitazione dell'accesso a una regione specifica.

Per visualizzare un elenco di chiavi di CodePipeline condizione, consulta Condition Keys for AWS CodePipeline nella IAM User Guide. Per sapere con quali azioni e risorse puoi utilizzare una chiave di condizione, consulta Azioni definite da AWS CodePipeline.

Esempi

Per visualizzare esempi di politiche CodePipeline basate sull'identità, vedere. Esempi di policy di AWS CodePipeline basate su identità

CodePipeline politiche basate sulle risorse

CodePipeline non supporta politiche basate sulle risorse. Tuttavia, viene fornito un esempio di policy basata sulle risorse per il servizio S3 relativo a. CodePipeline

Esempi

Per visualizzare esempi di politiche basate sulle risorse, CodePipeline consulta, Esempi di policy basate su risorse AWS CodePipeline

Autorizzazione basata su tag CodePipeline

Puoi allegare tag alle CodePipeline risorse o passare tag in una richiesta a. CodePipeline Per controllare l'accesso basato su tag, fornisci informazioni sui tag nell'elemento condizione di una policy utilizzando le chiavi di condizione `codepipeline:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* o `aws:TagKeys`. Per ulteriori informazioni sull'etichettatura CodePipeline delle risorse, consultaAssegnazione di tag alle risorse.

Per visualizzare una policy basata sulle identità di esempio per limitare l'accesso a una risorsa basata su tag su tale risorsa, consulta Utilizzo dei tag per controllare l'accesso alle CodePipeline risorse.

CodePipeline Ruoli IAM

Un ruolo IAM è un'entità nel tuo AWS account che dispone di autorizzazioni specifiche.

Utilizzo di credenziali temporanee con CodePipeline

È possibile utilizzare credenziali temporanee per effettuare l'accesso con la federazione, assumere un ruolo IAM o un ruolo multi-account. È possibile ottenere credenziali di sicurezza temporanee chiamando operazioni AWS STS API come AssumeRoleo. GetFederationToken

CodePipeline supporta l'uso di credenziali temporanee.

Ruoli di servizio

CodePipeline consente a un servizio di assumere un ruolo di servizio per conto dell'utente. Questo ruolo consente al servizio di accedere alle risorse in altri servizi per completare un'azione per conto dell'utente. I ruoli dei servizi sono visualizzati nell'account IAM e sono di proprietà dell'account. Ciò significa che un amministratore IAM può modificare le autorizzazioni per questo ruolo. Tuttavia, questo potrebbe pregiudicare la funzionalità del servizio.

CodePipeline supporta i ruoli di servizio.

Esempi di policy di AWS CodePipeline basate su identità

Per impostazione predefinita, gli utenti e i ruoli IAM non sono autorizzati a creare o modificare CodePipeline risorse. Inoltre, non possono eseguire attività utilizzando l' AWS API AWS Management Console AWS CLI, o. Un amministratore IAM deve creare policy IAM che concedono a utenti e ruoli l'autorizzazione per eseguire operazioni API specifiche sulle risorse specificate di cui hanno bisogno. L'amministratore deve quindi allegare queste policy a utenti o IAM che richiedono tali autorizzazioni.

Per informazioni su come creare una policy basata su identità IAM utilizzando questi documenti di policy JSON di esempio, consulta Creazione di policy nella scheda JSON nella Guida per l'utente IAM.

Per informazioni su come creare una pipeline che utilizzi le risorse di un altro account e per i relativi esempi di policy, consultaCrea una pipeline CodePipeline che utilizzi le risorse di un altro account AWS.

Argomenti

- Best practice per le policy
- Visualizzazione di risorse nella console
- Consentire agli utenti di visualizzare le loro autorizzazioni
- Esempi di politiche basate sull'identità (IAM)
- Utilizzo dei tag per controllare l'accesso alle CodePipeline risorse
- Autorizzazioni necessarie per l'uso della console CodePipeline
- AWS politiche gestite per AWS CodePipeline
- Esempi di policy gestite dal cliente

Best practice per le policy

Le politiche basate sull'identità determinano se qualcuno può creare, accedere o eliminare CodePipeline risorse nel tuo account. Queste azioni possono comportare costi aggiuntivi per l' Account AWS. Quando crei o modifichi policy basate su identità, segui queste linee guida e raccomandazioni:

- Inizia con le policy AWS gestite e passa alle autorizzazioni con privilegi minimi: per iniziare a concedere autorizzazioni a utenti e carichi di lavoro, utilizza le politiche gestite che concedono

le autorizzazioni per molti casi d'uso comuni.AWS Sono disponibili nel tuo. Account AWS Ti consigliamo di ridurre ulteriormente le autorizzazioni definendo politiche gestite dai AWS clienti specifiche per i tuoi casi d'uso. Per ulteriori informazioni, consulta Policy gestite da AWS o Policy gestite da AWS per le funzioni dei processi nella Guida per l'utente IAM.

- Applica le autorizzazioni con privilegi minimi: quando imposti le autorizzazioni con le policy IAM, concedi solo le autorizzazioni richieste per eseguire un'attività. Puoi farlo definendo le azioni che possono essere intraprese su risorse specifiche in condizioni specifiche, note anche come autorizzazioni con privilegi minimi. Per ulteriori informazioni sull'utilizzo di IAM per applicare le autorizzazioni, consulta Policy e autorizzazioni in IAM nella Guida per l'utente di IAM.
- Condizioni d'uso nelle policy IAM per limitare ulteriormente l'accesso: per limitare l'accesso ad azioni e risorse puoi aggiungere una condizione alle tue policy. Ad esempio, è possibile scrivere una condizione di policy per specificare che tutte le richieste devono essere inviate utilizzando SSL. Puoi anche utilizzare le condizioni per concedere l'accesso alle azioni del servizio se vengono utilizzate tramite uno specifico Servizio AWS, ad esempio AWS CloudFormation. Per ulteriori informazioni, consulta la sezione Elementi delle policy JSON di IAM: condizione nella Guida per l'utente di IAM.
- Utilizzo di IAM Access Analyzer per convalidare le policy IAM e garantire autorizzazioni sicure e funzionali: IAM Access Analyzer convalida le policy nuove ed esistenti in modo che aderiscano alla sintassi della policy IAM (JSON) e alle best practice di IAM. IAM Access Analyzer offre oltre 100 controlli delle policy e consigli utili per creare policy sicure e funzionali. Per ulteriori informazioni, consulta Convalida delle policy per IAM Access Analyzer nella Guida per l'utente di IAM.
- Richiedi l'autenticazione a più fattori (MFA): se hai uno scenario che richiede utenti IAM o un utente root nel Account AWS tuo, attiva l'MFA per una maggiore sicurezza. Per richiedere la MFA quando vengono chiamate le operazioni API, aggiungi le condizioni MFA alle policy. Per ulteriori informazioni, consulta Configurazione dell'accesso alle API protetto con MFA nella Guida per l'utente di IAM.

Per maggiori informazioni sulle best practice in IAM, consulta Best practice di sicurezza in IAM nella Guida per l'utente di IAM.

Visualizzazione di risorse nella console

La CodePipeline console richiede l'`ListRepositories`autorizzazione per visualizzare un elenco di repository per il tuo AWS account nella AWS regione in cui hai effettuato l'accesso. La console, inoltre, include a funzione Go to resource (Vai alla risorsa) per eseguire rapidamente una ricerca di risorse senza distinzione tra maiuscole e minuscole. Questa ricerca viene eseguita nel tuo AWS

account nella AWS regione in cui hai effettuato l'accesso. Le seguenti risorse sono visualizzate per i seguenti servizi:

- AWS CodeBuild: progetti di compilazione
- AWS CodeCommit: repository
- AWS CodeDeploy: applicazioni
- AWS CodePipeline: pipeline

Per eseguire la ricerca nelle risorse di tutti i servizi, è necessario disporre delle autorizzazioni seguenti:

- CodeBuild: `ListProjects`
- CodeCommit: `ListRepositories`
- CodeDeploy: `ListApplications`
- CodePipeline: `ListPipelines`

I risultati non vengono restituiti per le risorse di un servizio se non hai le autorizzazioni per quel servizio. Anche se hai le autorizzazioni per la visualizzazione delle risorse, alcune risorse non verranno restituite se c'è un esplicito `Deny` a visualizzare tali risorse.

Consentire agli utenti di visualizzare le loro autorizzazioni

Questo esempio mostra in che modo è possibile creare una policy che consente agli utenti IAM di visualizzare le policy inline e gestite che sono cpllegate alla relativa identità utente. Questa politica include le autorizzazioni per completare questa azione sulla console o utilizzando l'API o a livello di codice. AWS CLI AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
```

```
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Esempi di politiche basate sull'identità (IAM)

Puoi collegare le policy alle identità IAM. Ad esempio, puoi eseguire le operazioni seguenti:

- Allega una politica di autorizzazioni a un utente o a un gruppo nel tuo account: per concedere a un utente l'autorizzazione a visualizzare le pipeline nella CodePipeline console, puoi allegare una politica di autorizzazioni a un utente o gruppo a cui appartiene l'utente.
- Collega una policy di autorizzazione a un ruolo (assegnazione di autorizzazioni tra account): per concedere autorizzazioni tra più account, è possibile collegare una policy di autorizzazione basata su identità a un ruolo IAM. Ad esempio, l'amministratore dell'Account A può creare un ruolo per concedere autorizzazioni su più account a un altro account (ad esempio, AWS Account B) oppure un ruolo come segue: Servizio AWS
  1. L'amministratore dell'account A crea un ruolo IAM e attribuisce una policy di autorizzazione al ruolo che concede le autorizzazioni sulle risorse per l'account A.
  2. L'amministratore dell'account A attribuisce una policy di attendibilità al ruolo, identificando l'account B come il principale per tale ruolo.
  3. L'amministratore dell'Account B può quindi delegare le autorizzazioni per assumere il ruolo a qualsiasi utente dell'Account B. In questo modo gli utenti dell'Account B possono creare o

accedere alle risorse nell'Account A. Il responsabile della politica di fiducia può anche essere un Servizio AWS principale se si desidera concedere l' Servizio AWS autorizzazione per assumere il ruolo.

Per ulteriori informazioni sull'uso di IAM per delegare le autorizzazioni, consulta Access Management nella IAM User Guide (Guida per l'utente di IAM).

Di seguito viene illustrato un esempio di politica delle autorizzazioni che concede le autorizzazioni per disabilitare e abilitare le transizioni tra tutte le fasi della pipeline denominata in: `MyFirstPipeline` `us-west-2 region`

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codepipeline:EnableStageTransition",
        "codepipeline:DisableStageTransition"
      ],
      "Resource" : [
        "arn:aws:codepipeline:us-west-2:111222333444:MyFirstPipeline/*"
      ]
    }
  ]
}
```

L'esempio seguente mostra una politica nell'account 111222333444 che consente agli utenti di visualizzare, ma non modificare, la pipeline denominata nella console. `MyFirstPipeline` CodePipeline Questa policy è basata sulla policy gestita `AWSCodePipeline_ReadOnlyAccess`, ma poiché è relativa alla pipeline `MyFirstPipeline`, non può utilizzare direttamente la policy gestita. Se non desideri limitare la policy a una pipeline specifica, prendi in considerazione l'utilizzo di una delle politiche gestite create e gestite da. CodePipeline Per ulteriori informazioni, consulta la sezione relativa all'utilizzo di policy gestite. Devi collegare questa policy a un ruolo IAM che crei per l'accesso, ad esempio un ruolo denominato`CrossAccountPipelineViewers`:

```
{
  "Statement": [
    {
      "Action": [
```

```
        "codepipeline:GetPipeline",
        "codepipeline:GetPipelineState",
        "codepipeline:GetPipelineExecution",
        "codepipeline:ListPipelineExecutions",
        "codepipeline:ListActionExecutions",
        "codepipeline:ListActionTypes",
        "codepipeline:ListPipelines",
        "codepipeline:ListTagsForResource",
        "iam:ListRoles",
        "s3:ListAllMyBuckets",
        "codecommit:ListRepositories",
        "codedeploy:ListApplications",
        "lambda:ListFunctions",
        "codestar-notifications:ListNotificationRules",
        "codestar-notifications:ListEventTypes",
        "codestar-notifications:ListTargets"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:codepipeline:us-west-2:111222333444:MyFirstPipeline"
    },
    {
      "Action": [
        "codepipeline:GetPipeline",
        "codepipeline:GetPipelineState",
        "codepipeline:GetPipelineExecution",
        "codepipeline:ListPipelineExecutions",
        "codepipeline:ListActionExecutions",
        "codepipeline:ListActionTypes",
        "codepipeline:ListPipelines",
        "codepipeline:ListTagsForResource",
        "iam:ListRoles",
        "s3:GetBucketPolicy",
        "s3:GetObject",
        "s3:ListBucket",
        "codecommit:ListBranches",
        "codedeploy:GetApplication",
        "codedeploy:GetDeploymentGroup",
        "codedeploy:ListDeploymentGroups",
        "elasticbeanstalk:DescribeApplications",
        "elasticbeanstalk:DescribeEnvironments",
        "lambda:GetFunctionConfiguration",
        "opsworks:DescribeApps",
        "opsworks:DescribeLayers",
        "opsworks:DescribeStacks"
```

```
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Sid": "CodeStarNotificationsReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "codestar-notifications:DescribeNotificationRule"
      ],
      "Resource": "*",
      "Condition": {
        "StringLike": {
          "codestar-notifications:NotificationsForResource": "arn:aws:codepipeline:*"
        }
      }
    }
  ],
  "Version": "2012-10-17"
}
```

Dopo aver creato questa policy, crea il ruolo IAM nell'account 111222333444 e collega la policy a quel ruolo. Nelle relazioni di fiducia del ruolo, devi aggiungere l' AWS account che assumerà questo ruolo. L'esempio seguente mostra una politica che consente agli utenti dell' AWS account *1111 di* assumere i ruoli definiti nell'account 111222333444:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111111111111:root"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

L'esempio seguente mostra una politica creata nell' AWS account *1111 che consente agli utenti di assumere il ruolo denominato nell'account 111222333444*: *CrossAccountPipelineViewers*

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::111222333444:role/CrossAccountPipelineViewers"
        }
    ]
}
```

Puoi creare policy IAM per limitare le chiamate e le risorse a cui gli utenti del tuo account hanno accesso e quindi collegare tali policy al tuo utente amministrativo. Per ulteriori informazioni su come creare ruoli IAM e per scoprire esempi di istruzioni sulle policy IAM CodePipeline, consultaEsempi di policy gestite dal cliente.

Utilizzo dei tag per controllare l'accesso alle CodePipeline risorse

Le condizioni nelle dichiarazioni delle policy IAM fanno parte della sintassi utilizzata per specificare le autorizzazioni per le risorse richieste dalle azioni. CodePipeline L'utilizzo di tag nelle condizioni è un modo per controllare l'accesso alle risorse e alle richieste. Per informazioni sull'etichettatura delle CodePipeline risorse, consulta. Assegnazione di tag alle risorse In questo argomento viene illustrato il controllo degli accessi basato su tag.

Durante la progettazione di policy IAM, potrebbe essere necessario impostare autorizzazioni granulari concedendo l'accesso a risorse specifiche. Poiché il numero di risorse che puoi gestire cresce, questa operazione diventa più difficile. Il tagging delle risorse e l'utilizzo di tag in condizioni di dichiarazione di policy possono semplificare questa attività. Puoi concedere l'accesso in blocco a qualsiasi risorsa con un determinato tag. Quindi applicare ripetutamente questo tag a risorse pertinenti, durante la creazione o in seguito.

I tag possono essere collegati alla risorsa o trasferiti nella richiesta verso servizi che supportano il tagging. In CodePipeline, le risorse possono avere tag e alcune azioni possono includere tag. Quando si crea una policy IAM, è possibile utilizzare le chiavi di condizione di tag per controllare:

- Quali utenti possono eseguire operazioni su una risorsa della pipeline, in base ai tag di cui la risorsa dispone già.
- Quali tag possono essere passati in una richiesta di operazione.
- Se delle chiavi di tag specifiche possono essere utilizzate in una richiesta.

Gli operatori di condizioni stringa consentono di creare elementi `Condition` che limitano l'accesso in base al confronto con una chiave con un valore di stringa. È possibile aggiungere `IfExists` alla fine di qualsiasi condizione il nome dell'operatore tranne la condizione Null. Questa aggiunta ha lo scopo di dichiarare che "se la chiave di policy è presente nel contesto della richiesta, la chiave deve essere elaborata come specificato nella policy". Se la chiave non è presente, l'elemento della condizione viene valutato come "true". Ad esempio, è possibile utilizzare `StringEqualsIfExists` per limitare per condizione chiavi che potrebbero non essere presenti su altri tipi di risorse.

Per la sintassi e la semantica complete delle chiavi di condizione dei tag, consulta Controllo dell'accesso tramite tag. Per ulteriori informazioni sulle chiavi di condizione, consulta le seguenti risorse. Gli esempi di CodePipeline policy in questa sezione si allineano alle seguenti informazioni sulle chiavi di condizione e le ampliano con esempi di sfumature, CodePipeline ad esempio l'annidamento delle risorse.

- Operatori di condizione stringa
- Servizi AWS che funzionano con IAM
- Sintassi delle SCP
- Elementi della policy JSON IAM: Condition
- aws: RequestTag /tag-key
- Chiavi di condizione per CodePipeline

Gli esempi seguenti mostrano come specificare le condizioni dei tag nelle politiche per CodePipeline gli utenti.

Example 1: Limitare le operazioni in base ai tag nella richiesta

La politica degli utenti `AWSCodePipeline_FullAccess` gestiti offre agli utenti il permesso illimitato di eseguire qualsiasi CodePipeline azione su qualsiasi risorsa.

La seguente politica limita questo potere e nega agli utenti non autorizzati l'autorizzazione a creare pipeline in cui nella richiesta sono elencati tag specifici. A tale scopo, nega l'operazione `CreatePipeline` se nella richiesta è specificato un tag denominato `Project` con uno dei valori `ProjectA` o `ProjectB`. La chiave di condizione `aws:RequestTag` viene utilizzata per controllare quali tag possono essere passati in una richiesta IAM.

Nell'esempio seguente, l'intento della policy è negare agli utenti non autorizzati l'autorizzazione a creare una pipeline con i valori dei tag specificati. Tuttavia, la creazione di una pipeline richiede l'accesso a risorse oltre alla pipeline stessa (ad esempio, azioni e fasi della pipeline). Poiché il criterio

`'Resource'` specificato è`'*'`, il criterio viene valutato rispetto a ogni risorsa che dispone di un ARN e viene creata durante la creazione della pipeline. Queste risorse aggiuntive non dispongono della chiave tag condition, quindi il `StringEquals` controllo ha esito negativo e all'utente non viene concessa la possibilità di creare alcuna pipeline. Per risolvere questo problema, utilizzare l'operatore di condizione `StringEqualsIfExists`. In questo modo, il test viene effettuato solo se la chiave di condizione esiste.

Potresti leggere quanto segue: «Se la risorsa da controllare ha una chiave di `"RequestTag/Project"` condizione del tag, consenti l'azione solo se il valore della chiave inizia con`projectA`. Se la risorsa verificata non include quella chiave di condizione, non preoccuparti".

Inoltre, la politica impedisce a questi utenti non autorizzati di manomettere le risorse utilizzando la chiave di `aws:TagKeys` condizione per impedire che le azioni di modifica dei tag includano gli stessi valori dei tag. L'amministratore di un cliente deve collegare questa policy IAM agli utenti amministrativi non autorizzati, oltre alla policy per gli utenti gestiti.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codepipeline:CreatePipeline",
        "codepipeline:TagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEqualsIfExists": {
          "aws:RequestTag/Project": ["ProjectA", "ProjectB"]
        }
      }
    },
    {
      "Effect": "Deny",
      "Action": [
        "codepipeline:UntagResource"
      ],
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["Project"]
        }
```

```
      }
    }
  ]
}
```

Example 2: Limita le azioni di etichettatura in base ai tag delle risorse

La politica degli utenti `AWSCodePipeline_FullAccess` gestiti offre agli utenti autorizzazioni illimitate per eseguire qualsiasi CodePipeline azione su qualsiasi risorsa.

La policy seguente limita questa capacità e nega agli utenti non autorizzati l'autorizzazione a eseguire operazioni su pipeline specifiche di progetto. A tale scopo, rifiuta alcune operazioni se la risorsa ha un tag denominato `Project` con uno dei valori `ProjectA` o `ProjectB`. La chiave di condizione `aws:ResourceTag` viene utilizzata per controllare l'accesso alle risorse in base ai tag su tali risorse. L'amministratore di un cliente deve collegare questa policy IAM a utenti IAM non autorizzati oltre alla policy gestita dall'utente.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codepipeline:TagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Project": ["ProjectA", "ProjectB"]
        }
      }
    }
  ]
}
```

Example 3: Consentire operazioni in base ai tag nella richiesta

La seguente politica concede agli utenti il permesso di creare pipeline di sviluppo in. CodePipeline

A questo scopo, consente le operazioni `CreatePipeline`e `TagResource` se la richiesta specifica un tag denominato `Project` con il valore `ProjectA`. In altre parole, l'unica chiave di tag che può essere specificata è`Project`, e il suo valore deve essere. `ProjectA`

La chiave `aws:RequestTag` condition viene utilizzata per controllare quali tag possono essere passati in una richiesta IAM. La condizione `aws:TagKeys` garantisce che la chiave tag rileva la distinzione tra maiuscole e minuscole. Questa policy è utile per gli utenti o i ruoli a cui non è associata la policy per gli utenti `AWSCodePipeline_FullAccess` gestiti. La policy gestita offre agli utenti autorizzazioni illimitate per eseguire qualsiasi CodePipeline azione su qualsiasi risorsa.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codepipeline:CreatePipeline",
        "codepipeline:TagResource"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/Project": "ProjectA"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["Project"]
        }
      }
    }
  ]
}
```

Example 4: Limita le azioni di rimozione dei tag in base ai tag delle risorse

La politica degli utenti `AWSCodePipeline_FullAccess` gestiti offre agli utenti autorizzazioni illimitate per eseguire qualsiasi CodePipeline azione su qualsiasi risorsa.

La policy seguente limita questa capacità e nega agli utenti non autorizzati l'autorizzazione a eseguire operazioni su pipeline specifiche di progetto. A tale scopo, rifiuta alcune operazioni se la risorsa ha un tag denominato `Project` con uno dei valori `ProjectA` o `ProjectB`.

Inoltre, la politica impedisce a questi utenti non autorizzati di manomettere le risorse utilizzando la chiave di `aws:TagKeys` condizione per impedire che le azioni di modifica dei tag rimuovano completamente il tag. `Project` L'amministratore di un cliente deve collegare questa policy IAM a utenti o ruoli non autorizzati, oltre alla policy per gli utenti gestiti.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codepipeline:UntagResource"
      ],
      "Resource": "*",
      "Condition": {
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["Project"]
        }
      }
    }
  ]
}
```

Autorizzazioni necessarie per l'uso della console CodePipeline

Per CodePipeline utilizzarla nella CodePipeline console, devi disporre di un set minimo di autorizzazioni per i seguenti servizi:

- AWS Identity and Access Management
- Amazon Simple Storage Service

Queste autorizzazioni ti consentono di descrivere altre AWS risorse per il tuo AWS account.

A seconda degli altri servizi incorporati nelle pipeline, potrebbero essere necessarie autorizzazioni da uno o più dei seguenti:

- AWS CodeCommit
- CodeBuild
- AWS CloudFormation
- AWS CodeDeploy
- AWS Elastic Beanstalk
- AWS Lambda
- AWS OpsWorks

Se decidi di creare una policy IAM più restrittiva delle autorizzazioni minime richieste, la console non funzionerà come previsto per gli utenti con tale policy IAM. Per garantire che tali utenti possano continuare a utilizzare la CodePipeline console, allega anche la policy `AWSCodePipeline_ReadOnlyAccess` gestita all'utente, come descritto in AWS politiche gestite per AWS CodePipeline.

Non è necessario consentire autorizzazioni minime per la console per gli utenti che effettuano chiamate all' AWS CLI o all' CodePipeline API.

AWS politiche gestite per AWS CodePipeline

Una politica AWS gestita è una politica autonoma creata e amministrata da AWS. AWS le politiche gestite sono progettate per fornire autorizzazioni per molti casi d'uso comuni, in modo da poter iniziare ad assegnare autorizzazioni a utenti, gruppi e ruoli.

Tieni presente che le policy AWS gestite potrebbero non concedere le autorizzazioni con il privilegio minimo per i tuoi casi d'uso specifici, poiché sono disponibili per tutti i clienti. AWS Consigliamo pertanto di ridurre ulteriormente le autorizzazioni definendo policy gestite dal cliente specifiche per i tuoi casi d'uso.

Non è possibile modificare le autorizzazioni definite nelle politiche gestite. AWS Se AWS aggiorna le autorizzazioni definite in una politica AWS gestita, l'aggiornamento ha effetto su tutte le identità principali (utenti, gruppi e ruoli) a cui è associata la politica. AWS è più probabile che aggiorni una policy AWS gestita quando ne Servizio AWS viene lanciata una nuova o quando diventano disponibili nuove operazioni API per i servizi esistenti.

Per ulteriori informazioni, consultare Policy gestite da AWSnella Guida per l'utente di IAM.

> **Important**
>
> Le policy `AWSCodePipelineFullAccess` gestite da AWS `AWSCodePipelineReadOnlyAccess` sono state sostituite. Usa le `AWSCodePipeline_ReadOnlyAccess` politiche `AWSCodePipeline_FullAccess` e.

AWS politica gestita: **`AWSCodePipeline_FullAccess`**

Questa è una politica che garantisce l'accesso completo a CodePipeline. Per visualizzare il documento sulla policy JSON nella console IAM, consulta. AWSCodePipeline_FullAccess

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

- `codepipeline`— Concede le autorizzazioni a. CodePipeline
- `chatbot`— Concede le autorizzazioni per consentire ai responsabili di gestire le risorse in. AWS Chatbot
- `cloudformation`— Concede le autorizzazioni per consentire ai responsabili di gestire le pile di risorse in. AWS CloudFormation
- `cloudtrail`— Concede le autorizzazioni per consentire ai responsabili di gestire le risorse di registrazione. CloudTrail
- `codebuild`— Concede le autorizzazioni per consentire ai mandanti di accedere alle risorse di compilazione in. CodeBuild
- `codecommit`— Concede le autorizzazioni per consentire ai responsabili di accedere alle risorse di origine in. CodeCommit
- `codedeploy`— Concede le autorizzazioni per consentire ai responsabili di accedere alle risorse di distribuzione in. CodeDeploy
- `codestar-notifications`— Concede le autorizzazioni per consentire ai responsabili di accedere alle risorse nelle notifiche. AWS CodeStar
- `ec2`— Concede le autorizzazioni per consentire le distribuzioni per gestire il bilanciamento elastico del carico in CodeCatalyst Amazon EC2.
- `ecr`— Concede le autorizzazioni per consentire l'accesso alle risorse in Amazon ECR.
- `elasticbeanstalk`— Concede le autorizzazioni per consentire ai mandanti di accedere alle risorse in Elastic Beanstalk.
- `iam`— Concede le autorizzazioni per consentire ai dirigenti di gestire ruoli e policy in IAM.
- `lambda`— Concede le autorizzazioni per consentire ai responsabili di gestire le risorse in Lambda.
- `events`— Concede le autorizzazioni per consentire ai responsabili di gestire le risorse negli Eventi. CloudWatch

- `opsworks`— Concede le autorizzazioni per consentire ai responsabili di gestire le risorse in. AWS OpsWorks
- `s3`— Concede le autorizzazioni per consentire ai responsabili di gestire le risorse in Amazon S3.
- `sns`— Concede le autorizzazioni per consentire ai mandanti di gestire le risorse di notifica in Amazon SNS.
- `states`— Concede le autorizzazioni per consentire ai mandanti di visualizzare le macchine a stati in. AWS Step Functions Una macchina a stati è costituita da un insieme di stati che gestiscono le attività e le transizioni tra gli stati.

```
{
    "Statement": [
        {
            "Action": [
                "codepipeline:*",
                "cloudformation:DescribeStacks",
                "cloudformation:ListStacks",
                "cloudformation:ListChangeSets",
                "cloudtrail:DescribeTrails",
                "codebuild:BatchGetProjects",
                "codebuild:CreateProject",
                "codebuild:ListCuratedEnvironmentImages",
                "codebuild:ListProjects",
                "codecommit:ListBranches",
                "codecommit:GetReferences",
                "codecommit:ListRepositories",
                "codedeploy:BatchGetDeploymentGroups",
                "codedeploy:ListApplications",
                "codedeploy:ListDeploymentGroups",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ecr:DescribeRepositories",
                "ecr:ListImages",
                "ecs:ListClusters",
                "ecs:ListServices",
                "elasticbeanstalk:DescribeApplications",
                "elasticbeanstalk:DescribeEnvironments",
                "iam:ListRoles",
                "iam:GetRole",
                "lambda:ListFunctions",
```

```
                "events:ListRules",
                "events:ListTargetsByRule",
                "events:DescribeRule",
                "opsworks:DescribeApps",
                "opsworks:DescribeLayers",
                "opsworks:DescribeStacks",
                "s3:ListAllMyBuckets",
                "sns:ListTopics",
                "codestar-notifications:ListNotificationRules",
                "codestar-notifications:ListTargets",
                "codestar-notifications:ListTagsforResource",
                "codestar-notifications:ListEventTypes",
                "states:ListStateMachines"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Sid": "CodePipelineAuthoringAccess"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:ListBucket",
                "s3:GetBucketPolicy",
                "s3:GetBucketVersioning",
                "s3:GetObjectVersion",
                "s3:CreateBucket",
                "s3:PutBucketPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:s3::*:codepipeline-*",
            "Sid": "CodePipelineArtifactsReadWriteAccess"
        },
        {
            "Action": [
                "cloudtrail:PutEventSelectors",
                "cloudtrail:CreateTrail",
                "cloudtrail:GetEventSelectors",
                "cloudtrail:StartLogging"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:cloudtrail:*:*:trail/codepipeline-source-trail",
            "Sid": "CodePipelineSourceTrailReadWriteAccess"
        },
        {
```

```
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:iam::*:role/service-role/cwe-role-*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "events.amazonaws.com"
                    ]
                }
            },
            "Sid": "EventsIAMPassRole"
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "codepipeline.amazonaws.com"
                    ]
                }
            },
            "Sid": "CodePipelineIAMPassRole"
        },
        {
            "Action": [
                "events:PutRule",
                "events:PutTargets",
                "events:DeleteRule",
                "events:DisableRule",
                "events:RemoveTargets"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:events:*:*:rule/codepipeline-*"
            ],
            "Sid": "CodePipelineEventsReadWriteAccess"
```

```
        },
        {
            "Sid": "CodeStarNotificationsReadWriteAccess",
            "Effect": "Allow",
            "Action": [
                "codestar-notifications:CreateNotificationRule",
                "codestar-notifications:DescribeNotificationRule",
                "codestar-notifications:UpdateNotificationRule",
                "codestar-notifications:DeleteNotificationRule",
                "codestar-notifications:Subscribe",
                "codestar-notifications:Unsubscribe"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "codestar-notifications:NotificationsForResource":
 "arn:aws:codepipeline:*"
                }
            }
        },
        {
            "Sid": "CodeStarNotificationsSNSTopicCreateAccess",
            "Effect": "Allow",
            "Action": [
                "sns:CreateTopic",
                "sns:SetTopicAttributes"
            ],
            "Resource": "arn:aws:sns:*:*:codestar-notifications*"
        },
        {
            "Sid": "CodeStarNotificationsChatbotAccess",
            "Effect": "Allow",
            "Action": [
                "chatbot:DescribeSlackChannelConfigurations",
                "chatbot:ListMicrosoftTeamsChannelConfigurations"
            ],
            "Resource": "*"
        }
    ],
    "Version": "2012-10-17"
}
```

AWS politica gestita: AWSCodePipeline_ReadOnlyAccess

Questa è una politica che garantisce l'accesso in sola lettura a. CodePipeline Per visualizzare il documento sulla policy JSON nella console IAM, consulta. AWSCodePipeline_ReadOnlyAccess

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

- `codepipeline`— Concede le autorizzazioni alle azioni in. CodePipeline
- `codestar-notifications`— Concede le autorizzazioni per consentire ai mandanti di accedere alle risorse nelle notifiche. AWS CodeStar
- `s3`— Concede le autorizzazioni per consentire ai responsabili di gestire le risorse in Amazon S3.
- `sns`— Concede le autorizzazioni per consentire ai mandanti di gestire le risorse di notifica in Amazon SNS.

```
{
    "Statement": [
        {
            "Action": [
                "codepipeline:GetPipeline",
                "codepipeline:GetPipelineState",
                "codepipeline:GetPipelineExecution",
                "codepipeline:ListPipelineExecutions",
                "codepipeline:ListActionExecutions",
                "codepipeline:ListActionTypes",
                "codepipeline:ListPipelines",
                "codepipeline:ListTagsForResource",
                "s3:ListAllMyBuckets",
                "codestar-notifications:ListNotificationRules",
                "codestar-notifications:ListEventTypes",
                "codestar-notifications:ListTargets"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:ListBucket",
```

```
                "s3:GetBucketPolicy"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:s3::*:codepipeline-*"
        },
        {
            "Sid": "CodeStarNotificationsReadOnlyAccess",
            "Effect": "Allow",
            "Action": [
                "codestar-notifications:DescribeNotificationRule"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "codestar-notifications:NotificationsForResource":
 "arn:aws:codepipeline:*"
                }
            }
        }
    ],
    "Version": "2012-10-17"
}
```

AWS politica gestita: **`AWSCodePipelineApproverAccess`**

Si tratta di una politica che concede l'autorizzazione ad approvare o rifiutare un'azione di approvazione manuale. Per visualizzare il documento sulla policy JSON nella console IAM, consulta.. AWSCodePipelineApproverAccess

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

- `codepipeline`— Concede le autorizzazioni alle azioni in. CodePipeline

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Action": [
                "codepipeline:GetPipeline",
                "codepipeline:GetPipelineState",
                "codepipeline:GetPipelineExecution",
                "codepipeline:ListPipelineExecutions",
                "codepipeline:ListPipelines",
                "codepipeline:PutApprovalResult"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

AWS politica gestita: AWSCodePipelineCustomActionAccess

Questa è una politica che concede l'autorizzazione a creare azioni personalizzate CodePipeline o integrare le risorse Jenkins per azioni di compilazione o test. Per visualizzare il documento relativo alla policy JSON nella console IAM, consulta. AWSCodePipelineCustomActionAccess

Dettagli dell'autorizzazione

Questa policy include le seguenti autorizzazioni:

- `codepipeline`— Concede le autorizzazioni alle azioni in. CodePipeline

```
{
    "Statement": [
        {
            "Action": [
                "codepipeline:AcknowledgeJob",
                "codepipeline:GetJobDetails",
                "codepipeline:PollForJobs",
                "codepipeline:PutJobFailureResult",
                "codepipeline:PutJobSuccessResult"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
```

```
    ],
    "Version": "2012-10-17"
}
```

CodePipeline politiche e notifiche gestite

CodePipeline supporta le notifiche, che possono notificare agli utenti importanti modifiche alle pipeline. Le politiche gestite CodePipeline includono dichiarazioni politiche per la funzionalità di notifica. Per ulteriori informazioni, vedere Cosa sono le notifiche?.

Autorizzazioni correlate alle notifiche nelle policy gestite di accesso completo

Questa politica gestita concede le CodePipeline autorizzazioni per i servizi correlati e CodeCommit CodeBuild CodeDeploy AWS CodeStar le notifiche. La policy concede anche le autorizzazioni necessarie per lavorare con altri servizi che si integrano con le tue pipeline, come Amazon S3, Elastic Beanstalk, Amazon EC2 e. CloudTrail AWS CloudFormation Gli utenti a cui viene applicata questa policy gestita possono anche creare e gestire argomenti Amazon SNS per le notifiche, iscrivere e annullare l'iscrizione degli utenti agli argomenti, elencare argomenti da scegliere come obiettivi per le regole di notifica ed elencare AWS Chatbot i client configurati per Slack.

Le policy gestite `AWSCodePipeline_FullAccess` includono le seguenti dichiarazioni per consentire l'accesso completo alle notifiche.

```
    {
        "Sid": "CodeStarNotificationsReadWriteAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:CreateNotificationRule",
            "codestar-notifications:DescribeNotificationRule",
            "codestar-notifications:UpdateNotificationRule",
            "codestar-notifications:DeleteNotificationRule",
            "codestar-notifications:Subscribe",
            "codestar-notifications:Unsubscribe"
        ],
        "Resource": "*",
        "Condition" : {
            "StringLike" : {"codestar-notifications:NotificationsForResource" :
 "arn:aws:codepipeline:us-west-2:111222333444:MyFirstPipeline"}
        }
    },
    {
        "Sid": "CodeStarNotificationsListAccess",
```

```
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListTargets",
            "codestar-notifications:ListTagsforResource",
            "codestar-notifications:ListEventTypes"
        ],
        "Resource": "*"
    },
    {
        "Sid": "CodeStarNotificationsSNSTopicCreateAccess",
        "Effect": "Allow",
        "Action": [
            "sns:CreateTopic",
            "sns:SetTopicAttributes"
        ],
        "Resource": "arn:aws:sns:*:*:codestar-notifications*"
    },
    {
        "Sid": "SNSTopicListAccess",
        "Effect": "Allow",
        "Action": [
            "sns:ListTopics"
        ],
        "Resource": "*"
    },
    {
        "Sid": "CodeStarNotificationsChatbotAccess",
        "Effect": "Allow",
        "Action": [
            "chatbot:DescribeSlackChannelConfigurations",
            "chatbot:ListMicrosoftTeamsChannelConfigurations"
          ],
       "Resource": "*"
    }
```

Autorizzazioni correlate alle notifiche nelle policy gestite di sola lettura

Le policy gestite `AWSCodePipeline_ReadOnlyAccess` includono le seguenti dichiarazioni per consentire l'accesso in sola lettura alle notifiche. Gli utenti con questa policy applicata possono visualizzare le notifiche per le risorse, ma non possono crearle, gestirle o sottoscriverle.

```
    {
```

```
        "Sid": "CodeStarNotificationsPowerUserAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:DescribeNotificationRule"
        ],
        "Resource": "*",
        "Condition" : {
            "StringLike" : {"codestar-notifications:NotificationsForResource" :
 "arn:aws:codepipeline:us-west-2:111222333444:MyFirstPipeline"}
        }
    },
    {
        "Sid": "CodeStarNotificationsListAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListEventTypes",
            "codestar-notifications:ListTargets"
        ],
        "Resource": "*"
    }
```

Per ulteriori informazioni su IAM e le notifiche, consulta Identity and Access Management for AWS CodeStar Notifications.

AWS CodePipeline aggiornamenti alle politiche AWS gestite

Visualizza i dettagli sugli aggiornamenti delle politiche AWS gestite CodePipeline da quando questo servizio ha iniziato a tenere traccia di queste modifiche. Per ricevere avvisi automatici sulle modifiche a questa pagina, iscriviti al feed RSS nella pagina della cronologia dei CodePipeline documenti.

Modifica	Descrizione	Data
AWSCodePipeline_FullAccess— Aggiornamenti alla politica esistente	CodePipeline ha aggiunto un'autorizzazione a questa politica per il supporto `ListStacks` in AWS CloudFormation.	15 marzo 2024

Modifica	Descrizione	Data
AWSCodePipeline_FullAccess— Aggiornamenti alla politica esistente	Questa politica è stata aggiornata per aggiungere autorizzazioni per AWS Chatbot. Per ulteriori informazioni, consulta CodePipeline politiche e notifiche gestite.	21 giugno 2023
AWSCodePipeline_FullAccesse politiche AWSCodePipeline_ReadOnlyAccessgestite: aggiornamenti alla politica esistente	CodePipeline ha aggiunto un'autorizzazione a queste politiche per supportare un tipo di notifica aggiuntivo utilizzando AWS Chatbot,`chatbot:ListMicrosoftTeamsChannelConfigurations` .	16 maggio 2023
AWSCodePipelineFullAccess— Obsoleto	Questa policy è stata sostituita da `AWSCodePipeline_FullAccess` . Dopo il 17 novembre 2022, questa politica non può essere associata a nuovi utenti, gruppi o ruoli. Per ulteriori informazioni, consulta AWS politiche gestite per AWS CodePipeline.	17 novembre 2022

Modifica	Descrizione	Data
AWSCodePipelineReadOnlyAccess— Obsoleto	Questa policy è stata sostituita da `AWSCodePipeline_ReadOnlyAccess` . Dopo il 17 novembre 2022, questa politica non può essere associata a nuovi utenti, gruppi o ruoli. Per ulteriori informazioni, consulta AWS politiche gestite per AWS CodePipeline.	17 novembre 2022
CodePipeline ha iniziato a tenere traccia delle modifiche	CodePipeline ha iniziato a tenere traccia delle modifiche per le sue politiche AWS gestite.	12 marzo 2021

Esempi di policy gestite dal cliente

In questa sezione, puoi trovare esempi di politiche utente che concedono autorizzazioni per varie azioni. CodePipeline Queste politiche funzionano quando utilizzi l' CodePipeline API, gli AWS SDK o il. AWS CLI Quando si utilizza la console, devi concedere autorizzazioni specifiche aggiuntive alla console. Per ulteriori informazioni, consulta Autorizzazioni necessarie per l'uso della console CodePipeline .

> **Note**
>
> Tutti gli esempi utilizzano la regione Stati Uniti occidentali (Oregon) (`us-west-2`) e contengono ID account fittizi.

Examples (Esempi)

- Esempio 1: concessione di autorizzazioni per ottenere lo stato di una pipeline
- Esempio 2: concessione delle autorizzazioni per abilitare e disabilitare le transizioni tra fasi

- Esempio 3: concessione delle autorizzazioni per ottenere un elenco di tutti i tipi di operazione disponibili
- Esempio 4: concessione delle autorizzazioni per approvare o rifiutare operazioni di approvazione manuali
- Esempio 5: concessione delle autorizzazioni per eseguire il polling di processi per un'operazione personalizzata
- Esempio 6: allega o modifica una politica per l'integrazione di Jenkins con AWS CodePipeline
- Esempio 7: configurazione dell'accesso tra account a una pipeline
- Esempio 8: utilizzo delle risorse AWS associate a un altro account in una pipeline

Esempio 1: concessione di autorizzazioni per ottenere lo stato di una pipeline

L'esempio seguente concede le autorizzazioni per ottenere lo stato della pipeline denominata `MyFirstPipeline`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:GetPipelineState"
            ],
            "Resource": "arn:aws:codepipeline:us-west-2:111222333444:MyFirstPipeline"
        }
    ]
}
```

Esempio 2: concessione delle autorizzazioni per abilitare e disabilitare le transizioni tra fasi

L'esempio seguente concede le autorizzazioni per disabilitare e abilitare le transizioni tra tutte le fasi nella pipeline denominata `MyFirstPipeline`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "codepipeline:DisableStageTransition",
                "codepipeline:EnableStageTransition"
            ],
            "Resource": "arn:aws:codepipeline:us-west-2:111222333444:MyFirstPipeline/*"
        }
    ]
}
```

Per consentire all'utente di disabilitare e abilitare le transizioni per una singola fase in una pipeline, occorre specificare la fase. Ad esempio, per permettere all'utente di abilitare e disabilitare le transizioni per una fase denominata `Staging` in una pipeline denominata `MyFirstPipeline`:

```
"Resource": "arn:aws:codepipeline:us-west-2:111222333444:MyFirstPipeline/Staging"
```

Esempio 3: concessione delle autorizzazioni per ottenere un elenco di tutti i tipi di operazione disponibili

L'esempio seguente concede le autorizzazioni per ottenere un elenco di tutti i tipi di operazione disponibili per pipeline nella regione `us-west-2`:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:ListActionTypes"
            ],
            "Resource": "arn:aws:codepipeline:us-west-2:111222333444:actiontype:*"
        }
    ]
}
```

Esempio 4: concessione delle autorizzazioni per approvare o rifiutare operazioni di approvazione manuali

L'esempio seguente concede le autorizzazioni per approvare o rifiutare operazioni di approvazione manuale in una fase denominata `Staging` in una pipeline denominata `MyFirstPipeline`:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:PutApprovalResult"
            ],
            "Resource": "arn:aws:codepipeline:us-west-2:111222333444:MyFirstPipeline/
Staging/*"
        }
    ]
}
```

Esempio 5: concessione delle autorizzazioni per eseguire il polling di processi per un'operazione personalizzata

L'esempio seguente concede le autorizzazioni per eseguire il polling di processi per l'operazione personalizzata denominata `TestProvider`, che è un tipo di operazione di `Test` nella sua prima versione, in tutte le pipeline:

> **Note**
>
> Il job worker incaricato di un'azione personalizzata potrebbe essere configurato con un AWS account diverso o richiedere un ruolo IAM specifico per funzionare.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:PollForJobs"
            ],
            "Resource": [
                "arn:aws:codepipeline:us-
west-2:111222333444:actionType:Custom/Test/TestProvider/1"
            ]
        }
    ]
}
```

Esempio 6: allega o modifica una politica per l'integrazione di Jenkins con AWS CodePipeline

Se configuri una pipeline per utilizzare Jenkins per la compilazione o il test, crea un'identità separata per tale integrazione e allega una policy IAM con le autorizzazioni minime richieste per l'integrazione tra Jenkins e. CodePipeline Questa policy è identica alla policy gestita `AWSCodePipelineCustomActionAccess`. L'esempio seguente mostra una politica per l'integrazione con Jenkins:

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:AcknowledgeJob",
                "codepipeline:GetJobDetails",
                "codepipeline:PollForJobs",
                "codepipeline:PutJobFailureResult",
                "codepipeline:PutJobSuccessResult"
            ],
            "Resource": "*"
        }
    ],
    "Version": "2012-10-17"
}
```

Esempio 7: configurazione dell'accesso tra account a una pipeline

Puoi configurare l'accesso a pipeline per utenti e gruppi in un altro account AWS . Il modo consigliato consiste nel creare un ruolo nell'account in cui è stata creata la pipeline. Il ruolo dovrebbe consentire agli utenti dell'altro AWS account di assumere quel ruolo e accedere alla pipeline. Per ulteriori informazioni, consulta Procedura guidata: accesso tra account mediante i ruoli.

L'esempio seguente mostra una politica nell'account 80398EXAMPLE che consente agli utenti di visualizzare, ma non modificare, la pipeline denominata nella console. `MyFirstPipeline` CodePipeline Questa policy è basata sulla policy gestita `AWSCodePipeline_ReadOnlyAccess`, ma poiché è relativa alla pipeline `MyFirstPipeline`, non può utilizzare direttamente la policy gestita. Se non desideri limitare la policy a una pipeline specifica, prendi in considerazione l'utilizzo di una delle politiche gestite create e gestite da. CodePipeline Per ulteriori informazioni, consulta la sezione relativa all'utilizzo di policy gestite. Devi collegare questa policy a un ruolo IAM che crei per l'accesso, ad esempio un ruolo denominato`CrossAccountPipelineViewers`:

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codepipeline:GetPipeline",
                "codepipeline:GetPipelineState",
                "codepipeline:ListActionTypes",
                "codepipeline:ListPipelines",
                "iam:ListRoles",
                "s3:GetBucketPolicy",
                "s3:GetObject",
                "s3:ListAllMyBuckets",
                "s3:ListBucket",
                "codedeploy:GetApplication",
                "codedeploy:GetDeploymentGroup",
                "codedeploy:ListApplications",
                "codedeploy:ListDeploymentGroups",
                "elasticbeanstalk:DescribeApplications",
                "elasticbeanstalk:DescribeEnvironments",
                "lambda:GetFunctionConfiguration",
                "lambda:ListFunctions"
            ],
            "Resource": "arn:aws:codepipeline:us-east-2:80398EXAMPLE:MyFirstPipeline"
        }
    ],
    "Version": "2012-10-17"
}
```

Dopo aver creato questa policy, crea il ruolo IAM nell'account 80398EXAMPLE e collega la policy a quel ruolo. Nelle relazioni di fiducia del ruolo, devi aggiungere l' AWS account che assume questo ruolo. L'esempio seguente mostra una politica che consente agli utenti dell' AWS account *1111 di assumere* i ruoli definiti nell'account 80398EXAMPLE:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111111111111:root"
            },
```

```
            "Action": "sts:AssumeRole"
        }
    ]
}
```

L'esempio seguente mostra una politica creata nell' AWS account *1111* che consente agli utenti di assumere il ruolo denominato `CrossAccountPipelineViewers` nell'account 80398EXAMPLE:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "sts:AssumeRole",
            "Resource": "arn:aws:iam::80398EXAMPLE:role/CrossAccountPipelineViewers"
        }
    ]
}
```

Esempio 8: utilizzo delle risorse AWS associate a un altro account in una pipeline

È possibile configurare politiche che consentano a un utente di creare una pipeline che utilizza le risorse di un altro account. AWS Ciò richiede la configurazione di policy e ruoli sia nell'account che crea la pipeline (AccountA) sia nell'account che ha creato le risorse da utilizzare nella pipeline (AccountB). È inoltre necessario creare una chiave gestita dal cliente AWS Key Management Service da utilizzare per l'accesso tra account diversi. Per ulteriori informazioni ed step-by-step esempi, consulta Crea una pipeline CodePipeline che utilizzi le risorse di un altro account AWS eConfigura la crittografia lato server per gli artefatti archiviati in Amazon S3 per CodePipeline.

Nell'esempio seguente viene illustrata una policy configurata da AccountA per un bucket S3 utilizzato per archiviare gli artefatti della pipeline. La policy concede l'accesso ad AccountB. Nel seguente esempio, l'ARN per AccountB è `012ID_ACCOUNT_B`. L'ARN per il bucket S3 è `codepipeline-us-east-2-1234567890`. Sostituire questi ARN con l'ARN del bucket S3 e dell'account cui si desidera permettere l'accesso:

```
{
    "Version": "2012-10-17",
    "Id": "SSEAndSSLPolicy",
    "Statement": [
        {
            "Sid": "DenyUnEncryptedObjectUploads",
```

```
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "aws:kms"
                }
            }
        },
        {
            "Sid": "DenyInsecureConnections",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": false
                }
            }
        },
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::012ID_ACCOUNT_B:root"
            },
            "Action": [
                "s3:Get*",
                "s3:Put*"
            ],
            "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*"
        },
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::012ID_ACCOUNT_B:root"
            },
            "Action": "s3:ListBucket",
            "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890"
        }
    ]
```

```
}
```

L'esempio seguente mostra una policy configurata da AccountA che consente ad AccountB di assumere un ruolo. Questa politica deve essere applicata al ruolo di servizio for CodePipeline (`CodePipeline_Service_Role`). Per ulteriori informazioni su come applicare le policy ai ruoli in IAM, consulta Modifying a Role. Nell'esempio seguente, `012ID_ACCOUNT_B` è l'ARN per AccountB:

```
{
    "Version": "2012-10-17",
    "Statement": {
        "Effect": "Allow",
        "Action": "sts:AssumeRole",
        "Resource": [
            "arn:aws:iam::012ID_ACCOUNT_B:role/*"
        ]
    }
}
```

L'esempio seguente mostra una politica configurata da AccountB e applicata al ruolo dell'istanza EC2 per. CodeDeploy *Questa politica consente l'accesso al bucket S3 utilizzato da AccounTA per archiviare gli artefatti della pipeline (-2-1234567890): codepipeline-us-east*

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:Get*"
            ],
            "Resource": [
                "arn:aws:s3:::codepipeline-us-east-2-1234567890/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": [
```

```
                "arn:aws:s3:::codepipeline-us-east-2-1234567890"
            ]
        }
    ]
}
```

L'esempio seguente mostra una politica relativa alla posizione AWS KMS dell'ARN della chiave gestita dal cliente creata in AccountA e configurata per consentire a AccountB di utilizzarla: ***arn:aws:kms:us-east-1:012ID_ACCOUNT_A:key/2222222-3333333-4444-556677EXAMPLE***

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:DescribeKey",
                "kms:GenerateDataKey*",
                "kms:Encrypt",
                "kms:ReEncrypt*",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-
east-1:012ID_ACCOUNT_A:key/2222222-3333333-4444-556677EXAMPLE"
            ]
        }
    ]
}
```

L'esempio seguente mostra una policy in linea per un ruolo IAM (`CrossAccount_Role`) creato da accountB che consente l'accesso CodeDeploy alle azioni richieste dalla pipeline in accounTa.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codedeploy:CreateDeployment",
                "codedeploy:GetDeployment",
```

```
                "codedeploy:GetDeploymentConfig",
                "codedeploy:GetApplicationRevision",
                "codedeploy:RegisterApplicationRevision"
            ],
            "Resource": "*"
        }
    ]
}
```

L'esempio seguente mostra una policy in linea per un ruolo IAM (`CrossAccount_Role`) creato da accountB che consente l'accesso al bucket S3 per scaricare artefatti di input e caricare artefatti di output:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:PutObject",
                "s3:PutObjectAcl"
            ],
            "Resource": [
                "arn:aws:s3:::codepipeline-us-east-2-1234567890/*"
            ]
        }
    ]
}
```

Per ulteriori informazioni su come modificare una pipeline per l'accesso tra più account alle risorse, consulta Fase 2: modifica della pipeline .

Esempi di policy basate su risorse AWS CodePipeline

Anche altri servizi, ad esempio Amazon S3, supportano policy di autorizzazioni basate su risorse. Ad esempio, è possibile associare una policy a un bucket S3 per gestire le autorizzazioni di accesso a quel bucket. Sebbene CodePipeline non supporti le politiche basate sulle risorse, memorizza gli artefatti da utilizzare nelle pipeline in bucket S3 con versione diversa.

Example Per creare una policy per un bucket S3 da utilizzare come archivio di artefatti per CodePipeline

Puoi utilizzare qualsiasi bucket S3 con versione come archivio degli artefatti per. CodePipeline Se utilizzi la procedura guidata Create Pipeline (Crea pipeline) per creare la prima pipeline, questo bucket S3 viene creato automaticamente per garantire che tutti gli oggetti caricati nello store di artefatti siano crittografati e che le connessioni al bucket siano sicure. Come best practice, se crei il tuo bucket S3, valuta se aggiungere la seguente policy o i relativi elementi al bucket. In questa policy, l'ARN per il bucket S3 è `codepipeline-us-east-2-1234567890`. Sostituisci questo ARN con l'ARN per il bucket S3:

```
{
    "Version": "2012-10-17",
    "Id": "SSEAndSSLPolicy",
    "Statement": [
        {
            "Sid": "DenyUnEncryptedObjectUploads",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:PutObject",
            "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*",
            "Condition": {
                "StringNotEquals": {
                    "s3:x-amz-server-side-encryption": "aws:kms"
                }
            }
        },
        {
            "Sid": "DenyInsecureConnections",
            "Effect": "Deny",
            "Principal": "*",
            "Action": "s3:*",
            "Resource": "arn:aws:s3:::codepipeline-us-east-2-1234567890/*",
            "Condition": {
                "Bool": {
                    "aws:SecureTransport": false
                }
            }
        }
    ]
}
```

Risoluzione dei problemi di identità e accesso in AWS CodePipeline

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare quando lavori con un IAM. CodePipeline

Argomenti

Non sono autorizzato a eseguire alcuna azione in CodePipeline

Se ti AWS Management Console dice che non sei autorizzato a eseguire un'azione, devi contattare l'amministratore per ricevere assistenza. L'amministratore è la persona che ti ha fornito il nome utente e la password.

L'errore di esempio seguente si verifica quando l'utente `mateojackson` IAM tenta di utilizzare la console per visualizzare i dettagli su una pipeline, ma non dispone delle `codepipeline:GetPipeline` autorizzazioni.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 codepipeline:GetPipeline on resource: my-pipeline
```

In questo caso, Mateo richiede al suo amministratore di aggiornare le policy per poter accedere alla risorsa `my-pipeline` utilizzando l'azione `codepipeline:GetPipeline`.

Non sono autorizzato a eseguire iam: PassRole

Se ricevi un errore che indica che non sei autorizzato a eseguire l'operazione `iam:PassRole`, devi contattare il tuo amministratore per ricevere assistenza. L'amministratore è la persona che ti ha fornito il nome utente e la password. Chiedi a quella persona di aggiornare le tue politiche per consentirti di assegnare un ruolo a CodePipeline.

Alcuni Servizi AWS consentono di trasferire un ruolo esistente a quel servizio, anziché creare un nuovo ruolo di servizio o un ruolo collegato al servizio. Per eseguire questa operazione, è necessario disporre delle autorizzazioni per trasmettere il ruolo al servizio.

Il seguente errore di esempio si verifica quando un utente IAM denominato `marymajor` tenta di utilizzare la console per eseguire un'azione in. CodePipeline Tuttavia, l'operazione richiede che il servizio disponga delle autorizzazioni concesse da un ruolo di servizio. Mary non dispone di autorizzazioni per passare il ruolo al servizio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In questo caso, Mary chiede all'amministratore di aggiornare la sue policy per poter eseguire l'operazione `iam:PassRole`.

Sono un amministratore e voglio consentire ad altri di accedere CodePipeline

Per consentire ad altri di accedere CodePipeline, devi creare un'entità IAM (utente o ruolo) per la persona o l'applicazione che necessita dell'accesso. Tale utente o applicazione utilizzerà le credenziali dell'entità per accedere ad AWS. Devi quindi allegare una policy all'entità che conceda loro le autorizzazioni corrette. CodePipeline

Per iniziare immediatamente, consulta Creazione dei primi utenti e gruppi delegati IAM nella Guida per l'utente di IAM.

Voglio consentire a persone esterne al mio AWS account di accedere alle mie risorse CodePipeline

È possibile creare un ruolo con il quale utenti in altri account o persone esterne all'organizzazione possono accedere alle tue risorse. È possibile specificare chi è attendibile per l'assunzione del ruolo. Per servizi che supportano policy basate su risorse o liste di controllo accessi (ACL), utilizza tali policy per concedere alle persone l'accesso alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Per sapere se CodePipeline supporta queste funzionalità, consultaCome AWS CodePipeline funziona con IAM.
- Per scoprire come fornire l'accesso alle tue risorse attraverso Account AWS le risorse di tua proprietà, consulta Fornire l'accesso a un utente IAM in un altro Account AWS di tua proprietà nella IAM User Guide.
- Per scoprire come fornire l'accesso alle tue risorse a terze parti Account AWS, consulta Fornire l'accesso a soggetti Account AWS di proprietà di terze parti nella Guida per l'utente IAM.

- Per informazioni su come fornire l'accesso tramite la federazione delle identità, consulta Fornire l'accesso a utenti autenticati esternamente (Federazione delle identità) nella Guida per l'utente di IAM.
- Per informazioni sulle differenze tra l'utilizzo di ruoli e policy basate su risorse per l'accesso multi-account, consulta Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente IAM.

CodePipeline riferimento alle autorizzazioni

Utilizza la tabella seguente come riferimento quando configuri il controllo degli accessi e scrivi politiche di autorizzazione da allegare a un'identità IAM (politiche basate sull'identità). La tabella elenca ogni operazione CodePipeline API e le azioni corrispondenti per le quali è possibile concedere le autorizzazioni per eseguire l'azione. Per le operazioni che supportano le autorizzazioni a livello di risorsa, la tabella elenca la AWS risorsa per la quale è possibile concedere le autorizzazioni. Puoi specificare le operazioni nel campo `Action` della policy.

Le autorizzazioni a livello di risorsa sono quelle che consentono di specificare su quali risorse gli utenti possono eseguire azioni. AWS CodePipeline fornisce un supporto parziale per le autorizzazioni a livello di risorsa. Ciò significa che per alcune chiamate AWS CodePipeline API, è possibile controllare quando gli utenti sono autorizzati a utilizzare tali azioni in base alle condizioni che devono essere soddisfatte o quali risorse gli utenti sono autorizzati a utilizzare. Ad esempio, puoi concedere agli utenti l'autorizzazione per elencare informazioni di esecuzione della pipeline, ma solo per una o più pipeline specifiche.

> **Note**
>
> Nella colonna Risorse sono elencate le risorse necessarie per le chiamate API che supportano le autorizzazioni a livello di risorsa. Per chiamate API che non supportano autorizzazioni a livello di risorsa, puoi concedere agli utenti l'autorizzazione per utilizzarla, ma devi specificare un carattere jolly (*) per l'elemento resource della dichiarazione di policy.

CodePipeline Operazioni API e autorizzazioni richieste per le azioni

AcknowledgeJob

Operazione: `codepipeline:AcknowledgeJob`

Obbligatoria per visualizzare informazioni relative a un processo specificato e se tale processo è stato ricevuto dal relativo esecutore. Utilizzata solo per operazioni personalizzate.

Risorse: supporta solo un carattere jolly (*) nell'elemento `Resource` della policy.

AcknowledgeThirdPartyJob

Operazione: `codepipeline:AcknowledgeThirdPartyJob`

Obbligatoria per confermare che un esecutore del processo ha ricevuto il processo specificato. Utilizzata solo per operazioni partner.

Risorse: supporta solo un carattere jolly (*) nell'elemento `Resource` della policy.

CreateCustomActionType

Operazione: `codepipeline:CreateCustomActionType`

Necessario per creare una nuova azione personalizzata che può essere utilizzata in tutte le pipeline associate all' AWS account. Utilizzata solo per operazioni personalizzate.

Risorse:

Tipo di operazione

`arn:aws:codepipeline:`*`region`*`:`*`account`*`:actiontype:`*`owner`*`/`*`category`*`/`*`provider`*`/`*`versio`*

CreatePipeline

Operazione: `codepipeline:CreatePipeline`

Obbligatoria per creare una pipeline.

Risorse:

Pipeline

`arn:aws:codepipeline:`*`region`*`:`*`account`*`:`*`pipeline-name`*

DeleteCustomActionType

Operazione: `codepipeline:DeleteCustomActionType`

Obbligatoria per contrassegnare un'operazione personalizzata come eliminata. `PollForJobs` per l'operazione personalizzata ha esito negativo dopo che l'operazione è stata contrassegnata per l'eliminazione. Utilizzata solo per operazioni personalizzate.

Risorse:

Tipo di operazione

`arn:aws:codepipeline:region:account:actiontype:owner/category/provider/versior`

DeletePipeline

Operazione: `codepipeline:DeletePipeline`

Obbligatoria per eliminare una pipeline.

Risorse:

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

DeleteWebhook

Operazione: `codepipeline:DeleteWebhook`

Obbligatoria per eliminare un webhook.

Risorse:

Webhook

`arn:aws:codepipeline:region:account:webhook:webhook-name`

DeregisterWebhookWithThirdParty

Operazione: `codepipeline:DeregisterWebhookWithThirdParty`

Prima di eliminare un webhook, è necessario rimuovere la connessione tra il webhook creato da CodePipeline e lo strumento esterno con gli eventi da rilevare. Attualmente supportato solo per i webhook che hanno come target un tipo di azione di. GitHub

Risorse:

Webhook

`arn:aws:codepipeline:region:account:webhook:webhook-name`

DisableStageTransition

Operazione: `codepipeline:DisableStageTransition`

Obbligatoria per evitare che gli artefatti in una pipeline eseguano la transizione alla fase successiva nella pipeline.

Risorse:

Pipeline

`arn:aws:codepipeline:`*`region`*`:`*`account`*`:`*`pipeline-name`*

EnableStageTransition

Operazione: `codepipeline:EnableStageTransition`

Obbligatoria per abilitare la transizione degli artefatti in una pipeline a una fase in una pipeline.

Risorse:

Pipeline

`arn:aws:codepipeline:`*`region`*`:`*`account`*`:`*`pipeline-name`*

GetJobDetails

Operazione: `codepipeline:GetJobDetails`

Obbligatoria per recuperare le informazioni relative a un processo. Utilizzata solo per operazioni personalizzate.

Risorse: nessuna risorsa richiesta.

GetPipeline

Operazione: `codepipeline:GetPipeline`

Obbligatoria per recuperare la struttura, le fasi, le operazioni e i metadati di una pipeline, incluso l'ARN della pipeline.

Risorse:

Pipeline

`arn:aws:codepipeline:`*`region`*`:`*`account`*`:`*`pipeline-name`*

GetPipelineExecution

Operazione: `codepipeline:GetPipelineExecution`

Obbligatoria per recuperare informazioni relative a un'esecuzione di una pipeline, inclusi i dettagli su artefatti, ID di esecuzione della pipeline, nonché nome, versione e stato della pipeline.

Risorse:

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

GetPipelineState

Operazione: `codepipeline:GetPipelineState`

Obbligatoria per recuperare le informazioni relative allo stato di una pipeline, incluse le fasi e le operazioni.

Risorse:

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

GetThirdPartyJobDetails

Operazione: `codepipeline:GetThirdPartyJobDetails`

Obbligatoria per richiedere i dettagli di un processo per un'operazione di terze parti. Utilizzata solo per operazioni partner.

Risorse: supporta solo un carattere jolly (*) nell'elemento `Resource` della policy.

ListActionTypes

Operazione: `codepipeline:ListActionTypes`

Necessario per generare un riepilogo di tutti i tipi di CodePipeline azione associati al tuo account.

Risorse:

Tipo di operazione

`arn:aws:codepipeline:region:account:actiontype:owner/category/provider/versio`

ListPipelineExecutions

Operazione: `codepipeline:ListPipelineExecutions`

Obbligatoria per generare un riepilogo delle esecuzioni più recenti di una pipeline.

Risorse:

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

ListPipelines

Operazione: `codepipeline:ListPipelines`

Obbligatoria per generare un riepilogo di tutte le pipeline associate all'account.

Risorse:

ARN della pipeline con wildcard (le autorizzazioni a livello di risorsa a livello di nome della pipeline non sono supportate)

`arn:aws:codepipeline:region:account:*`

ListTagsForResource

Operazione: `codepipeline:ListTagsForResource`

Obbligatoria per elencare i tag per una risorsa specificata.

Risorse:

Tipo di operazione

`arn:aws:codepipeline:region:account:actiontype:owner/category/provider/versior`

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

Webhook

`arn:aws:codepipeline:region:account:webhook:webhook-name`

ListWebhooks

Operazione: `codepipeline:ListWebhooks`

Obbligatoria per elencare tutti i webhook nell'account per tale regione.

Risorse:

Webhook

`arn:aws:codepipeline:region:account:webhook:webhook-name`

PollForJobs

Operazioni: `codepipeline:PollForJobs`

Necessario per recuperare informazioni su eventuali lavori su cui agire. CodePipeline

Risorse:

Tipo di operazione

`arn:aws:codepipeline:region:account:actiontype:owner/category/provider/versio`

PollForThirdPartyJobs

Operazione: `codepipeline:PollForThirdPartyJobs`

Obbligatoria per determinare se esistono processi di terze parti per un esecutore del processo su cui agire. Utilizzata solo per operazioni partner.

Risorse: supporta solo un carattere jolly (*) nell'elemento `Resource` della policy.

PutActionRevision

Operazione: `codepipeline:PutActionRevision`

Necessario per riportare informazioni CodePipeline sulle nuove revisioni a una fonte.

Risorse:

Azione

`arn:aws:codepipeline:region:account:pipeline-name/stage-name/action-name`

PutApprovalResult

Operazione: `codepipeline:PutApprovalResult`

Obbligatorio per segnalare la risposta a una richiesta di approvazione manuale a CodePipeline. Le risposte valide sono `Approved` e `Rejected`.

Risorse:

Azione

`arn:aws:codepipeline:`*`region`*`:`*`account`*`:`*`pipeline-name`*`/`*`stage-name`*`/`*`action-name`*

> **Note**
>
> Questa chiamata API supporta le autorizzazioni a livello di risorsa. Tuttavia, se utilizzi la console IAM oppure Policy Generator per creare policy con `"codepipeline:PutApprovalResult"` che specificano una risorsa ARN, si potrebbe verificare un errore. In caso di errore, puoi utilizzare la scheda JSON nella console IAM o l'interfaccia a riga di comando per creare una policy.

PutJobFailureResult

Operazione: `codepipeline:PutJobFailureResult`

Obbligatoria per segnalare la mancata esecuzione di un processo come restituito alla pipeline da un esecutore del processo. Utilizzata solo per operazioni personalizzate.

Risorse: supporta solo un carattere jolly (*) nell'elemento `Resource` della policy.

PutJobSuccessResult

Operazione: `codepipeline:PutJobSuccessResult`

Obbligatoria per segnalare il completamento di un processo come restituito alla pipeline da un esecutore del processo. Utilizzata solo per operazioni personalizzate.

Risorse: supporta solo un carattere jolly (*) nell'elemento `Resource` della policy.

PutThirdPartyJobFailureResult

Operazione: `codepipeline:PutThirdPartyJobFailureResult`

Obbligatoria per segnalare l'errore in un processo di terze parti come restituito alla pipeline da un esecutore del processo. Utilizzata solo per operazioni partner.

Risorse: supporta solo un carattere jolly (*) nell'elemento `Resource` della policy.

PutThirdPartyJobSuccessResult

Operazione: `codepipeline:PutThirdPartyJobSuccessResult`

Obbligatoria per segnalare il completamento di un processo di terze parti come restituito alla pipeline da un esecutore del processo. Utilizzata solo per operazioni partner.

Risorse: supporta solo un carattere jolly (*) nell'elemento `Resource` della policy.

PutWebhook

Operazione: `codepipeline:PutWebhook`

Obbligatoria per creare un webhook.

Risorse:

Webhook

`arn:aws:codepipeline:`*`region`*`:`*`account`*`:webhook:`*`webhook-name`*

RegisterWebhookWithThirdParty

Operazione: `codepipeline:RegisterWebhookWithThirdParty`

Risorse:

Dopo che un webhook è stato creato, obbligatoria per configurare terze parti supportate per chiamare l'URL del webhook generato.

Webhook

`arn:aws:codepipeline:`*`region`*`:`*`account`*`:webhook:`*`webhook-name`*

RetryStageExecution

Operazione: `codepipeline:RetryStageExecution`

Obbligatoria per riprendere l'esecuzione della pipeline riprovando le ultime operazioni non riuscite in una fase.

Risorse:

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

StartPipelineExecution

Operazione: `codepipeline:StartPipelineExecution`

Obbligatoria per avviare la pipeline specificata (in particolare, per avviare l'elaborazione del commit più recente al percorso di origine specificato come parte della pipeline).

Risorse:

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

TagResource

Operazione: `codepipeline:TagResource`

Obbligatoria per aggiungere tag alla risorsa specificata.

Risorse:

Tipo di operazione

`arn:aws:codepipeline:region:account:actiontype:owner/category/provider/version`

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

Webhook

`arn:aws:codepipeline:region:account:webhook:webhook-name`

UntagResource

Operazione: `codepipeline:UntagResource`

Obbligatoria per aggiungere tag alla risorsa specificata.

Risorse:

Tipo di operazione

`arn:aws:codepipeline:region:account:actiontype:owner/category/provider/version`

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

Webhook

`arn:aws:codepipeline:region:account:webhook:webhook-name`

UpdatePipeline

Operazione: `codepipeline:UpdatePipeline`

Obbligatoria per aggiornare una pipeline specificata con modifiche alla relativa struttura.

Risorse:

Pipeline

`arn:aws:codepipeline:region:account:pipeline-name`

Gestisci il ruolo CodePipeline del servizio

Il ruolo CodePipeline di servizio è configurato con una o più politiche che controllano l'accesso alle AWS risorse utilizzate dalla pipeline. Potresti voler allegare più politiche a questo ruolo, modificare la politica associata al ruolo o configurare politiche per altri ruoli di servizio in AWS. Potrebbe inoltre essere necessario collegare una policy a un ruolo durante la configurazione dell'accesso tra più account alla pipeline.

> **Important**
>
> Modificando una dichiarazione di policy o collegando un'altra policy al ruolo può impedire il funzionamento delle pipeline. Assicurati di comprendere le implicazioni prima di modificare CodePipeline in qualsiasi modo il ruolo di servizio. Assicurati di testare le pipeline dopo aver apportato eventuali modifiche al ruolo del servizio.

> **Note**
>
> Nella console, i ruoli del servizio creati prima di settembre 2018 vengono creati con il nome `oneClick_AWS-CodePipeline-Service_`*`ID-Number`*.
> I ruoli del servizio creati dopo settembre 2018 utilizzano il formato del nome del ruolo del servizio `AWSCodePipelineServiceRole-`*`Region`*`-`*`Pipeline_Name`*. Ad esempio, per una pipeline denominata `MyFirstPipeline` in`eu-west-2`, la console assegna un nome al ruolo e alla policy`AWSCodePipelineServiceRole-eu-west-2-MyFirstPipeline`.

Rimozione delle autorizzazioni dal ruolo di servizio CodePipeline

Puoi modificare la dichiarazione del ruolo del servizio per rimuovere l'accesso alle risorse non utilizzate. Ad esempio, se nessuna delle tue pipeline include Elastic Beanstalk, puoi modificare l'informativa per rimuovere la sezione che concede l'accesso alle risorse Elastic Beanstalk.

Allo stesso modo, se nessuna delle tue pipeline include CodeDeploy, puoi modificare l'informativa per rimuovere la sezione che concede l'accesso alle risorse: CodeDeploy

```
    {
    "Action": [
        "codedeploy:CreateDeployment",
        "codedeploy:GetApplicationRevision",
        "codedeploy:GetDeployment",
        "codedeploy:GetDeploymentConfig",
        "codedeploy:RegisterApplicationRevision"
    ],
    "Resource": "*",
    "Effect": "Allow"
},
```

Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline

Prima di poterla utilizzare nelle pipeline, è necessario aggiornare la dichiarazione sulla politica relativa al ruolo di servizio con le autorizzazioni relative a una dichiarazione Servizio AWS non ancora inclusa nell'informativa sulla politica del ruolo di servizio predefinita.

Ciò è particolarmente importante se il ruolo di servizio che utilizzi per le tue pipeline è stato creato prima dell'aggiunta del supporto per un CodePipeline . Servizio AWS

La tabella seguente mostra quando è stato aggiunto il supporto per altri Servizi AWS.

Servizio AWS	CodePipeline data di supporto
AWS CloudFormation StackSets azioni	30 dicembre 2020
CodeCommit formato di artefatto di output clone completo	11 novembre 2020
CodeBuild build in batch	30 luglio 2020
AWS AppConfig	22 giugno 2020
AWS Step Functions	27 maggio 2020
AWS CodeStar Connessioni	18 dicembre 2019
L'operazione `CodeDeployToECS`	27 novembre 2018
Amazon ECR	27 novembre 2018
Catalogo dei servizi	16 ottobre 2018
AWS Device Farm	19 luglio 2018
Amazon ECS	12 dicembre 2017 /Aggiornamento per l'attivazione dell'autorizzazione all'aggiunta di tag il 21 luglio 2017
CodeCommit	18 Aprile 2016
AWS OpsWorks	2 giugno 2016
AWS CloudFormation	3 Novembre 2016
AWS CodeBuild	1° dicembre 2016
Elastic Beanstalk	Avvio iniziale del servizio

Per aggiungere le autorizzazioni per un servizio supportato, procedi nel seguente modo:

1. Accedi AWS Management Console e apri la console IAM all'indirizzo https://console.aws.amazon.com/iam/.
2. Nella console IAM, nel riquadro di navigazione, scegli Ruoli, quindi scegli il tuo `AWS-CodePipeline-Service` ruolo dall'elenco dei ruoli.
3. Nella scheda Autorizzazioni, in Politiche in linea, nella riga relativa alla politica del ruolo di servizio, scegli Modifica politica.
4. Aggiungi le autorizzazioni richieste nella casella del documento Policy.

> **Note**
>
> Quando crei policy IAM, segui i consigli di sicurezza standard che prevedono la concessione del privilegio minimo, ovvero la concessione solo delle autorizzazioni necessarie per eseguire un'attività. Alcune chiamate API supportano autorizzazioni basate su risorse e permettono la limitazione degli accessi. Ad esempio, in questo caso, per limitare le autorizzazioni quando si chiamano le operazioni `DescribeTasks` e `ListTasks`, puoi sostituire il carattere jolly (*) con un ARN della risorsa o con ARN della risorsa che contiene un carattere jolly (*). Per ulteriori informazioni sulla creazione di una policy che garantisca l'accesso con privilegi minimi, consulta. https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#grant-least-privilege

Ad esempio, per ricevere CodeCommit assistenza, aggiungi quanto segue alla tua dichiarazione sulla politica:

```
{
  "Effect": "Allow",
  "Action": [
      "codecommit:GetBranch",
      "codecommit:GetCommit",
      "codecommit:UploadArchive",
      "codecommit:GetUploadArchiveStatus",
      "codecommit:CancelUploadArchive"
           ],
  "Resource": "resource_ARN"
},
```

Per AWS OpsWorks ricevere assistenza, aggiungi quanto segue alla tua dichiarazione sulla politica:

```
{
    "Effect": "Allow",
    "Action": [
        "opsworks:CreateDeployment",
        "opsworks:DescribeApps",
        "opsworks:DescribeCommands",
        "opsworks:DescribeDeployments",
        "opsworks:DescribeInstances",
        "opsworks:DescribeStacks",
        "opsworks:UpdateApp",
        "opsworks:UpdateStack"
    ],
    "Resource": "resource_ARN"
},
```

Per AWS CloudFormation ricevere assistenza, aggiungi quanto segue alla tua dichiarazione sulla politica:

```
{
    "Effect": "Allow",
    "Action": [
        "cloudformation:CreateStack",
        "cloudformation:DeleteStack",
        "cloudformation:DescribeStackEvents",
        "cloudformation:DescribeStacks",
        "cloudformation:UpdateStack",
        "cloudformation:CreateChangeSet",
        "cloudformation:DeleteChangeSet",
        "cloudformation:DescribeChangeSet",
        "cloudformation:ExecuteChangeSet",
        "cloudformation:SetStackPolicy",
        "cloudformation:ValidateTemplate",
        "iam:PassRole"
    ],
    "Resource": "resource_ARN"
},
```

Tieni presente che l'`cloudformation:DescribeStackEvents`autorizzazione è facoltativa. Consente all' AWS CloudFormation azione di mostrare un messaggio di errore più dettagliato. Questa autorizzazione può essere revocata dal ruolo IAM se non desideri che i dettagli delle

risorse vengano visualizzati nei messaggi di errore della pipeline. Per ulteriori informazioni, consulta AWS CloudFormation.

Per ricevere CodeBuild assistenza, aggiungi quanto segue alla tua dichiarazione sulla politica:

```
{
    "Effect": "Allow",
    "Action": [
        "codebuild:BatchGetBuilds",
        "codebuild:StartBuild"
    ],
    "Resource": "resource_ARN"
},
```

> **Note**
>
> Il supporto per le build in batch è stato aggiunto in un secondo momento. Vedi il passaggio 11 per le autorizzazioni da aggiungere al ruolo di servizio per le build in batch.

Per ricevere AWS Device Farm assistenza, aggiungi quanto segue alla tua dichiarazione politica:

```
{
    "Effect": "Allow",
    "Action": [
        "devicefarm:ListProjects",
        "devicefarm:ListDevicePools",
        "devicefarm:GetRun",
        "devicefarm:GetUpload",
        "devicefarm:CreateUpload",
        "devicefarm:ScheduleRun"
    ],
    "Resource": "resource_ARN"
},
```

Per il supporto di Service Catalog, aggiungi quanto segue alla tua informativa sulla politica:

```
{
    "Effect": "Allow",
    "Action": [
        "servicecatalog:ListProvisioningArtifacts",
```

```
            "servicecatalog:CreateProvisioningArtifact",
            "servicecatalog:DescribeProvisioningArtifact",
            "servicecatalog:DeleteProvisioningArtifact",
            "servicecatalog:UpdateProduct"
        ],
        "Resource": "resource_ARN"
    },
    {
        "Effect": "Allow",
        "Action": [
            "cloudformation:ValidateTemplate"
        ],
        "Resource": "resource_ARN"
    }
```

5. Per il supporto di Amazon ECR, aggiungi quanto segue alla tua dichiarazione sulla politica:

```
    {
        "Effect": "Allow",
        "Action": [
            "ecr:DescribeImages"
        ],
        "Resource": "resource_ARN"
    },
```

6. Per Amazon ECS, le seguenti sono le autorizzazioni minime necessarie per creare pipeline con un'azione di distribuzione di Amazon ECS.

```
    {
        "Effect": "Allow",
        "Action": [
            "ecs:DescribeServices",
            "ecs:DescribeTaskDefinition",
            "ecs:DescribeTasks",
            "ecs:ListTasks",
            "ecs:RegisterTaskDefinition",
            "ecs:TagResource",
            "ecs:UpdateService"
        ],
        "Resource": "resource_ARN"
    },
```

Puoi scegliere di utilizzare l'autorizzazione all'etichettatura in Amazon ECS. Iscrivendoti, devi concedere le seguenti autorizzazioni:. `ecs:TagResource` Per ulteriori informazioni su come attivare e determinare se l'autorizzazione è richiesta e l'autorizzazione dei tag è applicata, consulta la cronologia dell'autorizzazione all'etichettatura nella Amazon Elastic Container Service Developer Guide.

È inoltre necessario aggiungere le `iam:PassRole` autorizzazioni per utilizzare i ruoli IAM per le attività. Per ulteriori informazioni, consulta il ruolo IAM di esecuzione delle attività di Amazon ECS e IAM Roles for Tasks. Utilizza il seguente testo di policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::aws_account_ID:role/ecsTaskExecutionRole_or_TaskRole_name"
            ]
        }
    ]
}
```

7. Per l'`CodeDeployToECS`azione (distribuzioni blu/verdi), le seguenti sono le autorizzazioni minime necessarie per creare pipeline con un'azione di distribuzione blu/verde CodeDeploy verso Amazon ECS.

```
{
    "Effect": "Allow",
    "Action": [
        "codedeploy:CreateDeployment",
        "codedeploy:GetDeployment",
        "codedeploy:GetApplication",
        "codedeploy:GetApplicationRevision",
        "codedeploy:RegisterApplicationRevision",
        "codedeploy:GetDeploymentConfig",
        "ecs:RegisterTaskDefinition",
        "ecs:TagResource"
    ],
    "Resource": "resource_ARN"
```

```
},
```

Puoi scegliere di utilizzare l'autorizzazione all'etichettatura in Amazon ECS. Iscrivendoti, devi concedere le seguenti autorizzazioni:. `ecs:TagResource` Per ulteriori informazioni su come attivare e determinare se l'autorizzazione è richiesta e l'autorizzazione dei tag è applicata, consulta la cronologia dell'autorizzazione all'etichettatura nella Amazon Elastic Container Service Developer Guide.

È inoltre necessario aggiungere le `iam:PassRole` autorizzazioni per utilizzare i ruoli IAM per le attività. Per ulteriori informazioni, consulta il ruolo IAM di esecuzione delle attività di Amazon ECS e IAM Roles for Tasks. Utilizza il seguente testo di policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::aws_account_ID:role/ecsTaskExecutionRole_or_TaskRole_name"
            ]
        }
    ]
}
```

È inoltre possibile aggiungere `ecs-tasks.amazonaws.com` all'elenco dei servizi in base alla `iam:PassedToService` condizione, come mostrato in questo esempio.

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "iam:PassRole"
            ],
            "Resource": "resource_ARN",
            "Condition": {
                "StringEqualsIfExists": {
                    "iam:PassedToService": [
                        "cloudformation.amazonaws.com",
```

```
                        "elasticbeanstalk.amazonaws.com",
                        "ec2.amazonaws.com",
                        "ecs-tasks.amazonaws.com"
                    ]
                }
            }
        },
```

8. Per AWS CodeStar le connessioni, è richiesta la seguente autorizzazione per creare pipeline con una fonte che utilizza una connessione, come Bitbucket Cloud.

```
{
    "Effect": "Allow",
    "Action": [
        "codestar-connections:UseConnection"
    ],
    "Resource": "resource_ARN"
},
```

   Per ulteriori informazioni sulle autorizzazioni IAM per le connessioni, consulta Connections permissions reference.

9. Per l'`StepFunctions`azione, le seguenti sono le autorizzazioni minime necessarie per creare pipeline con un'azione di invoca Step Functions.

```
{
    "Effect": "Allow",
    "Action": [
        "states:DescribeStateMachine",
        "states:DescribeExecution",
        "states:StartExecution"
    ],
    "Resource": "resource_ARN"
},
```

10. Per l'`AppConfig`azione, le seguenti sono le autorizzazioni minime necessarie per creare pipeline con un'azione di invoca. AWS AppConfig

```
{
    "Effect": "Allow",
    "Action": [
        "appconfig:StartDeployment",
```

```
        "appconfig:GetDeployment",
        "appconfig:StopDeployment"
    ],
    "Resource": "resource_ARN"
},
```

11 Per il CodeBuild supporto per le build in batch, aggiungi quanto segue alla tua dichiarazione di politica:

```
{
    "Effect": "Allow",
    "Action": [
        "codebuild:BatchGetBuildBatches",
        "codebuild:StartBuildBatch"
    ],
    "Resource": "resource_ARN"
},
```

12 Per AWS CloudFormation StackSets le azioni, sono richieste le seguenti autorizzazioni minime.

- Per l'`CloudFormationStackSet`azione, aggiungi quanto segue alla tua dichiarazione politica:

```
{
    "Effect": "Allow",
    "Action": [
        "cloudformation:CreateStackSet",
        "cloudformation:UpdateStackSet",
        "cloudformation:CreateStackInstances",
        "cloudformation:DescribeStackSetOperation",
        "cloudformation:DescribeStackSet",
        "cloudformation:ListStackInstances"
    ],
    "Resource": "resource_ARN"
},
```

- Per l'`CloudFormationStackInstances`azione, aggiungi quanto segue alla tua dichiarazione politica:

```
{
    "Effect": "Allow",
    "Action": [
        "cloudformation:CreateStackInstances",
        "cloudformation:DescribeStackSetOperation"
```

```
        ],
        "Resource": "resource_ARN"
    },
```

13 Per CodeCommit supportare l'opzione di clonazione completa, aggiungi quanto segue alla tua dichiarazione politica:

```
{
    "Effect": "Allow",
    "Action": [
        "codecommit:GetRepository"
    ],
    "Resource": "resource_ARN"
},
```

> **Note**
>
> Per assicurarti che la tua CodeBuild azione possa utilizzare l'opzione full clone con una CodeCommit fonte, devi anche aggiungere l'`codecommit:GitPull`autorizzazione alla dichiarazione politica per il ruolo di CodeBuild servizio del tuo progetto.

14 Per Elastic Beanstalk, le seguenti sono le autorizzazioni minime necessarie per creare pipeline con un'azione di distribuzione. `ElasticBeanstalk`

```
{
    "Effect": "Allow",
    "Action": [
        "elasticbeanstalk:*",
        "ec2:*",
        "elasticloadbalancing:*",
        "autoscaling:*",
        "cloudwatch:*",
        "s3:*",
        "sns:*",
        "cloudformation:*",
        "rds:*",
        "sqs:*",
        "ecs:*"
    ],
    "Resource": "resource_ARN"
},
```

> **Note**
>
> È necessario sostituire i caratteri jolly nella politica delle risorse con le risorse dell'account a cui si desidera limitare l'accesso. Per ulteriori informazioni sulla creazione di una politica che garantisca l'accesso con privilegi minimi, consulta. https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#grant-least-privilege

15. Per una pipeline che desideri configurare per CloudWatch Logs, di seguito sono riportate le autorizzazioni minime che devi aggiungere al ruolo di servizio. CodePipeline

```
{
    "Effect": "Allow",
    "Action": [
        "logs:DescribeLogGroups",
        "logs:PutRetentionPolicy"
    ],
    "Resource": "resource_ARN"
},
```

> **Note**
>
> È necessario sostituire i caratteri jolly nella politica delle risorse con le risorse dell'account a cui si desidera limitare l'accesso. Per ulteriori informazioni sulla creazione di una politica che garantisca l'accesso con privilegi minimi, consulta. https://docs.aws.amazon.com/IAM/latest/UserGuide/best-practices.html#grant-least-privilege

16. Scegliere Review policy (Esamina policy) per accertarsi che la policy non contenga errori. Quando la politica è priva di errori, scegli Applica politica.

Registrazione e monitoraggio CodePipeline

Puoi utilizzare le funzionalità di registrazione AWS per determinare le azioni intraprese dagli utenti nel tuo account e le risorse utilizzate. I file di log visualizzano:

- La data e l'ora delle operazioni.
- L'indirizzo IP di origine di un'operazione.
- Quali operazioni non sono riuscite a causa di autorizzazioni inadeguate.

Le funzionalità di registrazione sono disponibili nelle seguenti versioni: Servizi AWS

- AWS CloudTrail può essere utilizzato per registrare le chiamate AWS API e gli eventi correlati effettuati da o per conto di un Account AWS. Per ulteriori informazioni, consulta Registrazione delle chiamate CodePipeline API con AWS CloudTrail.
- Amazon CloudWatch Events può essere usato per monitorare Cloud AWS le tue risorse e le applicazioni su cui esegui AWS. Puoi creare avvisi in Amazon CloudWatch Events in base a metriche da te definite. Per ulteriori informazioni, consulta Monitoraggio CodePipeline degli eventi.

Convalida della conformità per AWS CodePipeline

Per sapere se un Servizio AWS programma rientra nell'ambito di specifici programmi di conformità, consulta Servizi AWS la sezione Scope by Compliance Program Servizi AWS e scegli il programma di conformità che ti interessa. Per informazioni generali, consulta Programmi di AWS conformità Programmi di di .

È possibile scaricare report di audit di terze parti utilizzando AWS Artifact. Per ulteriori informazioni, consulta Scaricamento dei report in AWS Artifact .

La vostra responsabilità di conformità durante l'utilizzo Servizi AWS è determinata dalla sensibilità dei dati, dagli obiettivi di conformità dell'azienda e dalle leggi e dai regolamenti applicabili. AWS fornisce le seguenti risorse per contribuire alla conformità:

- Guide introduttive su sicurezza e conformità: queste guide all'implementazione illustrano considerazioni sull'architettura e forniscono passaggi per implementare ambienti di base incentrati sulla AWS sicurezza e la conformità.
- Progettazione per la sicurezza e la conformità HIPAA su Amazon Web Services: questo white paper descrive in che modo le aziende possono utilizzare AWS per creare applicazioni idonee all'HIPAA.

> **Note**
>
> Non Servizi AWS tutte sono idonee all'HIPAA. Per ulteriori informazioni, consulta la sezione Riferimenti sui servizi conformi ai requisiti HIPAA.

- AWS Risorse per la per la conformità: questa raccolta di cartelle di lavoro e guide potrebbe essere valida per il tuo settore e la tua località.

- AWS Guide alla conformità dei clienti: comprendi il modello di responsabilità condivisa attraverso la lente della conformità. Le guide riassumono le migliori pratiche per la protezione Servizi AWS e mappano le linee guida per i controlli di sicurezza su più framework (tra cui il National Institute of Standards and Technology (NIST), il Payment Card Industry Security Standards Council (PCI) e l'International Organization for Standardization (ISO)).
- Evaluating Resources with Rules nella AWS Config Developer Guide: il AWS Config servizio valuta la conformità delle configurazioni delle risorse alle pratiche interne, alle linee guida e alle normative del settore.
- AWS Security Hub— Ciò Servizio AWS fornisce una visione completa dello stato di sicurezza interno. AWS La Centrale di sicurezza utilizza i controlli di sicurezza per valutare le risorse AWS e verificare la conformità agli standard e alle best practice del settore della sicurezza. Per un elenco dei servizi e dei controlli supportati, consulta la pagina Documentazione di riferimento sui controlli della Centrale di sicurezza.
- Amazon GuardDuty: Servizio AWS rileva potenziali minacce ai tuoi carichi di lavoro Account AWS, ai contenitori e ai dati monitorando l'ambiente alla ricerca di attività sospette e dannose. GuardDuty può aiutarti a soddisfare vari requisiti di conformità, come lo standard PCI DSS, soddisfacendo i requisiti di rilevamento delle intrusioni imposti da determinati framework di conformità.
- AWS Audit Manager— Ciò Servizio AWS consente di verificare continuamente l' AWS utilizzo per semplificare la gestione del rischio e la conformità alle normative e agli standard di settore.

Resilienza in AWS CodePipeline

L'infrastruttura AWS globale è costruita attorno a AWS regioni e zone di disponibilità. AWS Le regioni forniscono più zone di disponibilità fisicamente separate e isolate, collegate con reti a bassa latenza, ad alto throughput e altamente ridondanti. Con le zone di disponibilità, puoi progettare e gestire applicazioni e database che eseguono automaticamente il failover tra zone di disponibilità senza interruzioni. Le zone di disponibilità sono più disponibili, tolleranti ai guasti e scalabili rispetto alle infrastrutture a data center singolo o multiplo tradizionali.

Per ulteriori informazioni su AWS regioni e zone di disponibilità, consulta Global Infrastructure.AWS

Sicurezza dell'infrastruttura in AWS CodePipeline

In quanto servizio gestito, AWS CodePipeline è protetto dalla sicurezza di rete AWS globale. Per informazioni sui servizi AWS di sicurezza e su come AWS protegge l'infrastruttura, consulta AWS

Cloud Security. Per progettare il tuo AWS ambiente utilizzando le migliori pratiche per la sicurezza dell'infrastruttura, vedi Infrastructure Protection in Security Pillar AWS Well-Architected Framework.

Utilizzate chiamate API AWS pubblicate per accedere CodePipeline attraverso la rete. I client devono supportare quanto segue:

- Transport Layer Security (TLS). È richiesto TLS 1.2 ed è consigliato TLS 1.3.
- Suite di cifratura con Perfect Forward Secrecy (PFS), ad esempio Ephemeral Diffie-Hellman (DHE) o Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La maggior parte dei sistemi moderni, come Java 7 e versioni successive, supporta tali modalità.

Inoltre, le richieste devono essere firmate utilizzando un ID chiave di accesso e una chiave di accesso segreta associata a un principale IAM. O puoi utilizzare AWS Security Token Service (AWS STS) per generare credenziali di sicurezza temporanee per sottoscrivere le richieste.

Best practice di sicurezza

CodePipeline fornisce una serie di funzionalità di sicurezza da considerare durante lo sviluppo e l'implementazione delle proprie politiche di sicurezza. Le seguenti best practice sono linee guida generali e non rappresentano una soluzione di sicurezza completa. Poiché queste best practice potrebbero non essere appropriate o sufficienti per l'ambiente, gestiscile come considerazioni utili anziché prescrizioni.

Utilizza la crittografia e l'autenticazione per i repository di origine che si connettono alle pipeline. Queste sono le CodePipeline migliori pratiche per la sicurezza:

- Se create una configurazione di pipeline o di azione che deve includere segreti, come token o password, non inserite i segreti direttamente nella configurazione dell'azione o i valori predefiniti delle variabili definite a livello di pipeline o di AWS CloudFormation configurazione, poiché le informazioni verranno visualizzate nei log. Utilizzare Secrets Manager per impostare e archiviare i segreti, quindi utilizzare il segreto di riferimento nella configurazione della pipeline e dell'azione, come descritto in. Utilizzalo per tenere traccia delle password del database o delle chiavi AWS Secrets Manager API di terze parti
- Se crei una pipeline che utilizza un bucket di origine S3, configura la crittografia lato server per gli artefatti archiviati in Amazon S3 per la gestione, come descritto CodePipeline in. AWS KMS keysConfigura la crittografia lato server per gli artefatti archiviati in Amazon S3 per CodePipeline

- Se utilizzi un provider di operazioni Jenkins, quando utilizzi un provider di compilazione Jenkins per l'operazione di compilazione e di test della pipeline, installa Jenkins su un'istanza EC2 e configura un profilo dell'istanza EC2 separato. Assicurati che il profilo dell'istanza conceda a Jenkins solo le AWS autorizzazioni necessarie per eseguire attività per il tuo progetto, come il recupero di file da Amazon S3. Per ulteriori informazioni su come creare il ruolo per il profilo dell'istanza Jenkins, consulta le fasi descritte in Crea un ruolo IAM da utilizzare per l'integrazione con Jenkins.

AWS CodePipeline riferimento alla riga di comando

Utilizzate questo riferimento quando lavorate con i AWS CodePipeline comandi e come supplemento alle informazioni documentate nella Guida per l'AWS CLI utente e nella Guida di AWS CLI riferimento.

Prima di utilizzare il AWS CLI, assicuratevi di completare i prerequisiti in. Guida introduttiva con CodePipeline

Per visualizzare un elenco di tutti i CodePipeline comandi disponibili, esegui il comando seguente:

```
aws codepipeline help
```

Per visualizzare informazioni su un CodePipeline comando specifico, esegui il comando seguente, dove *command-name* è il nome di uno dei comandi elencati di seguito (ad esempio,create-pipeline):

```
aws codepipeline command-name help
```

Per iniziare a imparare a usare i comandi dell' CodePipeline estensione di AWS CLI, consulta una o più delle seguenti sezioni:

- Creazione di un'operazione personalizzata
- Creazione di una pipeline (CLI)
- Eliminazione di una pipeline (CLI)
- Disabilitazione o abilitazione delle transizioni (CLI)
- Visualizzazione dei dettagli e della cronologia della pipeline (CLI)
- Nuovo tentativo di operazioni non riuscite (CLI)
- Avvio manuale di una pipeline (CLI)
- Modifica di una pipeline (AWS CLI)

Puoi anche visualizzare esempi di come utilizzare la maggior parte di questi comandi in CodePipeline tutorial.

CodePipeline riferimento alla struttura della tubazione

Per impostazione predefinita, qualsiasi pipeline creata con successo AWS CodePipeline ha una struttura valida. Tuttavia, se crei o modifichi manualmente un file JSON per creare una pipeline o aggiorni una pipeline da AWS CLI, potresti creare inavvertitamente una struttura non valida. Il seguente riferimento può aiutarti a comprendere meglio i requisiti della struttura della pipeline e la risoluzione dei problemi. Consulta i vincoli in Quote in AWS CodePipeline, che si applicano a tutte le pipeline.

Argomenti

Tipi di azioni e provider validi in CodePipeline

Il formato della struttura della pipeline viene utilizzato per creare operazioni e fasi in una pipeline. Un tipo di operazione è costituito da una categoria di operazione e un tipo di provider.

Di seguito sono elencate le categorie di azioni valide in CodePipeline:

- Origine
- Creazione
- Test
- Implementazione
- Approval
- Invoke

Ogni categoria di operazione dispone di una serie definita di provider. Ogni provider di azioni, ad esempio Amazon S3, ha un nome di provider, ad esempio`S3`, che deve essere utilizzato nel `Provider` campo della categoria di azioni nella struttura della pipeline.

Esistono tre valori validi per il campo `Owner` nella sezione categoria di azioni nella struttura della pipeline: `AWS`, `ThirdParty` e `Custom`.

Per trovare il nome del provider e le informazioni sul proprietario del provider di azioni, vedere Riferimento per la struttura delle operazioni o Numero di artefatti di input e output per ogni tipo di operazione.

La seguente tabella elenca i provider validi per tipo di operazione.

> **Note**
>
> Per le azioni Bitbucket Cloud GitHub, GitHub Enterprise Server GitLab o.com, fai riferimento all'argomento di riferimento sull'CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestiteazione.

Provider validi per tipo di operazione

Categoria di operazione	Provider di operazione validi	Riferimento per l'operazione
Origine	Amazon S3	Azione all'origine di Amazon S3
	Amazon ECR	Amazon ECR
	CodeCommit	CodeCommit
	CodeStarSourceConnection (per le azioni Bitbucket Cloud GitHub, GitHub Enterprise Server o .com) GitLab	CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite
Creazione	CodeBuild	AWS CodeBuild
	Personalizzato CloudBees	Numero di artefatti di input e output per ogni

Categoria di operazione	Provider di operazione validi	Riferimento per l'operazione
		tipo di operazione
	Jenkins personalizzato	Numero di artefatti di input e output per ogni tipo di operazione
	Personalizzato TeamCity	Numero di artefatti di input e output per ogni tipo di operazione
Test	CodeBuild	AWS CodeBuild
	AWS Device Farm	Numero di artefatti di input e output per ogni tipo di operazione
	ThirdParty GhostInspector	Numero di artefatti di input e output per ogni tipo di operazione
	Jenkins personalizzato	Numero di artefatti di input e output per ogni tipo di operazione

Categoria di operazione	Provider di operazione validi	Riferimento per l'operazione
	ThirdParty StormRunner Carico Micro Focus	Numero di artefatti di input e output per ogni tipo di operazione
	ThirdParty Nuova vola	Numero di artefatti di input e output per ogni tipo di operazione
Implementazione	Amazon S3	Azione di distribuzione di Amazon S3
	AWS CloudFormation	AWS CloudFormation
	AWS CloudFormation StackSets (include le `CloudFormationStackInstances` azioni `CloudFormationStackSet` e)	AWS CloudFormation StackSets
	CodeDeploy	Numero di artefatti di input e output per ogni tipo di operazione
	Amazon ECS	Numero di artefatti di input e output per ogni tipo di operazione

Categoria di operazione	Provider di operazione validi	Riferimento per l'operazione
	Amazon ECS (Blu/Verde) (questa è l'operazione `CodeDeployToECS`)	Numero di artefatti di input e output per ogni tipo di operazione
	Elastic Beanstalk	Numero di artefatti di input e output per ogni tipo di operazione
	AWS AppConfig	AWS AppConfig
	AWS OpsWorks	Numero di artefatti di input e output per ogni tipo di operazione
	Catalogo dei servizi	Numero di artefatti di input e output per ogni tipo di operazione
	Amazon Alexa	Numero di artefatti di input e output per ogni tipo di operazione

Categoria di operazione	Provider di operazione validi	Riferimento per l'operazione
	Personalizzato XebiaLabs	Numero di artefatti di input e output per ogni tipo di operazione
Approval	Manuale	Numero di artefatti di input e output per ogni tipo di operazione
Invoke	AWS Lambda	AWS Lambda
	AWS Step Functions	AWS Step Functions

Alcuni tipi di azione CodePipeline sono disponibili solo in alcune AWS regioni. È possibile che un tipo di azione sia disponibile in una AWS regione, ma non è disponibile un AWS provider per quel tipo di azione.

Per ulteriori informazioni su ogni provider di operazione, consulta Integrazioni con tipi di CodePipeline azioni.

Le seguenti sezioni forniscono esempi di informazioni sui provider e sulle proprietà di configurazione per ogni tipo di operazione.

Requisiti della pipeline e della struttura dello stadio in CodePipeline

Una pipeline a due fasi ha la seguente struttura di base:

```
{
    "roleArn": "An IAM ARN for a service role, such as arn:aws:iam::80398EXAMPLE:role/
CodePipeline_Service_Role",
    "stages": [
        {
```

```
            "name": "SourceStageName",
            "actions": [
            ... See Requisiti della struttura delle azioni in  CodePipeline ...
            ]
        },
        {
            "name": "NextStageName",
            "actions": [
            ... See Requisiti della struttura delle azioni in  CodePipeline ...
            ]
        }
    ],
    "artifactStore": {
        "type": "S3",
        "location": "The name of the Amazon S3 bucket automatically generated for you
 the first time you create a pipeline
            using the console, such as codepipeline-us-east-2-1234567890, or any Amazon
 S3 bucket you provision for this purpose"
    },
    "name": "YourPipelineName",
    "version": 1
}
```

La struttura della pipeline ha i seguenti requisiti:

- Una pipeline deve contenere almeno due fasi.
- La prima fase di una pipeline deve contenere almeno un'operazione di origine. Può contenere solo operazioni di origine.
- Solo la prima fase di una pipeline può contenere operazioni di origine.
- Almeno una fase in ogni pipeline deve contenere un'operazione diversa da un'operazione di origine.
- Tutti i nomi delle fasi all'interno di una pipeline devono essere univoci.
- I nomi delle fasi non possono essere modificati nella CodePipeline console. Se modificate il nome di una fase utilizzando il AWS CLI e lo stage contiene un'azione con uno o più parametri segreti (come un token OAuth), il valore di tali parametri segreti non viene mantenuto. È necessario inserire manualmente il valore dei parametri (che sono mascherati da quattro asterischi nel codice JSON restituito da AWS CLI) e includerli nella struttura JSON.
- Il `artifactStore` campo contiene il tipo e la posizione del bucket di artefatti per una pipeline con tutte le azioni nella stessa regione. AWS Se aggiungi azioni in una regione diversa dalla pipeline,

la `artifactStores` mappatura viene utilizzata per elencare il bucket di artefatti per ogni regione in cui vengono eseguite le azioni. AWS Quando crei o modifichi una pipeline, devi disporre di un bucket di artefatti nella regione della pipeline e di un bucket di artefatti per ogni regione in cui prevedi di eseguire un'operazione.

L'esempio seguente mostra la struttura di base per una pipeline con operazioni tra più regioni che usa il parametro `artifactStores`:

```
    "pipeline": {
        "name": "YourPipelineName",
        "roleArn": "CodePipeline_Service_Role",
        "artifactStores": {
            "us-east-1": {
                "type": "S3",
                "location": "S3 artifact bucket name, such as codepipeline-us-
east-1-1234567890"
            },
            "us-west-2": {
                "type": "S3",
                "location": "S3 artifact bucket name, such as codepipeline-us-
west-2-1234567890"
            }
        },
        "stages": [
            {

...
```

- I campi dei metadati della pipeline sono distinti dalla struttura della pipeline e non possono essere modificati. Quando aggiorni una pipeline, la data nel campo dei metadati `updated` viene modificata automaticamente.
- Quando modifichi o aggiorni una pipeline, il nome della pipeline non può essere modificato.

> **Note**
>
> Se vuoi rinominare una pipeline esistente, puoi utilizzare il comando `get-pipeline` dell'interfaccia a riga di comando per creare un file JSON che contiene la struttura della pipeline. Quindi puoi utilizzare il comando `create-pipeline` dell'interfaccia a riga di comando per creare una pipeline con quella struttura e assegnarle un nuovo nome.

Il numero di versione di una pipeline viene generato automaticamente e cambia ogni volta che si aggiorna la pipeline.

Requisiti della struttura delle azioni in CodePipeline

Un'operazione ha la seguente struttura ad alto livello:

```
[
                {
                   "inputArtifacts": [
                        An input artifact structure, if supported for the action
 category
                    ],
                   "name": "ActionName",
                   "region": "Region",
                   "namespace": "source_namespace",
                   "actionTypeId": {
                        "category": "An action category",
                        "owner": "AWS",
                        "version": "1"
                        "provider": "A provider type for the action category",
                   },
                   "outputArtifacts": [
                        An output artifact structure, if supported for the action
 category
                   ],
                   "configuration": {
                        Configuration details appropriate to the provider type
                   },
                   "runOrder": A positive integer that indicates the run order within
 the stage,
                }
            ]
```

Per un elenco di dettagli `configuration` di esempio appropriati per il tipo di provider, consulta Dettagli di configurazione per tipo di provider.

La struttura dell'operazione ha i seguenti requisiti:

- Tutti i nomi delle operazioni all'interno di una fase devono essere univoci.

- L'artefatto di input di un'azione deve corrispondere esattamente all'artefatto di output dichiarato in un'azione precedente. Ad esempio, se un'operazione precedente include la seguente dichiarazione:

```
"outputArtifacts": [
    {
    "MyApp"
    }
],
```

  e non ci sono altri artefatti di output, l'artefatto di input di un'azione successiva deve essere:

```
"inputArtifacts": [
    {
    "MyApp"
    }
],
```

  Questo vale per tutte le operazioni, sia che si trovino nella stessa fase o in fasi successive, ma l'artefatto di input non deve essere in stretta sequenza l'azione successiva dell'operazione che ha fornito l'artefatto di output. Le operazioni in parallelo possono dichiarare diversi bundle di artefatti di output che vengono a loro volta usati da diverse operazioni successive.
- I nomi degli artefatti di output devono essere univoci in una pipeline. Ad esempio, una pipeline può includere un'operazione che ha un artefatto di output denominato `"MyApp"` e un'altra operazione che ha un artefatto di output denominato `"MyBuiltApp"`. Una pipeline invece non può includere due operazioni che hanno entrambe un artefatto di output denominato `"MyApp"`.
- Le azioni interregionali utilizzano il `Region` campo per designare la AWS regione in cui devono essere create le azioni. Le AWS risorse create per questa azione devono essere create nella stessa regione fornita nel `region` campo. Non è possibile creare operazioni tra regioni per i seguenti tipi di operazione:
  - Operazioni di origine
  - Operazioni da provider di terze parti
  - Operazioni da provider personalizzati
- Le azioni possono essere configurate con variabili. È possibile utilizzare il campo `namespace` per impostare lo spazio dei nomi e le informazioni variabili per le variabili di esecuzione. Per informazioni di riferimento sulle variabili di esecuzione e sulle variabili di output delle azioni, vedere Variables.

- Per tutti i tipi di azione attualmente supportati, l'unica stringa proprietaria valida è `AWSThirdParty`, o`Custom`. Per ulteriori informazioni, consulta l'CodePipeline API Reference.
- Il valore predefinito di `runOrder` per un'operazione è 1. Il valore deve essere un numero intero positivo (numero naturale). Non puoi utilizzare frazioni, decimali, numeri negativi o zero. Per specificare una sequenza di operazioni in serie, utilizza il numero più piccolo per la prima operazione e i numeri più grandi per ciascuna delle restanti operazioni della sequenza. Per specificare operazioni parallele, usa lo stesso numero intero per ciascuna operazione che vuoi eseguire in parallelo. Nella console, puoi specificare una sequenza seriale per un'azione scegliendo Aggiungi gruppo di azioni al livello della fase in cui desideri che venga eseguita oppure puoi specificare una sequenza parallela scegliendo Aggiungi azione. Il gruppo di azioni si riferisce a un ordine di esecuzione di una o più azioni allo stesso livello.

  Ad esempio, se vuoi eseguire tre azioni in sequenza in una fase, darai alla prima operazione il valore di `runOrder` 1, alla seconda operazione il valore di `runOrder` 2 e alla terza il valore di `runOrder` 3. Tuttavia, se vuoi che la seconda e la terza operazione vengano eseguite in parallelo, darai alla prima operazione il valore di `runOrder` 1 e alla seconda e alla terza operazione il valore di `runOrder` 2.

  > **Note**
  >
  > La numerazione delle operazioni in serie non deve essere rigorosamente in sequenza. Ad esempio, se hai tre operazioni in una sequenza e decidi di rimuovere la seconda operazione, non è necessario rinumerare il valore di `runOrder` della terza operazione. Infatti, poiché il valore di `runOrder` di quell'operazione (3) è superiore al valore di `runOrder` della prima operazione (1), viene comunque eseguita in serie dopo la prima operazione della fase.

- Quando utilizzi un bucket Amazon S3 come posizione di distribuzione, specifichi anche una chiave di oggetto. Una chiave oggetto può essere un nome di file (oggetto) o una combinazione di un prefisso (percorso di cartella) e nome del file. È possibile utilizzare le variabili per specificare il nome della posizione che la pipeline deve usare. Le operazioni di distribuzione Amazon S3 supportano l'uso delle seguenti variabili nelle chiavi oggetto Amazon S3.

Utilizzo di variabili in Amazon S3

Variabile	Esempio di input di console	Output
datetime	js-application/{datetime}.zip	Timestamp UTC in questo formato: <AAAA> - <MM> -GG> _ <HH> - <MM> - <SS> Esempio: js-application/2019-01-10_07-39-57.zip
uuid	js-application/{uuid}.zip	L'UUID è un identificatore univoco globale che è sicuramente diverso da qualsiasi altro identificatore. L'UUID è in questo formato (tutte cifre in formato esadecimale): <8 cifre>-<4 cifre>-4 cifre>-<4 cifre>-<12 cifre> Esempio: js-application/54a60075-b96a-4bf3-9013-db3a9EXAMPLE.zip

- Queste sono le `actionTypeId` categorie valide per CodePipeline:
  - `Source`
  - `Build`
  - `Approval`
  - `Deploy`
  - `Test`
  - `Invoke`

  Alcuni tipi di provider e opzioni di configurazione sono forniti di seguito.
- I tipi di provider validi per una categoria di azioni dipendono dalla categoria. Ad esempio, per il tipo di operazione di origine, un tipo di provider valido è `S3`, `GitHub`, `CodeCommit` o `Amazon ECR`. Questo esempio illustra la struttura per un'operazione di origine con un provider `S3`:

```
"actionTypeId": {
  "category": "Source",
  "owner": "AWS",
  "version": "1",
  "provider": "S3"},
```

- Ogni operazione deve avere una configurazione valida che dipende dal tipo di provider dell'operazione stessa. Nella seguente tabella sono elencati gli elementi di configurazione dell'operazione richiesti per ogni tipo di provider valido:

  Proprietà di configurazione dell'operazione per i tipi di provider

Nome del provider	Nome del provider nel tipo di operazione	Proprietà di configurazione	Proprietà obbligatoria?
Amazon S3 (provider di azioni di distribuzione)	Per ulteriori informazioni, inclusi esempi relativi ai parametri di azione di distribuzione di Amazon S3, consulta. Azione di distribuzione di Amazon S3		
Amazon S3 (provider Source Action)	Per ulteriori informazioni, inclusi esempi relativi ai parametri di azione del codice sorgente di Amazon S3, consulta. Azione all'origine di Amazon S3		
Amazon ECR	Per ulteriori informazioni, inclusi esempi relativi ai parametri di Amazon ECR, consultaAmazon ECR.		
CodeCommit	Per ulteriori informazioni, inclusi esempi relativi ai CodeCommit parametri, consultaCodeCommit.		
GitHub	Per ulteriori informazioni, inclusi esempi relativi ai GitHub parametri, vedereGitHub riferimento alla struttura delle azioni di origine della versione 1.		
AWS CloudFormation	Per ulteriori informazioni, inclusi esempi relativi ai AWS CloudFormation parametri, vedereAWS CloudFormation.		

Nome del provider	Nome del provider nel tipo di operazione	Proprietà di configurazione	Proprietà obbligatoria?
CodeBuild	Per ulteriori descrizioni ed esempi relativi ai CodeBuild parametri, vedereAWS CodeBuild.		
CodeDeploy	Per ulteriori descrizioni ed esempi relativi ai CodeDeploy parametri, vedereAWS CodeDeploy.		
AWS Device Farm	Per ulteriori descrizioni ed esempi relativi ai AWS Device Farm parametri, vedereAWS Device Farm.		
AWS Elastic Beanstalk	`ElasticBeanstalk`	`ApplicationName`	Richiesto
		`EnvironmentName`	Richiesto
AWS Lambda	Per ulteriori informazioni, inclusi esempi relativi ai AWS Lambda parametri, vedereAWS Lambda.		
AWS OpsWorks Stacks	`OpsWorks`	`Stack`	Richiesto
		`Layer`	Facoltativo
		`App`	Richiesto
Amazon ECS	Per ulteriori descrizioni ed esempi relativi ai parametri di Amazon ECS, consultaAmazon Elastic Container Service.		
Amazon ECS e CodeDeploy (blu/verde)	Per ulteriori descrizioni ed esempi relativi ad Amazon ECS e ai parametri CodeDeploy blu/green, consulta. Amazon Elastic Container Service e CodeDeploy blu-verde		
Catalogo dei servizi	`ServiceCatalog`	`TemplateFilePath`	Richiesto
		`ProductVersionName`	Richiesto

Nome del provider	Nome del provider nel tipo di operazione	Proprietà di configurazione	Proprietà obbligatoria?
		`ProductType`	Richiesto
		`ProductVersionDescription`	Facoltativo
		`ProductId`	Richiesto
Alexa Skills Kit	`AlexaSkillsKit`	`ClientId`	Richiesto
		`ClientSecret`	Richiesto
		`RefreshToken`	Richiesto
		`SkillId`	Richiesto
Jenkins	Il nome dell'azione che hai fornito nel CodePipeline Plugin per Jenkins (ad esempio,) *`MyJenkinsProviderName`*	`ProjectName`	Richiesto
Approvazione manuale	`Manual`	`CustomData`	Facoltativo
		`ExternalEntityLink`	Facoltativo
		`NotificationArn`	Facoltativo

Argomenti

- Numero di artefatti di input e output per ogni tipo di operazione
- Impostazioni predefinite per il parametro PollForSourceChanges
- Dettagli di configurazione per tipo di provider

Numero di artefatti di input e output per ogni tipo di operazione

A seconda del tipo di operazione, puoi avere il seguente numero di artefatti di input e di output:

Vincoli del tipo di operazione per gli artefatti

Owner	Tipo di operazione	Provider	Numero di artefatti di input validi	Numero di artefatti di output validi
`AWS`	Origine	Amazon S3	0	1
`AWS`	Origine	CodeCommit	0	1
`AWS`	Origine	Amazon ECR	0	1
`ThirdParty`	Origine	GitHub	0	1
`AWS`	Creazione	CodeBuild	Da 1 a 5	Da 0 a 5
`AWS`	Test	CodeBuild	Da 1 a 5	Da 0 a 5
`AWS`	Test	AWS Device Farm	1	0
`AWS`	Approval	Manuale	0	0
`AWS`	Implementazione	Amazon S3	1	0
`AWS`	Implementazione	AWS CloudFormation	Da 0 a 10	Da 0 a 1
`AWS`	Implementazione	CodeDeploy	1	0
`AWS`	Implementazione	AWS Elastic Beanstalk	1	0
`AWS`	Implementazione	AWS OpsWorks Stacks	1	0
`AWS`	Implementazione	Amazon ECS	1	0

Owner	Tipo di operazione	Provider	Numero di artefatti di input validi	Numero di artefatti di output validi
`AWS`	Implementazione	Catalogo dei servizi	1	0
`AWS`	Invoke	AWS Lambda	Da 0 a 5	Da 0 a 5
`ThirdParty`	Implementazione	Alexa Skills Kit	Da 1 a 2	0
`Custom`	Creazione	Jenkins	Da 0 a 5	Da 0 a 5
`Custom`	Test	Jenkins	Da 0 a 5	Da 0 a 5
`Custom`	Qualsiasi categoria supportata	Come specificato nell'operazione personalizzata	Da 0 a 5	Da 0 a 5

Impostazioni predefinite per il parametro PollForSourceChanges

Il valore predefinito del parametro `PollForSourceChanges` è determinato dal metodo utilizzato per creare la pipeline, come decritto nella seguente tabella. In molti casi, il parametro `PollForSourceChanges` è preimpostato su "true" e deve essere disabilitato.

Quando il parametro `PollForSourceChanges` è preimpostato su "true", devi eseguire le operazioni seguenti:

- Aggiungere il parametro `PollForSourceChanges` al file JSON o al modello AWS CloudFormation .
- Crea risorse per il rilevamento delle modifiche (regola CloudWatch Eventi, se applicabile).
- Imposta il parametro `PollForSourceChanges` su false.

> **Note**
>
> Se si crea una regola CloudWatch Events o un webhook, è necessario impostare il parametro su false per evitare di attivare la pipeline più di una volta.

Il `PollForSourceChanges` parametro non viene utilizzato per le azioni di origine di Amazon ECR.

- PollForSourceChanges impostazioni predefinite dei parametri

Origine	Metodo di creazione	Esempio di output di struttura JSON "Configuration"
CodeCommit	La pipeline viene creata con la console (e le risorse di rilevamento modifiche sono create dalla console). Il parametro viene visualizzato nell'output della struttura della pipeline ed è preimpostato su `false`.	`BranchName": "main", "PollForSourceChanges": "false", "RepositoryName": "my-repo"`
	La pipeline viene creata con la CLI AWS CloudFormation o e `PollForSourceChanges` il parametro non viene visualizzato nell'output JSON, ma è impostato su ². `true`	`BranchName": "main", "RepositoryName": "my-repo"`
Amazon S3	La pipeline viene creata con la console (e le risorse di rilevamento modifiche sono create dalla console). Il parametro viene visualizzato nell'output della struttura della pipeline ed è preimpostato su `false`.	`"S3Bucket": "my-bucket", "S3ObjectKey": "object.zip", "PollForSourceChanges": "false"`
	La pipeline viene creata con la CLI AWS CloudFormation o e `PollForSourceChanges` il parametro non viene visualizzato nell'output JSON, ma è impostato su ². `true`	`"S3Bucket": "my-bucket", "S3ObjectKey": "object.zip"`

Origine	Metodo di creazione	Esempio di output di struttura JSON "Configuration"
GitHub	La pipeline viene creata con la console (e le risorse di rilevamento modifiche sono create dalla console). Il parametro viene visualizzato nell'output della struttura della pipeline ed è preimpostato su `false`.	`"Owner": "MyGitHubAccountName ", "Repo": " MyGitHubRepositoryName " "PollForSourceChanges": "false", "Branch": " main" "OAuthToken": " ****"`
	La pipeline viene creata con la CLI AWS CloudFormation o e `PollForSourceChanges` il parametro non viene visualizzato nell'output JSON, ma è impostato su ². `true`	`"Owner": "MyGitHubAccountName ", "Repo": "MyGitHubRepositoryName ", "Branch": " main", "OAuthToken": " ****"`
	² Se `PollForSourceChanges` è stato aggiunto in qualsiasi momento alla struttura JSON o al AWS CloudFormation modello, viene visualizzato come mostrato: `"PollForSourceChanges": "true",` ³ Per informazioni sulle risorse di rilevamento delle modifiche che si applicano a ciascun provider di origine, vedere Metodi di rilevamento delle modifiche.	

Dettagli di configurazione per tipo di provider

Questa sezione elenca i parametri `configuration` validi per ogni provider di operazione.

L'esempio seguente mostra una configurazione valida per un'azione di distribuzione che utilizza Service Catalog, per una pipeline creata nella console senza un file di configurazione separato:

```
"configuration": {
  "TemplateFilePath": "S3_template.json",
```

```
  "ProductVersionName": "devops S3 v2",
  "ProductType": "CLOUD_FORMATION_TEMPLATE",
  "ProductVersionDescription": "Product version description",
  "ProductId": "prod-example123456"
}
```

L'esempio seguente mostra una configurazione valida per un'azione di distribuzione che utilizza Service Catalog, per una pipeline creata nella console con un file di `sample_config.json` configurazione separato:

```
"configuration": {
  "ConfigurationFilePath": "sample_config.json",
  "ProductId": "prod-example123456"
}
```

L'esempio seguente mostra la configurazione valida per un'operazione di distribuzione che utilizza Alexa Skills Kit:

```
"configuration": {
  "ClientId": "amzn1.application-oa2-client.aadEXAMPLE",
  "ClientSecret": "****",
  "RefreshToken": "****",
  "SkillId": "amzn1.ask.skill.22649d8f-0451-4b4b-9ed9-bfb6cEXAMPLE"
}
```

L'esempio seguente mostra una configurazione valida per un'approvazione manuale:

```
"configuration": {
  "CustomData": "Comments on the manual approval",
  "ExternalEntityLink": "http://my-url.com",
  "NotificationArn": "arn:aws:sns:us-west-2:12345EXAMPLE:Notification"
}
```

Riferimento per la struttura delle operazioni

Questa sezione è un riferimento solo per la configurazione dell'operazione. Per una panoramica dei concetti di base della struttura della pipeline, consulta CodePipeline riferimento alla struttura della tubazione.

Ogni provider di azioni CodePipeline utilizza un set di campi di configurazione obbligatori e opzionali nella struttura della pipeline. Questa sezione fornisce le seguenti informazioni di riferimento in base al provider di operazione:

- Valori validi per i campi `ActionType` inclusi nel blocco dell'operazione della struttura della pipeline, ad esempio `Owner` e `Provider`.
- Descrizioni e altre informazioni di riferimento per i parametri `Configuration` (obbligatori e facoltativi) inclusi nella sezione relativa all'operazione della struttura della pipeline.
- Campi delle operazioni JSON e YAML di esempio validi.

Questa sezione viene aggiornata periodicamente con più provider di operazione. Le informazioni di riferimento sono attualmente disponibili per i seguenti provider di operazione:

Argomenti

- Amazon ECR
- Amazon Elastic Container Service e CodeDeploy blu-verde
- Amazon Elastic Container Service
- Azione di distribuzione di Amazon S3
- Azione all'origine di Amazon S3
- AWS AppConfig
- AWS CloudFormation
- AWS CloudFormation StackSets
- AWS CodeBuild
- CodeCommit
- AWS CodeDeploy
- CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite
- AWS Device Farm

- AWS Lambda
- Riferimento alla struttura d'azione Snyk
- AWS Step Functions

Amazon ECR

Attiva la pipeline quando una nuova immagine viene inviata al repository Amazon ECR. Questa azione fornisce un file di definizioni delle immagini che fa riferimento all'URI per l'immagine che è stata inviata ad Amazon ECR. Questa azione di origine viene spesso utilizzata insieme a un'altra azione di origine, ad esempio per consentire una posizione di origine per tutti gli altri artefatti di origine. CodeCommit Per ulteriori informazioni, consulta Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy .

Quando si utilizza la console per creare o modificare la pipeline, CodePipeline crea una regola CloudWatch Events che avvia la pipeline quando si verifica una modifica nel repository.

Devi aver già creato un repository Amazon ECR e inviato un'immagine prima di connettere la pipeline tramite un'azione Amazon ECR.

Argomenti

Tipo di operazione

- Categoria: `Source`
- Proprietario: `AWS`
- Provider: `ECR`
- Versione: `1`

Parametri di configurazione

RepositoryName

Campo obbligatorio: sì

Il nome del repository Amazon ECR in cui è stata inserita l'immagine.

ImageTag

Campo obbligatorio: no

Il tag utilizzato per l'immagine.

> **Note**
>
> Se non viene specificato un valore per `ImageTag`, il valore predefinito è `latest`.

Input artifact (Artefatti di input)

- Numero di artefatti: `0`
- Descrizione: gli artefatti di input non si applicano a questo tipo di azione.

Artefatti di output

- Numero di artefatti: `1`
- Descrizione: questa azione produce un artefatto che contiene un file `imageDetail.json` contenente l'URI per l'immagine che ha attivato l'esecuzione della pipeline. Per ulteriori informazioni sul file `imageDetail.json`, consulta File ImageDetail.json per le azioni di distribuzione blu/verde di Amazon ECS.

Variabili di output

Quando è configurata, questa azione produce variabili che possono essere referenziate dalla configurazione dell'azione di un'azione downstream nella pipeline. Questa azione produce variabili che possono essere viste come variabili di output, anche se l'azione non ha uno spazio dei nomi. È possibile configurare un'azione con uno spazio dei nomi per rendere tali variabili disponibili per la configurazione delle azioni downstream.

Per ulteriori informazioni, consulta Variables.

RegistryId

L'ID AWS dell'account associato al registro che contiene il repository.

RepositoryName

Il nome del repository Amazon ECR in cui è stata inserita l'immagine.

ImageTag

Il tag utilizzato per l'immagine.

ImageDigest

Il file digest `sha256` del manifest delle immagini.

ImageURI

L'URI dell'immagine.

Dichiarazione di azione (esempio Amazon ECR)

YAML

```
Name: Source
Actions:
  - InputArtifacts: []
    ActionTypeId:
      Version: '1'
      Owner: AWS
      Category: Source
      Provider: ECR
    OutputArtifacts:
      - Name: SourceArtifact
    RunOrder: 1
    Configuration:
      ImageTag: latest
      RepositoryName: my-image-repo

    Name: ImageSource
```

JSON

```
{
    "Name": "Source",
    "Actions": [
        {
            "InputArtifacts": [],
            "ActionTypeId": {
                "Version": "1",
                "Owner": "AWS",
                "Category": "Source",
                "Provider": "ECR"
            },
            "OutputArtifacts": [
                {
                    "Name": "SourceArtifact"
                }
            ],
            "RunOrder": 1,
            "Configuration": {
                "ImageTag": "latest",
                "RepositoryName": "my-image-repo"
            },
            "Name": "ImageSource"
        }
    ]
},
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy — Questo tutorial fornisce un esempio di file di specifiche dell'app e un gruppo di CodeDeploy applicazione e distribuzione di esempio per creare una pipeline con una fonte CodeCommit Amazon ECR da distribuire su istanze Amazon ECS.

Amazon Elastic Container Service e CodeDeploy blu-verde

Puoi configurare una pipeline in AWS CodePipeline cui distribuire applicazioni container utilizzando una distribuzione blu/verde. In una distribuzione blu/verde, puoi lanciare una nuova versione dell'applicazione insieme a quella precedente e testare la nuova versione prima di reindirizzare il traffico verso di essa. Puoi anche monitorare il processo di distribuzione ed eseguire rapidamente il rollback in caso di problemi.

La pipeline completata rileva le modifiche alle immagini o al file di definizione delle attività e le utilizza CodeDeploy per indirizzare e distribuire il traffico verso un cluster Amazon ECS e un sistema di bilanciamento del carico. CodeDeploy crea un nuovo listener sul tuo sistema di bilanciamento del carico che può indirizzare la nuova attività tramite una porta speciale. Puoi anche configurare la pipeline per utilizzare una posizione di origine, ad esempio un CodeCommit repository, in cui è archiviata la definizione delle attività di Amazon ECS.

Prima di creare la pipeline, devi aver già creato le risorse Amazon ECS, le risorse, il CodeDeploy sistema di bilanciamento del carico e il gruppo target. Devi aver già taggato e archiviato l'immagine nel tuo repository di immagini e aver caricato la definizione dell'attività e il file nel tuo archivio di AppSpec file.

> **Note**
>
> Questo argomento descrive l'azione di distribuzione di Amazon ECS to CodeDeploy blue/green per. CodePipeline Per informazioni di riferimento sulle azioni di distribuzione standard di Amazon ECS in CodePipeline, consultaAmazon Elastic Container Service.

Argomenti

Tipo di operazione

- Categoria: `Deploy`
- Proprietario: `AWS`
- Provider: `CodeDeployToECS`
- Versione: `1`

Parametri di configurazione

ApplicationName

Campo obbligatorio: sì

Il nome dell'applicazione in CodeDeploy. Prima di creare la pipeline, è necessario aver già creato l'applicazione in CodeDeploy.

DeploymentGroupName

Campo obbligatorio: sì

Il gruppo di distribuzione specificato per i set di attività di Amazon ECS che hai creato per la tua CodeDeploy applicazione. Prima di creare la pipeline, devi aver già creato il gruppo di distribuzione in. CodeDeploy

TaskDefinitionTemplateArtifact

Campo obbligatorio: sì

Il nome dell'elemento di input che fornisce il file di definizione dell'attività all'azione di distribuzione. Questo è in genere il nome dell'artefatto di output dell'azione di origine. Quando si utilizza la console, il nome predefinito per l'elemento di output dell'azione di origine è. `SourceArtifact`

AppSpecTemplateArtifact

Campo obbligatorio: sì

Il nome dell'elemento di input che fornisce il AppSpec file all'azione di distribuzione. Tale valore viene aggiornato quando la pipeline viene eseguita. Questo è generalmente il nome dell'artefatto di output dell'azione di origine. Quando si utilizza la console, il nome predefinito per l'elemento di output dell'azione di origine è. `SourceArtifact` Per `TaskDefinition` nel AppSpec file, puoi conservare il testo `<TASK_DEFINITION>` segnaposto come mostrato qui.

AppSpecTemplatePath

Campo obbligatorio: no

Il nome del file memorizzato nella posizione del AppSpec file di origine della pipeline, ad esempio nel repository della pipeline. CodeCommit Il nome file predefinito è `appspec.yaml`. Se il AppSpec file ha lo stesso nome ed è archiviato al livello principale del repository di file, non è necessario fornire il nome del file. Se il percorso non è quello predefinito, inserite il percorso e il nome del file.

TaskDefinitionTemplatePath

Campo obbligatorio: no

Il nome del file della definizione dell'attività memorizzata nella posizione di origine del file della pipeline, ad esempio nel repository della CodeCommit pipeline. Il nome file predefinito è `taskdef.json`. Se il file di definizione dell'attività ha lo stesso nome ed è archiviato a livello principale nel repository dei file, non è necessario fornire il nome del file. Se il percorso non è quello predefinito, inserite il percorso e il nome del file.

Immagine **<**Number**>**ArtifactName

Campo obbligatorio: no

Il nome dell'elemento di input che fornisce l'immagine all'azione di distribuzione. Si tratta in genere dell'artefatto di output del repository di immagini, ad esempio l'output dell'azione sorgente di Amazon ECR.

I valori disponibili per `<Number>` sono compresi tra 1 e 4.

Immagine **<**Number**>**ContainerName

Campo obbligatorio: no

Il nome dell'immagine disponibile nell'archivio di immagini, ad esempio l'archivio dei sorgenti di Amazon ECR.

I valori disponibili per `<Number>` sono compresi tra 1 e 4.

Input artifact (Artefatti di input)

- Numero di artefatti: `1 to 5`

- Descrizione: l'`CodeDeployToECS`azione cerca innanzitutto il file di definizione dell'attività e il AppSpec file nell'archivio dei file di origine, quindi cerca l'immagine nell'archivio delle immagini, quindi genera dinamicamente una nuova revisione della definizione dell'attività e infine esegue AppSpec i comandi per distribuire il set di attività e il contenitore nel cluster.

  L'`CodeDeployToECS`azione cerca un `imageDetail.json` file che associa l'URI dell'immagine all'immagine. Quando esegui una modifica al tuo repository di immagini Amazon ECR, l'azione sorgente ECR della pipeline crea un `imageDetail.json` file per quel commit. Puoi anche aggiungere manualmente un `imageDetail.json` file per una pipeline in cui l'azione non è automatizzata. Per ulteriori informazioni sul file `imageDetail.json`, consulta File ImageDetail.json per le azioni di distribuzione blu/verde di Amazon ECS.

  L'`CodeDeployToECS`azione genera dinamicamente una nuova revisione della definizione dell'attività. In questa fase, questa azione sostituisce i segnaposto nel file di definizione dell'attività nell'URI dell'immagine recuperato dai file ImageDetail.json. <IMAGE1_NAME>Ad esempio, se imposti IMAGE1_NAME come ContainerName parametro Image1, devi specificare il segnaposto come valore del campo immagine nel file di definizione dell'attività. In questo caso, l'azione CodeDeployTo ECS sostituisce il segnaposto nell'<IMAGE1_NAME>URI dell'immagine effettivo recuperato da ImageDetail.json nell'artefatto specificato come Image1. ArtifactName

  Per gli CodeDeploy `AppSpec.yaml` aggiornamenti `TaskDefinition` delle definizioni delle attività, il file contiene la proprietà.

  ```
  TaskDefinition: <TASK_DEFINITION>
  ```

  Questa proprietà verrà aggiornata dall'`CodeDeployToECS`azione dopo la creazione della nuova definizione di attività.

  <TASK_DEFINITION>Per il valore del `TaskDefinition` campo, il testo segnaposto deve essere. L'`CodeDeployToECS`azione sostituisce questo segnaposto con l'ARN effettivo della definizione dell'attività generata dinamicamente.

Artefatti di output

- Numero di artefatti: `0`
- Descrizione: gli artefatti di output non si applicano a questo tipo di azione.

Dichiarazione dell'operazione

YAML

```
Name: Deploy
Actions:
  - Name: Deploy
    ActionTypeId:
      Category: Deploy
      Owner: AWS
      Provider: CodeDeployToECS
      Version: '1'
    RunOrder: 1
    Configuration:
      AppSpecTemplateArtifact: SourceArtifact
      ApplicationName: ecs-cd-application
      DeploymentGroupName: ecs-deployment-group
      Image1ArtifactName: MyImage
      Image1ContainerName: IMAGE1_NAME
      TaskDefinitionTemplatePath: taskdef.json
      AppSpecTemplatePath: appspec.yaml
      TaskDefinitionTemplateArtifact: SourceArtifact
    OutputArtifacts: []
    InputArtifacts:
      - Name: SourceArtifact
      - Name: MyImage
    Region: us-west-2
    Namespace: DeployVariables
```

JSON

```
{
    "Name": "Deploy",
    "Actions": [
        {
            "Name": "Deploy",
            "ActionTypeId": {
                "Category": "Deploy",
                "Owner": "AWS",
                "Provider": "CodeDeployToECS",
                "Version": "1"
            },
            "RunOrder": 1,
```

```
            "Configuration": {
                "AppSpecTemplateArtifact": "SourceArtifact",
                "ApplicationName": "ecs-cd-application",
                "DeploymentGroupName": "ecs-deployment-group",
                "Image1ArtifactName": "MyImage",
                "Image1ContainerName": "IMAGE1_NAME",
                "TaskDefinitionTemplatePath": "taskdef.json",
                "AppSpecTemplatePath": "appspec.yaml",
                "TaskDefinitionTemplateArtifact": "SourceArtifact"
            },
            "OutputArtifacts": [],
            "InputArtifacts": [
                {
                    "Name": "SourceArtifact"
                },
                {
                    "Name": "MyImage"
                }
            ],
            "Region": "us-west-2",
            "Namespace": "DeployVariables"
        }
    ]
}
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy — Questo tutorial illustra la creazione delle CodeDeploy risorse Amazon ECS necessarie per una distribuzione blu/verde. Il tutorial mostra come inviare un'immagine Docker ad Amazon ECR e creare una definizione di attività Amazon ECS che elenchi il nome dell'immagine Docker, il nome del contenitore, il nome del servizio Amazon ECS e la configurazione del load balancer. Il tutorial ti guida quindi attraverso la creazione del AppSpec file e della pipeline per la tua implementazione.

> **Note**
>
> Questo argomento e questo tutorial descrivono l'azione CodeDeploy /ECS blue/green per. CodePipeline Per informazioni sulle azioni standard ECS in CodePipeline, consulta Tutorial: Continuous Deployment with. CodePipeline

- AWS CodeDeploy Guida per l'utente: per informazioni su come utilizzare il load balancer, il listener di produzione, i gruppi target e l'applicazione Amazon ECS in una distribuzione blu/verde, consulta Tutorial: Deploy an Amazon ECS Service. Queste informazioni di riferimento nella Guida per l'AWS CodeDeploy utente forniscono una panoramica delle distribuzioni blu/green con Amazon ECS e. AWS CodeDeploy
- Guida per sviluppatori di Amazon Elastic Container Service: per informazioni sull'utilizzo di immagini e contenitori Docker, servizi e cluster ECS e set di attività ECS, consulta What Is Amazon ECS?

Amazon Elastic Container Service

Puoi utilizzare un'azione Amazon ECS per distribuire un servizio Amazon ECS e un set di attività. Un servizio Amazon ECS è un'applicazione contenitore che viene distribuita in un cluster Amazon ECS. Un cluster Amazon ECS è una raccolta di istanze che ospitano l'applicazione container nel cloud. La distribuzione richiede una definizione di attività creata in Amazon ECS e un file di definizioni delle immagini da utilizzare CodePipeline per distribuire l'immagine.

> **Important**
>
> L'azione di distribuzione standard di Amazon ECS CodePipeline crea la propria revisione della definizione dell'attività in base alla revisione utilizzata dal servizio Amazon ECS. Se crei nuove revisioni per la definizione dell'attività senza aggiornare il servizio Amazon ECS, l'azione di distribuzione ignorerà tali revisioni.

Prima di creare la pipeline, devi aver già creato le risorse Amazon ECS, taggato e archiviato l'immagine nel tuo repository di immagini e caricato il BuildSpec file nel tuo archivio di file.

> **Note**
>
> Questo argomento di riferimento descrive l'azione di distribuzione standard di Amazon ECS per CodePipeline. Per informazioni di riferimento sulle azioni di distribuzione da Amazon ECS a CodeDeploy blue/green in CodePipeline, consulta. Amazon Elastic Container Service e CodeDeploy blu-verde

Argomenti

Tipo di operazione

- Categoria: `Deploy`
- Proprietario: `AWS`
- Provider: `ECS`
- Versione: `1`

Parametri di configurazione

ClusterName

Campo obbligatorio: sì

Il cluster Amazon ECS in Amazon ECS.

ServiceName

Campo obbligatorio: sì

Il servizio Amazon ECS che hai creato in Amazon ECS.

FileName

Campo obbligatorio: no

Il nome del file di definizioni delle immagini, il file JSON che descrive il nome del contenitore del servizio, l'immagine e il tag. Utilizzi questo file per le distribuzioni standard ECS. Per ulteriori informazioni, consulta Input artifact (Artefatti di input) e file imagedefinitions.json per le azioni di distribuzione standard di Amazon ECS.

DeploymentTimeout

Campo obbligatorio: no

Il timeout dell'azione di distribuzione di Amazon ECS in minuti. Il timeout è configurabile fino al timeout predefinito massimo per questa operazione. Per esempio:

```
"DeploymentTimeout": "15"
```

Input artifact (Artefatti di input)

- Numero di artefatti: `1`
- Descrizione: l'azione cerca un `imagedefinitions.json` file nell'archivio dei file di origine della pipeline. Un documento di definizioni delle immagini è un file JSON che descrive il nome del contenitore Amazon ECS, l'immagine e il tag. CodePipeline utilizza il file per recuperare l'immagine dal tuo repository di immagini come Amazon ECR. Puoi aggiungere manualmente un `imagedefinitions.json` file per una pipeline in cui l'azione non è automatizzata. Per ulteriori informazioni sul file `imagedefinitions.json`, consulta file imagedefinitions.json per le azioni di distribuzione standard di Amazon ECS.

  L'azione richiede un'immagine esistente che è già stata inserita nel tuo archivio di immagini. Poiché la mappatura delle immagini è fornita dal `imagedefinitions.json` file, l'azione non richiede che la sorgente Amazon ECR sia inclusa come azione sorgente nella pipeline.

Artefatti di output

- Numero di artefatti: `0`
- Descrizione: gli artefatti di output non si applicano a questo tipo di azione.

Dichiarazione dell'operazione

YAML

```
Name: DeployECS
ActionTypeId:
  Category: Deploy
  Owner: AWS
  Provider: ECS
  Version: '1'
RunOrder: 2
Configuration:
  ClusterName: my-ecs-cluster
  ServiceName: sample-app-service
  FileName: imagedefinitions.json
  DeploymentTimeout: '15'
OutputArtifacts: []
InputArtifacts:
  - Name: my-image
```

JSON

```
{
    "Name": "DeployECS",
    "ActionTypeId": {
        "Category": "Deploy",
        "Owner": "AWS",
        "Provider": "ECS",
        "Version": "1"
    },
    "RunOrder": 2,
    "Configuration": {
        "ClusterName": "my-ecs-cluster",
        "ServiceName": "sample-app-service",
        "FileName": "imagedefinitions.json",
        "DeploymentTimeout": "15"
    },
    "OutputArtifacts": [],
    "InputArtifacts": [
        {
            "Name": "my-image"
        }
    ]
```

```
},
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Tutorial: distribuzione continua con CodePipeline: questo tutorial mostra come creare un Dockerfile da archiviare in un repository di file sorgente come. CodeCommit Successivamente, il tutorial mostra come incorporare un CodeBuild BuildSpec file che crea e invia la tua immagine Docker ad Amazon ECR e crea il tuo file imagedefinitions.json. Infine, crei un servizio Amazon ECS e una definizione di attività, quindi crei la pipeline con un'azione di distribuzione di Amazon ECS.

  > **Note**
  >
  > Questo argomento e questo tutorial descrivono l'azione di distribuzione standard di Amazon ECS per CodePipeline. Per informazioni sulle azioni di distribuzione da Amazon ECS a CodeDeploy blue/green in CodePipeline, consulta. Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy

- Guida per sviluppatori di Amazon Elastic Container Service: per informazioni sull'utilizzo di immagini e contenitori Docker, servizi e cluster Amazon ECS e set di attività Amazon ECS, consulta What Is Amazon ECS?

Azione di distribuzione di Amazon S3

Utilizzi un'azione di distribuzione di Amazon S3 per distribuire file in un bucket Amazon S3 per l'hosting o l'archiviazione di siti Web statici. Puoi specificare se estrarre i file di distribuzione prima di caricarli nel tuo bucket.

> **Note**
>
> Questo argomento di riferimento descrive l'azione di distribuzione di Amazon S3 per i CodePipeline casi in cui la piattaforma di distribuzione è un bucket Amazon S3 configurato per l'hosting. Per informazioni di riferimento sull'azione del codice sorgente di Amazon S3 in CodePipeline, consulta. Azione all'origine di Amazon S3

Argomenti

Tipo di operazione

- Categoria: `Deploy`
- Proprietario: `AWS`
- Provider: `S3`
- Versione: `1`

Parametri di configurazione

BucketName

Campo obbligatorio: sì

Il nome del bucket Amazon S3 in cui devono essere distribuiti i file.

Estrarre

Campo obbligatorio: sì

Se true, specifica che i file devono essere estratti prima del caricamento. In caso contrario, i file dell'applicazione rimangono compressi durante il caricamento, ad esempio nel caso di un sito Web statico ospitato. Se false, allora `ObjectKey` è necessario.

ObjectKey

Condizionale. Obbligatoria se `Extract` = false

Il nome della chiave oggetto Amazon S3 che identifica in modo univoco l'oggetto nel bucket S3.

KMS ARN EncryptionKey

Campo obbligatorio: no

L'ARN della chiave di AWS KMS crittografia per il bucket host. Il `KMSEncryptionKeyARN` parametro crittografa gli artefatti caricati con quanto fornito. AWS KMS key Per una chiave KMS, puoi utilizzare l'ID chiave, la chiave ARN o l'alias ARN.

> **Note**
>
> Gli alias sono riconosciuti solo nell'account che ha creato la chiave KMS. Per le operazioni tra account, puoi utilizzare solo l'ID della chiave o l'ARN della chiave per identificare la chiave. Le operazioni tra account comportano l'utilizzo del ruolo dell'altro account (AccountB), pertanto specificando l'ID chiave verrà utilizzata la chiave dell'altro account (AccountB).

> **Important**
>
> CodePipeline supporta solo chiavi KMS simmetriche. Non utilizzare una chiave KMS asimmetrica per crittografare i dati nel bucket S3.

CannedACL

Campo obbligatorio: no

Il `CannedACL` parametro applica l'ACL predefinito specificato agli oggetti distribuiti su Amazon S3. Tale applicazione sovrascrive l'ACL esistente applicata all'oggetto.

CacheControl

Campo obbligatorio: no

Il `CacheControl` parametro controlla il comportamento di memorizzazione nella cache per le richieste/risposte per gli oggetti nel bucket. Per un elenco dei valori validi, vedi il campo di intestazione `Cache-Control` per le operazioni HTTP. Per inserire più valori in `CacheControl`, utilizzare una virgola tra ogni valore. Puoi aggiungere uno spazio dopo ogni virgola (facoltativo), come mostrato in questo esempio per l'interfaccia a riga di comando:

```
"CacheControl": "public, max-age=0, no-transform"
```

Input artifact (Artefatti di input)

- Numero di artefatti: 1
- Descrizione: i file per la distribuzione o l'archiviazione sono ottenuti dal repository di origine, compressi e caricati da. CodePipeline

Artefatti di output

- Numero di artefatti: `0`
- Descrizione: gli artefatti di output non si applicano a questo tipo di azione.

Esempio di configurazione dell'operazione

Di seguito sono riportati alcuni esempi per la configurazione dell'azione.

Esempio di configurazione quando **`Extract`** è impostato su **`false`**

L'esempio seguente mostra la configurazione dell'azione predefinita quando l'azione viene creata con il `Extract` campo impostato su`false`.

YAML

```
Name: Deploy
Actions:
  - Name: Deploy
    ActionTypeId:
      Category: Deploy
      Owner: AWS
      Provider: S3
      Version: '1'
    RunOrder: 1
    Configuration:
      BucketName: website-bucket
      Extract: 'false'
    OutputArtifacts: []
    InputArtifacts:
```

```
      - Name: SourceArtifact
    Region: us-west-2
    Namespace: DeployVariables
```

JSON

```
{
    "Name": "Deploy",
    "Actions": [
        {
            "Name": "Deploy",
            "ActionTypeId": {
                "Category": "Deploy",
                "Owner": "AWS",
                "Provider": "S3",
                "Version": "1"
            },
            "RunOrder": 1,
            "Configuration": {
                "BucketName": "website-bucket",
                "Extract": "false"
                },
            "OutputArtifacts": [],
            "InputArtifacts": [
                {
                    "Name": "SourceArtifact"
                }
            ],
            "Region": "us-west-2",
            "Namespace": "DeployVariables"
        }
    ]
},
```

Esempio di configurazione quando **`Extract`** è impostato su **`true`**

L'esempio seguente mostra la configurazione dell'azione predefinita quando l'azione viene creata con il `Extract` campo impostato su`true`.

YAML

```
Name: Deploy
```

```
Actions:
  - Name: Deploy
    ActionTypeId:
      Category: Deploy
      Owner: AWS
      Provider: S3
      Version: '1'
    RunOrder: 1
    Configuration:
      BucketName: website-bucket
      Extract: 'true'
      ObjectKey: MyWebsite
    OutputArtifacts: []
    InputArtifacts:
      - Name: SourceArtifact
    Region: us-west-2
    Namespace: DeployVariables
```

JSON

```
{
    "Name": "Deploy",
    "Actions": [
        {
            "Name": "Deploy",
            "ActionTypeId": {
                "Category": "Deploy",
                "Owner": "AWS",
                "Provider": "S3",
                "Version": "1"
            },
            "RunOrder": 1,
            "Configuration": {
                "BucketName": "website-bucket",
                "Extract": "true",
                "ObjectKey": "MyWebsite"
                },
            "OutputArtifacts": [],
            "InputArtifacts": [
                {
                    "Name": "SourceArtifact"
                }
            ],
```

```
            "Region": "us-west-2",
            "Namespace": "DeployVariables"
        }
    ]
},
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Tutorial: crea una pipeline che utilizzi Amazon S3 come provider di distribuzione— Questo tutorial illustra due esempi di creazione di una pipeline con un'azione di distribuzione di S3. Scarichi file di esempio, carichi i file nel tuo CodeCommit repository, crei il tuo bucket S3 e configuri il bucket per l'hosting. Successivamente, usi la CodePipeline console per creare la pipeline e specificare una configurazione di distribuzione di Amazon S3.
- Azione all'origine di Amazon S3— Questo riferimento all'azione fornisce informazioni di riferimento ed esempi per le azioni di origine di Amazon S3 in. CodePipeline

Azione all'origine di Amazon S3

Attiva la pipeline quando un nuovo oggetto viene caricato nel bucket e nella chiave oggetto configurati.

> **Note**
>
> Questo argomento di riferimento descrive l'azione sorgente di Amazon S3 per CodePipeline cui la posizione di origine è un bucket Amazon S3 configurato per il controllo delle versioni. Per informazioni di riferimento sull'azione di distribuzione di Amazon S3 in CodePipeline, consulta. Azione di distribuzione di Amazon S3

Puoi creare un bucket Amazon S3 da utilizzare come posizione di origine per i file dell'applicazione.

> **Note**
>
> Quando crei il bucket di origine, assicurati di abilitare il controllo delle versioni nel bucket. Se desideri utilizzare un bucket Amazon S3 esistente, consulta Usare il controllo delle versioni per abilitare il controllo delle versioni su un bucket esistente.

Se utilizzi la console per creare o modificare la pipeline, CodePipeline crea una regola CloudWatch Events che avvia la pipeline quando si verifica una modifica nel bucket di origine S3.

È necessario aver già creato un bucket di origine Amazon S3 e aver caricato i file di origine come singolo file ZIP prima di connettere la pipeline tramite un'azione Amazon S3.

> **Note**
>
> Se Amazon S3 è il fornitore di origine per la tua pipeline, puoi comprimere il file o i file sorgente in un unico .zip e caricare il file.zip nel tuo bucket di origine. È inoltre possibile caricare un singolo file decompresso; tuttavia, le operazioni a valle che si aspettano un file con estensione .zip avranno esito negativo.

Argomenti

Tipo di operazione

- Categoria: `Source`
- Proprietario: `AWS`
- Provider: `S3`

- Versione: 1

Parametri di configurazione

S3Bucket

Campo obbligatorio: sì

Il nome del bucket Amazon S3 in cui devono essere rilevate le modifiche all'origine.

S3 ObjectKey

Campo obbligatorio: sì

Il nome della chiave oggetto Amazon S3 in cui devono essere rilevate le modifiche all'origine.

AllowOverrideForS3 ObjectKey

Campo obbligatorio: no

`AllowOverrideForS3ObjectKey`controlla se source overrides from `StartPipelineExecution` può sovrascrivere quello già configurato `S3ObjectKey` nell'azione source. Per ulteriori informazioni sulle sostituzioni dei sorgenti con la chiave S3 Object, consulta. Avvia una pipeline con una modifica della revisione del codice sorgente

> **Important**
>
> Se omettiAllowOverrideForS3ObjectKey, per CodePipeline impostazione predefinita la possibilità di sovrascrivere S3 ObjectKey nell'azione di origine impostando questo parametro su. `false`

I valori validi per questo parametro sono:

- `true`: Se impostata, la chiave oggetto S3 preconfigurata può essere sostituita dalle sostituzioni delle revisioni del codice sorgente durante l'esecuzione di una pipeline.

> **Note**
>
> Se intendi consentire a tutti CodePipeline gli utenti la possibilità di sovrascrivere la chiave oggetto S3 preconfigurata durante l'avvio di una nuova esecuzione della pipeline, devi impostare su. AllowOverrideForS3ObjectKey `true`

- `false`:

  Se impostata, non CodePipeline consentirà la sovrascrittura della chiave oggetto S3 utilizzando le sostituzioni delle revisioni del codice sorgente. Questo è anche il valore predefinito per questo parametro.

PollForSourceChanges

Campo obbligatorio: no

`PollForSourceChanges`controlla se interroga CodePipeline il bucket di origine di Amazon S3 per verificare eventuali modifiche alla fonte. Ti consigliamo invece di utilizzare CloudWatch Events e di rilevare CloudTrail le modifiche all'origine. Per ulteriori informazioni sulla configurazione CloudWatch degli eventi, consulta Migra le pipeline di polling con un codice sorgente e trail CloudTrail (CLI) S3 oMigra le pipeline di polling con una sorgente e un trail S3 (modello) CloudTrail AWS CloudFormation.

> **Important**
>
> Se intendi configurare gli CloudWatch eventi, devi impostarlo su `PollForSourceChanges` per `false` evitare esecuzioni duplicate della pipeline.

I valori validi per questo parametro sono:

- `true`: Se impostata, verifica la posizione di origine per CodePipeline verificare se sono state apportate modifiche alla fonte.

  > **Note**
  >
  > Se si omette`PollForSourceChanges`, per CodePipeline impostazione predefinita esegue il sondaggio della posizione di origine per verificare eventuali modifiche alla fonte. Questo comportamento è lo stesso se `PollForSourceChanges` è incluso e impostato su `true`.

- `false`: se impostata, CodePipeline non esegue il sondaggio della posizione di origine per verificare eventuali modifiche alla fonte. Utilizza questa impostazione se intendi configurare una regola CloudWatch Events per rilevare le modifiche all'origine.

Input artifact (Artefatti di input)

- Numero di artefatti: `0`
- Descrizione: gli artefatti di input non si applicano a questo tipo di azione.

Artefatti di output

- Numero di artefatti: `1`
- Descrizione: fornisce gli artefatti disponibili nel bucket di origine configurato per connettersi alla pipeline. Gli artefatti generati dal bucket sono gli artefatti di output per l'azione Amazon S3. I metadati degli oggetti Amazon S3 (ETag e ID di versione) vengono visualizzati CodePipeline come revisione di origine per l'esecuzione della pipeline attivata.

Variabili di output

Quando è configurata, questa azione produce variabili che possono essere referenziate dalla configurazione dell'azione di un'azione downstream nella pipeline. Questa azione produce variabili che possono essere viste come variabili di output, anche se l'azione non ha uno spazio dei nomi. È possibile configurare un'azione con uno spazio dei nomi per rendere tali variabili disponibili per la configurazione delle azioni downstream.

Per ulteriori informazioni sulle variabili in, consulta. CodePipeline Variables

BucketName

Il nome del bucket Amazon S3 relativo alla modifica dell'origine che ha attivato la pipeline.

ETag

Il tag entità per l'oggetto correlato alla modifica di origine che ha attivato la pipeline. L'ETag è un hash MD5 dell'oggetto. L'ETag riflette solo i cambiamenti ai contenuti di un oggetto, non i suoi metadata.

ObjectKey

Il nome della chiave oggetto Amazon S3 correlato alla modifica dell'origine che ha attivato la pipeline.

VersionId

L'ID di versione per la versione dell'oggetto correlata alla modifica di origine che ha attivato la pipeline.

Dichiarazione dell'operazione

YAML

```
Name: Source
Actions:
  - RunOrder: 1
    OutputArtifacts:
      - Name: SourceArtifact
    ActionTypeId:
      Provider: S3
      Owner: AWS
      Version: '1'
      Category: Source
    Region: us-west-2
    Name: Source
    Configuration:
      S3Bucket: my-bucket-oregon
      S3ObjectKey: my-application.zip
      PollForSourceChanges: 'false'
    InputArtifacts: []
```

JSON

```
{
    "Name": "Source",
    "Actions": [
        {
            "RunOrder": 1,
            "OutputArtifacts": [
                {
                    "Name": "SourceArtifact"
                }
            ],
            "ActionTypeId": {
                "Provider": "S3",
                "Owner": "AWS",
```

```
                "Version": "1",
                "Category": "Source"
            },
            "Region": "us-west-2",
            "Name": "Source",
            "Configuration": {
                "S3Bucket": "my-bucket-oregon",
                "S3ObjectKey": "my-application.zip",
                "PollForSourceChanges": "false"
            },
            "InputArtifacts": []
        }
    ]
},
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Tutorial: creazione di una semplice pipeline (bucket S3)— Questo tutorial fornisce un esempio di file di specifiche dell'app e un esempio di CodeDeploy applicazione e gruppo di distribuzione. Usa questo tutorial per creare una pipeline con un sorgente Amazon S3 da distribuire su istanze Amazon EC2.

AWS AppConfig

AWS AppConfig è una capacità di AWS Systems Manager. AppConfig supporta implementazioni controllate su applicazioni di qualsiasi dimensione e include controlli di convalida e monitoraggio integrati. Puoi utilizzarlo AppConfig con applicazioni ospitate su istanze Amazon EC2, container AWS Lambda, applicazioni mobili o dispositivi IoT.

L'azione di `AppConfig` distribuzione è un' AWS CodePipeline azione che distribuisce le configurazioni archiviate nella posizione di origine della pipeline in un' AppConfig applicazione, un ambiente e un profilo di configurazione specifici. Utilizza le preferenze definite in una strategia di distribuzione. AppConfig

Tipo di operazione

- Categoria: `Deploy`

- Proprietario: `AWS`
- Provider: `AppConfig`
- Versione: `1`

Parametri di configurazione

Applicazione

Campo obbligatorio: sì

L'ID dell' AWS AppConfig applicazione con i dettagli per la configurazione e la distribuzione.

Ambiente

Campo obbligatorio: sì

L'ID dell' AWS AppConfig ambiente in cui viene distribuita la configurazione.

ConfigurationProfile

Campo obbligatorio: sì

L'ID del profilo di AWS AppConfig configurazione da distribuire.

InputArtifactConfigurationPath

Campo obbligatorio: sì

Il percorso del file dei dati di configurazione all'interno dell'elemento di input da distribuire.

DeploymentStrategy

Campo obbligatorio: no

La strategia AWS AppConfig di distribuzione da utilizzare per la distribuzione.

Input artifact (Artefatti di input)

- Numero di artefatti: 1
- Descrizione: l'artefatto di input per l'azione di distribuzione.

Artefatti di output

Non applicabile.

Esempio di configurazione dell'operazione

YAML

```
name: Deploy
actions:
  - name: Deploy
    actionTypeId:
      category: Deploy
      owner: AWS
      provider: AppConfig
      version: '1'
    runOrder: 1
    configuration:
      Application: 2s2qv57
      ConfigurationProfile: PvjrpU
      DeploymentStrategy: frqt7ir
      Environment: 9tm27yd
      InputArtifactConfigurationPath: /
    outputArtifacts: []
    inputArtifacts:
      - name: SourceArtifact
    region: us-west-2
    namespace: DeployVariables
```

JSON

```
{
    "name": "Deploy",
    "actions": [
        {
            "name": "Deploy",
            "actionTypeId": {
                "category": "Deploy",
                "owner": "AWS",
                "provider": "AppConfig",
                "version": "1"
            },
```

```
            "runOrder": 1,
            "configuration": {
                "Application": "2s2qv57",
                "ConfigurationProfile": "PvjrpU",
                "DeploymentStrategy": "frqt7ir",
                "Environment": "9tm27yd",
                "InputArtifactConfigurationPath": "/"
            },
            "outputArtifacts": [],
            "inputArtifacts": [
                {
                    "name": "SourceArtifact"
                }
            ],
            "region": "us-west-2",
            "namespace": "DeployVariables"
        }
    ]
}
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- AWS AppConfig— Per informazioni sulle AWS AppConfig distribuzioni, consulta la Guida per l'utente.AWS Systems Manager
- Tutorial: crea una pipeline da utilizzare AWS AppConfig come provider di distribuzione— Questo tutorial consente di iniziare a configurare semplici file e AppConfig risorse di configurazione della distribuzione e mostra come utilizzare la console per creare una pipeline con un' AWS AppConfig azione di distribuzione.

AWS CloudFormation

Esegue un'operazione su uno AWS CloudFormation stack. Uno stack è una raccolta di AWS risorse che è possibile gestire come singola unità. Le risorse di uno stack sono definite dal modello AWS CloudFormation dello stack. Un set di modifiche crea un confronto che può essere visualizzato senza modificare lo stack originale. Per informazioni sui tipi di AWS CloudFormation azioni che possono essere eseguite su pile e set di modifiche, consulta il `ActionMode` parametro.

Per creare un messaggio di errore per un' AWS CloudFormation azione in cui un'operazione di stack non è riuscita, CodePipeline chiama l' AWS CloudFormation `DescribeStackEvents`API. Se un ruolo IAM di Action è autorizzato ad accedere a quell'API, i dettagli sulla prima risorsa fallita verranno inclusi nel CodePipeline messaggio di errore. Altrimenti, se la policy relativa al ruolo non dispone dell'autorizzazione appropriata, CodePipeline ignorerà l'accesso all'API e mostrerà invece un messaggio di errore generico. A tale scopo, è necessario aggiungere l'`cloudformation:DescribeStackEvents`autorizzazione al ruolo di servizio o ad altri ruoli IAM per la pipeline.

Se non desideri che i dettagli delle risorse vengano visualizzati nei messaggi di errore della pipeline, puoi revocare questa autorizzazione per il ruolo IAM dell'azione rimuovendo l'autorizzazione. `cloudformation:DescribeStackEvents`

Argomenti

Tipo di operazione

- Categoria: `Deploy`
- Proprietario: `AWS`
- Provider: `CloudFormation`
- Versione: `1`

Parametri di configurazione

ActionMode

Campo obbligatorio: sì

`ActionMode`è il nome dell'azione AWS CloudFormation eseguita su uno stack o un set di modifiche. Sono disponibili le modalità operazione seguenti:

- `CHANGE_SET_EXECUTE` esegue un set di modifiche per lo stack di risorse basato su un set di aggiornamenti delle risorse specificati. Con questa azione, AWS CloudFormation inizia a modificare lo stack.
- `CHANGE_SET_REPLACE` crea il set di modifiche, se non esiste, in base al nome dello stack e al modello che invii. Se il set di modifiche esiste, lo AWS CloudFormation elimina e ne crea uno nuovo.
- `CREATE_UPDATE` crea lo stack, se non esiste. Se lo stack esiste, lo AWS CloudFormation aggiorna. Utilizza questa operazione per aggiornare gli stack esistenti. Al contrario`REPLACE_ON_FAILURE`, se lo stack esiste e si trova in uno stato di errore, CodePipeline non eliminerà e sostituirà lo stack.
- `DELETE_ONLY` elimina uno stack. Se specifichi uno stack che non esiste, l'operazione viene completata senza l'eliminazione di uno stack.
- `REPLACE_ON_FAILURE` crea uno stack, se non esiste. Se lo stack esiste e si trova in uno stato di errore, AWS CloudFormation elimina lo stack e quindi ne crea uno nuovo. Se lo stack non è in uno stato di errore, lo aggiorna. AWS CloudFormation

  Lo stack si trova nello stato non riuscito quando uno dei seguenti tipi di stato viene visualizzato in AWS CloudFormation:

  - `ROLLBACK_FAILED`
  - `CREATE_FAILED`
  - `DELETE_FAILED`
  - `UPDATE_ROLLBACK_FAILED`

  Utilizza questa operazione per sostituire automaticamente gli stack non riusciti senza effettuarne il ripristino o eseguire la relativa risoluzione di problemi.

  > **Important**
  >
  > Ti consigliamo di utilizzare `REPLACE_ON_FAILURE` solo a scopo di test perché potrebbe eliminare lo stack.

StackName

Campo obbligatorio: sì

`StackName` è il nome di uno stack esistente o che desideri creare.

Funzionalità

Obbligatorio: condizionale

L'utilizzo di `Capabilities` conferma che il modello potrebbe disporre delle capacità per creare e aggiornare automaticamente alcune risorse e che queste funzionalità sono determinate in base ai tipi di risorse nel modello.

Questa proprietà è obbligatoria se disponi di risorse IAM nel modello di stack o crei uno stack direttamente da un modello contenente macro. Affinché l' AWS CloudFormation azione funzioni correttamente in questo modo, è necessario riconoscere esplicitamente che si desidera che venga eseguita con una delle seguenti funzionalità:

- `CAPABILITY_IAM`
- `CAPABILITY_NAMED_IAM`
- `CAPABILITY_AUTO_EXPAND`

Puoi specificare più funzionalità utilizzando una virgola (nessuno spazio) tra le funzionalità. L'esempio in Dichiarazione dell'operazione mostra una voce con entrambe le proprietà CAPABILITY_IAM e CAPABILITY_AUTO_EXPAND.

Per ulteriori informazioni in merito`Capabilities`, consulta le proprietà UpdateStacknella sezione AWS CloudFormation API Reference.

ChangeSetName

Obbligatorio: condizionale

`ChangeSetName` il nome di un set di modifiche esistente o di uno nuovo che desideri creare per lo stack specificato.

Questa proprietà è obbligatoria per le seguenti modalità di operazione: CHANGE_SET_REPLACE e CHANGE_SET_EXECUTE. La proprietà viene ignorata per tutte le altre modalità operazione.

RoleArn

Obbligatorio: condizionale

`RoleArn` è l'ARN del ruolo del servizio IAM che viene assunto da AWS CloudFormation quando utilizza risorse nello stack specificato. `RoleArn` non viene applicato durante l'esecuzione di un set

di modifiche. Se non lo utilizzate CodePipeline per creare il set di modifiche, assicuratevi che al set o allo stack di modifiche sia associato un ruolo.

> **Note**
>
> Questo ruolo deve trovarsi nello stesso account del ruolo per l'azione in esecuzione, come configurato nella dichiarazione `RoleArn` dell'azione.

Questa proprietà è obbligatoria per le seguenti modalità operazione:

- CREATE_UPDATE
- REPLACE_ON_FAILURE
- DELETE_ONLY
- CHANGE_SET_REPLACE

> **Note**
>
> AWS CloudFormation al modello viene assegnato un URL con firma S3; pertanto, `RoleArn` non è necessaria l'autorizzazione per accedere al bucket di artefatti. Tuttavia, l'azione `RoleArn` richiede l'autorizzazione per accedere al bucket di artefatti, per generare l'URL firmato.

TemplatePath

Obbligatorio: condizionale

`TemplatePath`rappresenta il file modello. AWS CloudFormation Includi il file in un artefatto di input per questa operazione. Il nome del file segue questo formato:

`Artifactname``::`*`TemplateFileName`*

`Artifactname`è il nome dell'artefatto di input così come appare in. CodePipeline Ad esempio, una fase di origine con il nome di artefatto di `SourceArtifact` e un nome file di `template-export.json` crea un nome `TemplatePath` come mostrato in questo esempio:

```
"TemplatePath": "SourceArtifact::template-export.json"
```

Questa proprietà è obbligatoria per le seguenti modalità operazione:

- CREATE_UPDATE
- REPLACE_ON_FAILURE
- CHANGE_SET_REPLACE

La proprietà viene ignorata per tutte le altre modalità operazione.

> **Note**
>
> Il file AWS CloudFormation modello contenente il corpo del modello ha una lunghezza minima di 1 byte e una lunghezza massima di 1 MB. Per le azioni AWS CloudFormation di distribuzione in CodePipeline, la dimensione massima dell'artefatto di input è sempre 256 MB. Per ulteriori informazioni, consulta Quote in AWS CodePipeline e Limiti di AWS CloudFormation.

OutputFileName

Campo obbligatorio: no

`OutputFileName`Utilizzatelo per specificare un nome di file di output, ad esempio`CreateStackOutput.json`, da CodePipeline aggiungere all'artefatto di output della pipeline per questa azione. Il file JSON contiene il contenuto della `Outputs` sezione dello stack. AWS CloudFormation

Se non specificate un nome, CodePipeline non genera un file o un artefatto di output.

ParameterOverrides

Campo obbligatorio: no

I parametri sono definiti nel modello di stack e consentono di fornire valori per gli stessi al momento della creazione o dell'aggiornamento dello stack. Puoi utilizzare un oggetto JSON per impostare i valori dei parametri nel modello. Questi valori sostituiscono quelli impostati nel file di configurazione del modello. Per ulteriori informazioni sull'utilizzo delle sostituzioni dei parametri, consulta Proprietà di configurazione (oggetto JSON).

Ti consigliamo di utilizzare il file di configurazione del modello per la maggior parte dei valori dei parametri. Utilizza le sostituzioni dei parametri solo per i valori che non sono noti finché la pipeline

non è in esecuzione. Per ulteriori informazioni, vedete Using Parameter Override Functions with CodePipeline Pipelines nella Guida per l'utente.AWS CloudFormation

> **Note**
>
> Tutti i nomi dei parametri devono essere presenti nel modello di stack.

TemplateConfiguration

Campo obbligatorio: no

`TemplateConfiguration` è il file di configurazione del modello. Includi il file in un artefatto di input per questa operazione. Può contenere i valori di parametro del modello e una policy stack. Per ulteriori informazioni sul formato del file di configurazione del modello, vedete AWS CloudFormation Artifacts.

Il nome del file di configurazione del modello segue questo formato:

`Artifactname``::`*`TemplateConfigurationFileName`*

`Artifactname`è il nome dell'artefatto di input così come appare in. CodePipeline Ad esempio, una fase di origine con il nome di artefatto di `SourceArtifact` e un nome file di `test-configuration.json` crea un nome `TemplateConfiguration` come mostrato in questo esempio:

```
"TemplateConfiguration": "SourceArtifact::test-configuration.json"
```

Input artifact (Artefatti di input)

- Numero di artefatti: `0 to 10`
- Descrizione: come input, l' AWS CloudFormation azione accetta facoltativamente artefatti per i seguenti scopi:
  - Per fornire il file di modello dello stack da eseguire. Consulta il parametro `TemplatePath`.
  - Per fornire il file di configurazione del modello da utilizzare. Consulta il parametro `TemplateConfiguration`. Per ulteriori informazioni sul formato del file di configurazione del modello, vedere Artifacts.AWS CloudFormation

- Fornire l'artefatto per una funzione Lambda da distribuire come parte dello stack. AWS CloudFormation

Artefatti di output

- Numero di artefatti: `0 to 1`
- Descrizione: se il parametro `OutputFileName` è specificato, esiste un artefatto di output prodotto da questa operazione che contiene un file JSON con il nome specificato. Il file JSON contiene i contenuti della sezione output dello stack AWS CloudFormation .

  Per ulteriori informazioni sulla sezione output che puoi creare per l'operazione AWS CloudFormation , consulta Output.

Variabili di output

Quando è configurata, questa azione produce variabili che possono essere referenziate dalla configurazione dell'azione di un'azione downstream nella pipeline. È possibile configurare un'azione con uno spazio dei nomi per rendere tali variabili disponibili per la configurazione delle azioni downstream.

Per AWS CloudFormation le azioni, le variabili vengono prodotte a partire da qualsiasi valore indicato nella `Outputs` sezione di un modello di pila. Tieni presente che le uniche modalità di CloudFormation azione che generano output sono quelle che comportano la creazione o l'aggiornamento di uno stack, come la creazione dello stack, gli aggiornamenti dello stack e l'esecuzione dei set di modifiche. Le modalità di operazione corrispondenti che generano variabili sono:

- `CHANGE_SET_EXECUTE`
- `CHANGE_SET_REPLACE`
- `CREATE_UPDATE`
- `REPLACE_ON_FAILURE`

Per ulteriori informazioni, consulta Variables. Per un tutorial che mostra come creare una pipeline con un'azione di CloudFormation distribuzione in una pipeline che utilizza variabili di output, consulta. CloudFormation Tutorial: crea una pipeline che utilizza le variabili delle azioni di AWS CloudFormation distribuzione

Dichiarazione dell'operazione

YAML

```
Name: ExecuteChangeSet
ActionTypeId:
  Category: Deploy
  Owner: AWS
  Provider: CloudFormation
  Version: '1'
RunOrder: 2
Configuration:
  ActionMode: CHANGE_SET_EXECUTE
  Capabilities: CAPABILITY_NAMED_IAM,CAPABILITY_AUTO_EXPAND
  ChangeSetName: pipeline-changeset
  ParameterOverrides: '{"ProjectId": "my-project","CodeDeployRole":
 "CodeDeploy_Role_ARN"}'
  RoleArn: CloudFormation_Role_ARN
  StackName: my-project--lambda
  TemplateConfiguration: 'my-project--BuildArtifact::template-configuration.json'
  TemplatePath: 'my-project--BuildArtifact::template-export.yml'
OutputArtifacts: []
InputArtifacts:
  - Name: my-project-BuildArtifact
```

JSON

```
{
    "Name": "ExecuteChangeSet",
    "ActionTypeId": {
        "Category": "Deploy",
        "Owner": "AWS",
        "Provider": "CloudFormation",
        "Version": "1"
    },
    "RunOrder": 2,
    "Configuration": {
        "ActionMode": "CHANGE_SET_EXECUTE",
        "Capabilities": "CAPABILITY_NAMED_IAM,CAPABILITY_AUTO_EXPAND",
        "ChangeSetName": "pipeline-changeset",
        "ParameterOverrides": "{\"ProjectId\": \"my-project\",\"CodeDeployRole\":
 \"CodeDeploy_Role_ARN\"}",
        "RoleArn": "CloudFormation_Role_ARN",
```

```
        "StackName": "my-project--lambda",
        "TemplateConfiguration": "my-project--BuildArtifact::template-
configuration.json",
        "TemplatePath": "my-project--BuildArtifact::template-export.yml"
    },
    "OutputArtifacts": [],
    "InputArtifacts": [
        {
             "Name": "my-project-BuildArtifact"
        }
    ]
},
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Riferimento alle proprietà di configurazione: questo capitolo di riferimento della Guida per l'AWS CloudFormation utente fornisce ulteriori descrizioni ed esempi per questi CodePipeline parametri.
- AWS CloudFormation Riferimento API: il CreateStackparametro nel riferimento AWS CloudFormation API descrive i parametri dello stack per i AWS CloudFormation modelli.

AWS CloudFormation StackSets

CodePipeline offre la possibilità di eseguire AWS CloudFormation StackSets operazioni come parte del processo CI/CD. È possibile utilizzare un set di stack per creare pile in AWS account di diverse AWS regioni utilizzando un unico modello. AWS CloudFormation Tutte le risorse incluse in ogni stack sono definite dal modello del set di stack. AWS CloudFormation Quando create il set di stack, specificate il modello da utilizzare, nonché tutti i parametri e le funzionalità richiesti dal modello.

Per ulteriori informazioni sui concetti di AWS CloudFormation StackSets, consulta StackSets i concetti nella Guida per l'AWS CloudFormation utente.

Puoi integrare la tua pipeline AWS CloudFormation StackSets attraverso due tipi di azioni distinti che usi insieme:

- L'`CloudFormationStackSet`azione crea o aggiorna un set di stack o istanze di stack dal modello memorizzato nella posizione di origine della pipeline. Ogni volta che uno stack set viene

creato o aggiornato, avvia una distribuzione di tali modifiche su istanze specifiche. Nella console, puoi scegliere il provider di azioni CloudFormation Stack Set quando crei o modifichi la tua pipeline.

- L'`CloudFormationStackInstances`azione distribuisce le modifiche dall'`CloudFormationStackSet`azione alle istanze specificate, crea nuove istanze dello stack e definisce le sostituzioni dei parametri alle istanze specificate. Nella console, puoi scegliere il provider di azioni CloudFormation Stack Instances quando modifichi una pipeline esistente.

Puoi utilizzare queste azioni per eseguire la distribuzione AWS sugli account di destinazione o sugli ID delle unità organizzative AWS Organizations di destinazione.

> **Note**
>
> Per distribuire gli account o gli ID AWS delle unità organizzative di destinazione e utilizzare il modello di autorizzazioni gestite dal servizio, è necessario abilitare l'accesso affidabile tra e Organizzazioni. AWS CloudFormation StackSets AWS Per ulteriori informazioni, consulta Enabling trusted access with Stacksets. AWS CloudFormation

Argomenti

Come funzionano le AWS CloudFormation StackSets azioni

Un'`CloudFormationStackSet`azione crea o aggiorna risorse a seconda che l'azione venga eseguita per la prima volta.

L'`CloudFormationStackSet`azione crea o aggiorna il set di stack e distribuisce tali modifiche in istanze specifiche.

> **Note**
>
> Se si utilizza questa azione per effettuare un aggiornamento che include l'aggiunta di istanze dello stack, le nuove istanze vengono distribuite per prime e l'aggiornamento viene completato per ultime. Le nuove istanze ricevono prima la versione precedente, quindi l'aggiornamento viene applicato a tutte le istanze.

- Crea: quando non viene specificata alcuna istanza e lo stack set non esiste, l'CloudFormationStackSetazione crea il set di stack senza creare alcuna istanza.
- Aggiornamento: quando l'CloudFormationStackSetazione viene eseguita per un set di stack già creato, l'azione aggiorna il set di stack. Se non viene specificata alcuna istanza e lo stack set esiste già, tutte le istanze vengono aggiornate. Se questa azione viene utilizzata per aggiornare istanze specifiche, tutte le istanze rimanenti passano allo stato OBSOLETO.

  È possibile utilizzare l'CloudFormationStackSetazione per aggiornare lo stack set nei seguenti modi.

  - Aggiorna il modello su alcune o tutte le istanze.
  - Aggiorna i parametri su alcune o tutte le istanze.
  - Aggiorna il ruolo di esecuzione per lo stack set (deve corrispondere al ruolo di esecuzione specificato nel ruolo Amministratore).
  - Modifica il modello di autorizzazioni (solo se non sono state create istanze).
  - Abilita/disabilita `AutoDeployment` se il modello di permessi del set di stack è. `Service Managed`
  - Agisci come amministratore delegato in un account membro se il modello di autorizzazioni dello stack set lo è. `Service Managed`
  - Aggiorna il ruolo di amministratore.
  - Aggiorna la descrizione sullo stack set.
  - Aggiungi obiettivi di distribuzione all'aggiornamento del set di stack per creare nuove istanze dello stack.

L'`CloudFormationStackInstances`azione crea nuove istanze dello stack o aggiorna le istanze dello stack obsolete. Un'istanza diventa obsoleta quando un set di stack viene aggiornato, ma non tutte le istanze al suo interno vengono aggiornate.

- Crea: se lo stack esiste già, l'`CloudFormationStackInstances`azione aggiorna solo le istanze e non crea istanze dello stack.
- Aggiornamento: dopo l'esecuzione dell'`CloudFormationStackSet`azione, se il modello o i parametri sono stati aggiornati solo in alcuni casi, il resto verrà contrassegnato. `OUTDATED` Nelle fasi successive della pipeline, `CloudFormationStackInstances` aggiorna il resto delle istanze dello stack in ondate in modo che tutte le istanze siano contrassegnate. `CURRENT` Questa azione può essere utilizzata anche per aggiungere istanze aggiuntive o sostituire parametri su istanze nuove o esistenti.

Come parte di un aggiornamento, le `CloudFormationStackInstances` azioni `CloudFormationStackSet` and possono specificare nuovi obiettivi di distribuzione, che creano nuove istanze stack.

Come parte di un aggiornamento, le `CloudFormationStackInstances` azioni `CloudFormationStackSet` and non eliminano set di stack, istanze o risorse. Quando l'azione aggiorna uno stack ma non specifica tutte le istanze da aggiornare, le istanze che non erano state specificate per l'aggiornamento vengono rimosse dall'aggiornamento e impostate su uno stato di. `OUTDATED`

Durante una distribuzione, le istanze dello stack possono anche mostrare lo stato `OUTDATED` se la distribuzione sulle istanze non è riuscita.

Come strutturare le StackSets azioni in una pipeline

Come best practice, è consigliabile costruire la pipeline in modo che lo stack set venga creato e inizialmente distribuito su un sottoinsieme o su una singola istanza. Dopo aver testato la distribuzione e visualizzato lo stack set generato, aggiungete l'`CloudFormationStackInstances`azione in modo che le istanze rimanenti vengano create e aggiornate.

Utilizza la console o la CLI per creare la struttura di pipeline consigliata come segue:

1. Crea una pipeline con un'azione di origine (obbligatoria) e l'`CloudFormationStackSet`azione come azione di distribuzione. Esegui la tua pipeline.
2. Quando la pipeline viene eseguita per la prima volta, l'`CloudFormationStackSet`azione crea il set di stack e almeno un'istanza iniziale. Verifica la creazione del set di stack ed esamina la distribuzione sull'istanza iniziale. Ad esempio, per la creazione iniziale dello stack set per l'account Account-A dove si `us-east-1` trova la regione specificata, l'istanza dello stack viene creata con lo stack set:

Istanza Stack	Regione	Stato
StackInstanceID-1	us-east-1	CURRENT

3. Modifica la pipeline da aggiungere `CloudFormationStackInstances` come seconda azione di distribuzione per creare/aggiornare le istanze dello stack per gli obiettivi designati. Ad esempio, per la creazione di istanze stack per un account `Account-A` in cui sono specificate le `eu-central-1` regioni `us-east-2` e, vengono create le istanze dello stack rimanenti e l'istanza iniziale rimane aggiornata come segue:

Istanza Stack	Regione	Stato
StackInstanceID-1	us-east-1	CURRENT
StackInstanceID-2	us-east-2	CURRENT
StackInstanceID-3	eu-central-1	CURRENT

4. Esegui la pipeline secondo necessità per aggiornare il set di stack e aggiornare o creare istanze dello stack.

Quando si avvia un aggiornamento dello stack in cui sono stati rimossi gli obiettivi di distribuzione dalla configurazione dell'azione, le istanze dello stack che non erano destinate all'aggiornamento vengono rimosse dalla distribuzione e passano allo stato OBSOLETO. Ad esempio, per l'aggiornamento dell'istanza dello stack per l'account `Account-A` in cui la `us-east-2` regione viene rimossa dalla configurazione dell'azione, vengono create le istanze dello stack rimanenti e l'istanza rimossa viene impostata su OBSOLETA come segue:

Istanza Stack	Regione	Stato
StackInstanceID-1	us-east-1	CURRENT
StackInstanceID-2	us-east-2	ANTIQUATO
StackInstanceID-3	eu-central-1	CURRENT

Per ulteriori informazioni sulle migliori pratiche per la distribuzione dei set di stack, consulta la sezione Best practice StackSets nella Guida per l'AWS CloudFormation utente.

L'operazione **`CloudFormationStackSet`**

Questa azione crea o aggiorna uno stack set a partire dal modello memorizzato nella posizione di origine della pipeline.

Dopo aver definito un set di stack, è possibile creare, aggiornare o eliminare gli stack negli account e nelle regioni di destinazione specificati nei parametri di configurazione. Durante la creazione, l'aggiornamento e l'eliminazione degli stack, è possibile specificare altre preferenze, come l'ordine delle regioni per le operazioni da eseguire, la percentuale di tolleranza agli errori oltre la quale le operazioni dello stack si interrompono e il numero di account in cui le operazioni vengono eseguite contemporaneamente sugli stack.

Un set di stack è una risorsa di livello regionale. Se si crea uno stack impostato in una AWS regione, non è possibile accedervi da altre regioni.

Quando questa azione viene utilizzata come azione di aggiornamento dello stack set, gli aggiornamenti allo stack non sono consentiti senza una distribuzione su almeno un'istanza dello stack.

Argomenti

Tipo di operazione

- Categoria: `Deploy`
- Proprietario: `AWS`
- Provider: `CloudFormationStackSet`
- Versione: `1`

Parametri di configurazione

StackSetName

Campo obbligatorio: sì

Il nome da associare al set di stack. Questo nome deve essere univoco nella regione in cui viene creato.

Il nome può contenere solo caratteri alfanumerici e trattini. Deve iniziare con un carattere alfabetico e contenere al massimo 128 caratteri.

Descrizione

Campo obbligatorio: no

Una descrizione del set di stack. Puoi usarlo per descrivere lo scopo del set di stack o altre informazioni pertinenti.

TemplatePath

Campo obbligatorio: sì

La posizione del modello che definisce le risorse nello stack set. Deve puntare a un modello con una dimensione massima di 460.800 byte.

Immettete il percorso del nome dell'artefatto di origine e del file modello nel formato`"InputArtifactName::TemplateFileName"`, come illustrato nell'esempio seguente.

```
SourceArtifact::template.txt
```

Parametri

Campo obbligatorio: no

Un elenco di parametri del modello per il set di stack da aggiornare durante una distribuzione.

Puoi fornire i parametri come elenco letterale o percorso di file:

- È possibile inserire i parametri nel seguente formato di sintassi abbreviata:.
  `ParameterKey=string,ParameterValue=string,UsePreviousValue=boolean,ResolvedV`
  `ParameterKey=string,ParameterValue=string,UsePreviousValue=boolean,ResolvedV`
  Per ulteriori informazioni su questi tipi di dati, vedere. Tipi di dati dei parametri del modello

L'esempio seguente mostra un parametro denominato `BucketName` con il valore `my-bucket`.

```
ParameterKey=BucketName,ParameterValue=my-bucket
```

L'esempio seguente mostra una voce con più parametri:

```
ParameterKey=BucketName,ParameterValue=my-bucket
ParameterKey=Asset1,ParameterValue=true
ParameterKey=Asset2,ParameterValue=true
```

- È possibile immettere la posizione del file contenente un elenco di sostituzioni dei parametri del modello immesse nel formato`"InputArtifactName::ParametersFileName"`, come illustrato nell'esempio seguente.

  ```
  SourceArtifact::parameters.txt
  ```

  L'esempio seguente mostra il contenuto del file per. `parameters.txt`

  ```
  [
      {
          "ParameterKey": "KeyName",
          "ParameterValue": "true"
      },
      {
          "ParameterKey": "KeyName",
          "ParameterValue": "true"
      }
  ]
  ```

Funzionalità

Campo obbligatorio: no

Indica che il modello può creare e aggiornare risorse, a seconda dei tipi di risorse presenti nel modello.

È necessario utilizzare questa proprietà se nel modello di stack sono presenti risorse IAM o se si crea uno stack direttamente da un modello contenente macro. AWS CloudFormation

Affinché l'azione funzioni correttamente in questo modo, è necessario utilizzare una delle seguenti funzionalità:

- `CAPABILITY_IAM`
- `CAPABILITY_NAMED_IAM`

È possibile specificare più di una funzionalità utilizzando una virgola e senza spazi tra le funzionalità. L'esempio in Esempio di configurazione CloudFormationStackSetdell'azione mostra una voce con funzionalità multiple.

PermissionModel

Campo obbligatorio: no

Determina come vengono creati e gestiti i ruoli IAM. Se il campo non è specificato, viene utilizzato il valore predefinito. Per informazioni, consulta Modelli di autorizzazioni per le operazioni relative agli stack set.

I valori validi sono:

- `SELF_MANAGED`(impostazione predefinita): è necessario creare ruoli di amministratore ed esecuzione da distribuire agli account di destinazione.
- `SERVICE_MANAGED`: crea AWS CloudFormation StackSets automaticamente i ruoli IAM necessari per la distribuzione negli account gestiti da AWS Organizations. Ciò richiede che un account sia membro di un'organizzazione.

> **Note**
>
> Questo parametro può essere modificato solo quando non esistono istanze di stack nello stack set.

AdministrationRoleArn

> **Note**
>
> Poiché AWS CloudFormation StackSets esegue operazioni su più account, è necessario definire le autorizzazioni necessarie in tali account prima di poter creare lo stack set.

Campo obbligatorio: no

> **Note**
>
> Questo parametro è facoltativo per il modello di autorizzazioni SELF_MANAGED e non viene utilizzato per il modello di autorizzazioni SERVICE_MANAGED.

L'ARN del ruolo IAM nell'account amministratore utilizzato per eseguire operazioni di stack set.

Il nome può contenere caratteri alfanumerici, uno dei seguenti caratteri: _+=, .@- e senza spazi. Il nome non fa distinzione tra maiuscole e minuscole. Questo nome di ruolo deve avere una lunghezza minima di 20 caratteri e una lunghezza massima di 2048 caratteri. I nomi dei ruoli devono essere univoci all'interno dell'account. Il nome del ruolo specificato qui deve essere un nome di ruolo esistente. Se non si specifica il nome del ruolo, viene impostato su AWSCloudFormationStackSetAdministrationRole. Se si specifica ServiceManaged, non è necessario definire un nome di ruolo.

ExecutionRoleName

> **Note**
>
> Poiché AWS CloudFormation StackSets esegue operazioni su più account, è necessario definire le autorizzazioni necessarie in tali account prima di poter creare lo stack set.

Campo obbligatorio: no

> **Note**
>
> Questo parametro è facoltativo per il modello di autorizzazioni SELF_MANAGED e non viene utilizzato per il modello di autorizzazioni SERVICE_MANAGED.

Il nome del ruolo IAM negli account di destinazione utilizzati per eseguire le operazioni di stack set. Il nome può contenere caratteri alfanumerici, uno dei seguenti caratteri: _+=, .@- e senza spazi. Il nome non fa distinzione tra maiuscole e minuscole. Questo nome di ruolo deve avere una lunghezza minima di 1 carattere e una lunghezza massima di 64 caratteri. I nomi dei ruoli devono essere univoci all'interno dell'account. Il nome del ruolo specificato qui deve essere un nome di ruolo esistente. Non specificare questo ruolo se si utilizzano ruoli di esecuzione personalizzati. Se non si specifica il nome del ruolo, viene impostato

suAWSCloudFormationStackSetExecutionRole. Se si imposta Service_Managed su true, non è necessario definire un nome di ruolo.

OrganizationsAutoDeployment

Campo obbligatorio: no

> **Note**
>
> Questo parametro è facoltativo per il modello di autorizzazioni SERVICE_MANAGED e non viene utilizzato per il modello di autorizzazioni SELF_MANAGED.

Descrive se AWS CloudFormation StackSets viene distribuito automaticamente agli account AWS Organizations aggiunti a un'organizzazione o a un'unità organizzativa (OU) di destinazione. Se `OrganizationsAutoDeployment` è specificato, non specificare `DeploymentTargets` e`Regions`.

> **Note**
>
> Se non viene fornito alcun input`OrganizationsAutoDeployment`, il valore predefinito è`Disabled`.

I valori validi sono:

- `Enabled`. Obbligatorio: No.

  StackSets distribuisce automaticamente istanze stack aggiuntive agli account AWS Organizations che vengono aggiunti a un'organizzazione o unità organizzativa (OU) di destinazione nelle regioni specificate. Se un account viene rimosso da un'organizzazione o unità organizzativa di destinazione, AWS CloudFormation StackSets elimina le istanze dello stack dall'account nelle regioni specificate.

- `Disabled`. Obbligatorio: No.

  StackSets non distribuisce automaticamente istanze stack aggiuntive agli account AWS Organizations che vengono aggiunti a un'organizzazione o unità organizzativa (OU) di destinazione nelle regioni specificate.

- `EnabledWithStackRetention`. Obbligatorio: No.

Le risorse dello stack vengono conservate quando un account viene rimosso da un'organizzazione o unità organizzativa di destinazione.

DeploymentTargets

Campo obbligatorio: no

> **Note**
>
> Per il modello di autorizzazioni SERVICE_MANAGED, è possibile fornire l'ID radice dell'organizzazione o gli ID delle unità organizzative per gli obiettivi di distribuzione. Per il modello di autorizzazioni SELF_MANAGED, puoi fornire solo account.

> **Note**
>
> Quando questo parametro è selezionato, è necessario selezionare anche Regioni.

Un elenco di AWS account o ID di unità organizzative in cui è necessario creare/aggiornare le istanze dello stack set.

- Account:

  È possibile fornire gli account come elenco letterale o percorso di file:

  - Letterale: immettete i parametri nel formato sintattico abbreviato`account_ID,account_ID`, come illustrato nell'esempio seguente.

    ```
    111111222222,333333444444
    ```

  - Percorso del file: la posizione del file contenente un elenco di AWS account in cui le istanze dello stack set devono essere create/aggiornate, inserita nel formato. `InputArtifactName::AccountsFileName` Se si utilizza il percorso del file per specificare gli account oppure OrganizationalUnitIds, il formato del file deve essere in JSON, come illustrato nell'esempio seguente.

    ```
    SourceArtifact::accounts.txt
    ```

    L'esempio seguente mostra il contenuto del file per`accounts.txt`.

```
[
    "111111222222"
]
```

L'esempio seguente mostra il contenuto del file per `accounts.txt` quando si elencano più di un account:

```
[
    "111111222222","333333444444"
]
```

- OrganizationalUnitIds:

> **Note**
>
> Questo parametro è facoltativo per il modello di autorizzazioni SERVICE_MANAGED e non viene utilizzato per il modello di autorizzazioni SELF_MANAGED. Non utilizzarlo se si seleziona. OrganizationsAutoDeployment

Le unità AWS organizzative in cui aggiornare le istanze dello stack associate.

Puoi fornire gli ID delle unità organizzative come elenco letterale o percorso di file:

- Letterale: inserisci una matrice di stringhe separate da virgole, come illustrato nell'esempio seguente.

```
ou-examplerootid111-exampleouid111,ou-examplerootid222-exampleouid222
```

- Percorso del file: la posizione del file contenente un elenco OrganizationalUnitIds in cui creare o aggiornare le istanze dello stack set. Se si utilizza il percorso del file per specificare gli account oppure OrganizationalUnitIds, il formato del file deve essere in JSON, come illustrato nell'esempio seguente.

  Immettete il percorso del file nel formato`InputArtifactName::OrganizationalUnitIdsFileName`.

```
SourceArtifact::OU-IDs.txt
```

L'esempio seguente mostra il contenuto del file per`OU-IDs.txt`:

```
[
    "ou-examplerootid111-exampleouid111","ou-examplerootid222-exampleouid222"
]
```

Regioni

Campo obbligatorio: no

> **Note**
>
> Quando questo parametro è selezionato, è necessario selezionare anche DeploymentTargets.

Un elenco di AWS regioni in cui vengono create o aggiornate le istanze dello stack set. Le regioni vengono aggiornate nell'ordine in cui vengono inserite.

Immettete un elenco di AWS regioni valide nel formato`Region1,Region2`, come illustrato nell'esempio seguente.

```
us-west-2,us-east-1
```

FailureTolerancePercentage

Campo obbligatorio: no

La percentuale di account per regione per i quali questa operazione di stack può fallire prima che l' AWS CloudFormation operazione venga interrotta in quella regione. Se l'operazione viene interrotta in una regione, AWS CloudFormation non tenta l'operazione nelle regioni successive. Quando si calcola il numero di conti in base alla percentuale specificata, viene AWS CloudFormation arrotondato per difetto al numero intero successivo.

MaxConcurrentPercentage

Campo obbligatorio: no

La percentuale massima di account in cui eseguire questa operazione simultaneamente. Quando si calcola il numero di conti in base alla percentuale specificata, AWS CloudFormation arrotonda per difetto al numero intero successivo. Se l'arrotondamento per difetto dà come risultato zero,

AWS CloudFormation imposta invece il numero a uno. Sebbene si utilizzi questa impostazione per specificare il valore massimo, per le distribuzioni di grandi dimensioni il numero effettivo di account su cui si agisce contemporaneamente potrebbe essere inferiore a causa della limitazione del servizio.

RegionConcurrencyType

Campo obbligatorio: no

È possibile specificare se lo stack set deve essere distribuito in Regioni AWS sequenza o in parallelo configurando il parametro Region concurrency deployment. Quando viene specificata la concorrenza Region per distribuire stack su più stack in Regioni AWS parallelo, ciò può comportare tempi di implementazione complessivi più rapidi.

- Parallelo: le implementazioni di set di stack verranno eseguite contemporaneamente, a condizione che gli errori di distribuzione in una regione non superino una tolleranza di errore specificata.
- Sequenziale: le implementazioni di set di stack verranno eseguite una alla volta, a condizione che gli errori di distribuzione in una regione non superino una tolleranza di errore specificata. La distribuzione sequenziale è la selezione predefinita.

ConcurrencyMode

Campo obbligatorio: no

La modalità di concorrenza consente di scegliere il comportamento del livello di concorrenza durante le operazioni di stack set, con una tolleranza agli errori rigorosa o morbida. La tolleranza rigorosa ai guasti riduce la velocità di implementazione quando si verificano errori nelle operazioni del set di stack, poiché la simultaneità diminuisce per ogni errore. Soft Failure Tolerance dà priorità alla velocità di implementazione sfruttando al contempo le funzionalità di sicurezza. AWS CloudFormation

- `STRICT_FAILURE_TOLERANCE`: Questa opzione riduce dinamicamente il livello di concorrenza per garantire che il numero di account falliti non superi mai una particolare tolleranza di errore. Questo è il comportamento che segue di default.
- `SOFT_FAILURE_TOLERANCE`: questa opzione disaccoppia la tolleranza agli errori dalla concorrenza effettiva. Ciò consente alle operazioni di stack set di essere eseguite a un livello di concorrenza prestabilito, indipendentemente dal numero di errori.

CallAs

Campo obbligatorio: no

> **Note**
>
> Questo parametro è facoltativo per il modello di `SERVICE_MANAGED` autorizzazioni e non viene utilizzato per il modello di autorizzazioni. `SELF_MANAGED`

Speciifica se si agisce nell'account di gestione dell'organizzazione o come amministratore delegato in un account membro.

> **Note**
>
> Se questo parametro è impostato su`DELEGATED_ADMIN`, assicurati che il ruolo IAM della pipeline disponga dell'autorizzazione. `organizations:ListDelegatedAdministrators` In caso contrario, l'azione avrà esito negativo durante l'esecuzione con un errore simile al seguente:`Account used is not a delegated administrator`.

- `SELF`: La distribuzione di Stack Set utilizzerà le autorizzazioni gestite dal servizio una volta effettuato l'accesso all'account di gestione.
- `DELEGATED_ADMIN`: la distribuzione di Stack Set utilizzerà le autorizzazioni gestite dal servizio una volta effettuato l'accesso a un account amministratore delegato.

Input artifact (Artefatti di input)

È necessario includere almeno un elemento di input che contenga il modello per lo stack set in un'azione. `CloudFormationStackSet` È possibile includere più artefatti di input per elenchi di obiettivi, account e parametri di distribuzione.

- Numero di artefatti: `1 to 3`
- Descrizione: puoi includere artefatti per fornire:
  - Il file modello dello stack. Consulta il parametro `TemplatePath`.
  - Il file dei parametri. Consulta il parametro `Parameters`.
  - Il file degli account. Consulta il parametro `DeploymentTargets`.

Artefatti di output

- Numero di artefatti: `0`
- Descrizione: gli artefatti di output non si applicano a questo tipo di azione.

Variabili di output

Se si configura questa azione, produce variabili a cui può fare riferimento la configurazione dell'azione di un'azione a valle nella pipeline. È possibile configurare un'azione con uno spazio dei nomi per rendere tali variabili disponibili per la configurazione delle azioni downstream.

- StackSetId: L'ID dello stack set.
- OperationId: L'ID dell'operazione di set di stack.

Per ulteriori informazioni, consulta Variables.

Esempio di configurazione CloudFormationStackSetdell'azione

Gli esempi seguenti mostrano la configurazione dell'CloudFormationStackSetazione.

Esempio di modello di autorizzazioni autogestite

L'esempio seguente mostra un'CloudFormationStackSetazione in cui l'obiettivo di distribuzione immesso è un ID AWS account.

YAML

```
Name: CreateStackSet
ActionTypeId:
  Category: Deploy
  Owner: AWS
  Provider: CloudFormationStackSet
  Version: '1'
RunOrder: 1
Configuration:
  DeploymentTargets: '111111222222'
  FailureTolerancePercentage: '20'
  MaxConcurrentPercentage: '25'
  PermissionModel: SELF_MANAGED
  Regions: us-east-1
  StackSetName: my-stackset
```

```
  TemplatePath: 'SourceArtifact::template.json'
OutputArtifacts: []
InputArtifacts:
  - Name: SourceArtifact
Region: us-west-2
Namespace: DeployVariables
```

JSON

```
{
    "Name": "CreateStackSet",
    "ActionTypeId": {
        "Category": "Deploy",
        "Owner": "AWS",
        "Provider": "CloudFormationStackSet",
        "Version": "1"
    },
    "RunOrder": 1,
    "Configuration": {
        "DeploymentTargets": "111111222222",
        "FailureTolerancePercentage": "20",
        "MaxConcurrentPercentage": "25",
        "PermissionModel": "SELF_MANAGED",
        "Regions": "us-east-1",
        "StackSetName": "my-stackset",
        "TemplatePath": "SourceArtifact::template.json"
    },
    "OutputArtifacts": [],
    "InputArtifacts": [
        {
            "Name": "SourceArtifact"
        }
    ],
    "Region": "us-west-2",
    "Namespace": "DeployVariables"
}
```

Esempio del modello di autorizzazioni gestite dal servizio

L'esempio seguente mostra un'CloudFormationStackSetazione per il modello di autorizzazioni gestite dal servizio in cui l'opzione per la distribuzione automatica in AWS Organizations è abilitata con stack retention.

YAML

```
Name: Deploy
ActionTypeId:
  Category: Deploy
  Owner: AWS
  Provider: CloudFormationStackSet
  Version: '1'
RunOrder: 1
Configuration:
  Capabilities: 'CAPABILITY_IAM,CAPABILITY_NAMED_IAM'
  OrganizationsAutoDeployment: EnabledWithStackRetention
  PermissionModel: SERVICE_MANAGED
  StackSetName: stacks-orgs
  TemplatePath: 'SourceArtifact::template.json'
OutputArtifacts: []
InputArtifacts:
  - Name: SourceArtifact
Region: eu-central-1
Namespace: DeployVariables
```

JSON

```
{
    "Name": "Deploy",
    "ActionTypeId": {
        "Category": "Deploy",
        "Owner": "AWS",
        "Provider": "CloudFormationStackSet",
        "Version": "1"
    },
    "RunOrder": 1,
    "Configuration": {
        "Capabilities": "CAPABILITY_IAM,CAPABILITY_NAMED_IAM",
        "OrganizationsAutoDeployment": "EnabledWithStackRetention",
        "PermissionModel": "SERVICE_MANAGED",
        "StackSetName": "stacks-orgs",
        "TemplatePath": "SourceArtifact::template.json"
    },
    "OutputArtifacts": [],
    "InputArtifacts": [
        {
            "Name": "SourceArtifact"
```

```
        }
    ],
    "Region": "eu-central-1",
    "Namespace": "DeployVariables"
}
```

L'operazione CloudFormationStackInstances

Questa azione crea nuove istanze e distribuisce set di stack su istanze specifiche. Un'istanza di stack è un riferimento a uno stack in un account target all'interno di una Regione. Un'istanza dello stack può esistere senza uno stack; ad esempio, se la creazione dello stack non ha esito positivo, l'istanza dello stack mostra il motivo dell'errore di creazione dello stack. Un'istanza di stack è associata a un solo set di stack.

Dopo la creazione iniziale di un set di stack, puoi aggiungere nuove istanze di stack utilizzando. `CloudFormationStackInstances` I valori dei parametri del modello possono essere sovrascritti a livello di istanza dello stack durante le operazioni di creazione o aggiornamento dell'istanza dello stack set.

Ogni set di stack ha un modello e un set di parametri di modello. Quando aggiorni il modello o i parametri del modello, li aggiorni per l'intero set. Quindi tutti gli stati dell'istanza vengono impostati su `OUTDATED` fino a quando le modifiche non vengono distribuite a quell'istanza.

Per sovrascrivere i valori dei parametri su istanze specifiche, ad esempio, se il modello contiene un parametro per `stage` con un valore di`prod`, è possibile sovrascrivere il valore di quel parametro come o. `beta` `gamma`

Argomenti

- Tipo di operazione
- Parametri di configurazione
- Input artifact (Artefatti di input)
- Artefatti di output
- Variabili di output
- Esempio di configurazione dell'operazione

Tipo di operazione

- Categoria: `Deploy`
- Proprietario: `AWS`
- Provider: `CloudFormationStackInstances`
- Versione: `1`

Parametri di configurazione

StackSetName

Campo obbligatorio: sì

Il nome da associare al set di stack. Questo nome deve essere univoco nella regione in cui è stato creato.

Il nome può contenere solo caratteri alfanumerici e trattini. Deve iniziare con un carattere alfabetico e contenere al massimo 128 caratteri.

DeploymentTargets

Campo obbligatorio: no

> **Note**
>
> Per il modello di autorizzazioni SERVICE_MANAGED, è possibile fornire l'ID radice dell'organizzazione o gli ID delle unità organizzative per gli obiettivi di distribuzione. Per il modello di autorizzazioni SELF_MANAGED, puoi fornire solo account.

> **Note**
>
> Quando questo parametro è selezionato, è necessario selezionare anche Regioni.

Un elenco di AWS account o ID di unità organizzative in cui è necessario creare/aggiornare le istanze dello stack set.

- Account:

È possibile fornire gli account come elenco letterale o percorso di file:

- Letterale: immettete i parametri nel formato sintattico abbreviato`account_ID,account_ID`, come illustrato nell'esempio seguente.

```
111111222222,333333444444
```

- Percorso del file: la posizione del file contenente un elenco di AWS account in cui le istanze dello stack set devono essere create/aggiornate, inserita nel formato. `InputArtifactName::AccountsFileName` Se si utilizza il percorso del file per specificare gli account oppure OrganizationalUnitIds, il formato del file deve essere in JSON, come illustrato nell'esempio seguente.

```
SourceArtifact::accounts.txt
```

L'esempio seguente mostra il contenuto del file per`accounts.txt`:

```
[
    "111111222222"
]
```

L'esempio seguente mostra il contenuto del file per `accounts.txt` quando si elencano più di un account:

```
[
    "111111222222","333333444444"
]
```

- OrganizationalUnitIds:

> **Note**
>
> Questo parametro è facoltativo per il modello di autorizzazioni SERVICE_MANAGED e non viene utilizzato per il modello di autorizzazioni SELF_MANAGED. Non utilizzarlo se si seleziona. OrganizationsAutoDeployment

Le unità AWS organizzative in cui aggiornare le istanze dello stack associate.

È possibile fornire gli ID delle unità organizzative come elenco letterale o percorso di file.

- Letterale: inserisci una matrice di stringhe separate da virgole, come illustrato nell'esempio seguente.

```
ou-examplerootid111-exampleouid111,ou-examplerootid222-exampleouid222
```

- Percorso del file: la posizione del file contenente un elenco OrganizationalUnitIds in cui creare o aggiornare le istanze dello stack set. Se si utilizza il percorso del file per specificare gli account oppure OrganizationalUnitIds, il formato del file deve essere in JSON, come illustrato nell'esempio seguente.

  Immettete il percorso del file nel formato`InputArtifactName::OrganizationalUnitIdsFileName`.

```
SourceArtifact::OU-IDs.txt
```

  L'esempio seguente mostra il contenuto del file per`OU-IDs.txt`:

```
[
    "ou-examplerootid111-exampleouid111","ou-examplerootid222-exampleouid222"
]
```

Regioni

Campo obbligatorio: sì

> **Note**
>
> Quando questo parametro è selezionato, è necessario selezionare anche DeploymentTargets.

Un elenco di AWS regioni in cui vengono create o aggiornate le istanze dello stack set. Le regioni vengono aggiornate nell'ordine in cui vengono inserite.

Immettete un elenco di AWS regioni valide nel formato:`Region1,Region2`, come illustrato nell'esempio seguente.

```
us-west-2,us-east-1
```

ParameterOverrides

Campo obbligatorio: no

Un elenco di parametri dello stack set che desideri sovrascrivere nelle istanze dello stack selezionate. I valori dei parametri sostituiti vengono applicati a tutte le istanze dello stack negli account e nelle regioni specificati.

Puoi fornire i parametri come elenco letterale o percorso di file:

- È possibile inserire i parametri nel seguente formato di sintassi abbreviata:. `ParameterKey=string,ParameterValue=string,UsePreviousValue=boolean,ResolvedV ParameterKey=string,ParameterValue=string,UsePreviousValue=boolean,ResolvedV` Per ulteriori informazioni su questi tipi di dati, vedere. Tipi di dati dei parametri del modello

  L'esempio seguente mostra un parametro denominato `BucketName` con il valore `my-bucket`.

  ```
  ParameterKey=BucketName,ParameterValue=my-bucket
  ```

  L'esempio seguente mostra una voce con più parametri.

  ```
  ParameterKey=BucketName,ParameterValue=my-bucket
  ParameterKey=Asset1,ParameterValue=true
  ParameterKey=Asset2,ParameterValue=true
  ```

- È possibile immettere la posizione del file contenente un elenco di sostituzioni dei parametri del modello immesse nel formato`InputArtifactName::ParameterOverridessFileName`, come illustrato nell'esempio seguente.

  ```
  SourceArtifact::parameter-overrides.txt
  ```

  L'esempio seguente mostra il contenuto del file per. `parameter-overrides.txt`

  ```
  [
      {
          "ParameterKey": "KeyName",
          "ParameterValue": "true"
      },
      {
  ```

```
        "ParameterKey": "KeyName",
        "ParameterValue": "true"
    }
]
```

FailureTolerancePercentage

Campo obbligatorio: no

La percentuale di account per regione per i quali questa operazione di stack può fallire prima dell' AWS CloudFormation interruzione dell'operazione in quella regione. Se l'operazione viene interrotta in una regione, AWS CloudFormation non tenta l'operazione nelle regioni successive. Quando si calcola il numero di conti in base alla percentuale specificata, viene AWS CloudFormation arrotondato per difetto al numero intero successivo.

MaxConcurrentPercentage

Campo obbligatorio: no

La percentuale massima di account su cui eseguire questa operazione contemporaneamente. Quando si calcola il numero di conti in base alla percentuale specificata, viene AWS CloudFormation arrotondato per difetto al numero intero successivo. Se l'arrotondamento per difetto dà come risultato zero, AWS CloudFormation imposta invece il numero a uno. Sebbene si specifichi il valore massimo, per le distribuzioni di grandi dimensioni il numero effettivo di account su cui si agisce contemporaneamente potrebbe essere inferiore a causa della limitazione del servizio.

RegionConcurrencyType

Campo obbligatorio: no

È possibile specificare se lo stack set deve essere distribuito in Regioni AWS sequenza o in parallelo configurando il parametro region concurrency deployment. Quando viene specificata la concorrenza Region per distribuire stack su più stack in Regioni AWS parallelo, ciò può comportare tempi di implementazione complessivi più rapidi.

- Parallelo: le implementazioni di set di stack verranno eseguite contemporaneamente, a condizione che gli errori di distribuzione in una regione non superino una tolleranza di errore specificata.
- Sequenziale: le implementazioni di set di stack verranno eseguite una alla volta, a condizione che gli errori di distribuzione in una regione non superino una tolleranza di errore specificata. La distribuzione sequenziale è la selezione predefinita.

ConcurrencyMode

Campo obbligatorio: no

La modalità di concorrenza consente di scegliere il comportamento del livello di concorrenza durante le operazioni di stack set, con una tolleranza agli errori rigorosa o morbida. La tolleranza rigorosa ai guasti riduce la velocità di implementazione quando si verificano errori nelle operazioni del set di stack, poiché la simultaneità diminuisce per ogni errore. Soft Failure Tolerance dà priorità alla velocità di implementazione sfruttando al contempo le funzionalità di sicurezza. AWS CloudFormation

- `STRICT_FAILURE_TOLERANCE`: Questa opzione riduce dinamicamente il livello di concorrenza per garantire che il numero di account falliti non superi mai una particolare tolleranza di errore. Questo è il comportamento che segue di default.
- `SOFT_FAILURE_TOLERANCE`: questa opzione disaccoppia la tolleranza agli errori dalla concorrenza effettiva. Ciò consente alle operazioni di stack set di essere eseguite a un livello di concorrenza prestabilito, indipendentemente dal numero di errori.

CallAs

Campo obbligatorio: no

> **Note**
>
> Questo parametro è facoltativo per il modello di `SERVICE_MANAGED` autorizzazioni e non viene utilizzato per il modello di autorizzazioni. `SELF_MANAGED`

Speciifica se si agisce nell'account di gestione dell'organizzazione o come amministratore delegato in un account membro.

> **Note**
>
> Se questo parametro è impostato su`DELEGATED_ADMIN`, assicurati che il ruolo IAM della pipeline disponga dell'autorizzazione. `organizations:ListDelegatedAdministrators` In caso contrario, l'azione avrà esito negativo durante l'esecuzione con un errore simile al seguente:`Account used is not a delegated administrator`.

- `SELF`: La distribuzione di Stack Set utilizzerà le autorizzazioni gestite dal servizio una volta effettuato l'accesso all'account di gestione.
- `DELEGATED_ADMIN`: la distribuzione di Stack Set utilizzerà le autorizzazioni gestite dal servizio una volta effettuato l'accesso a un account amministratore delegato.

Input artifact (Artefatti di input)

`CloudFormationStackInstances`può contenere artefatti che elencano gli obiettivi e i parametri di distribuzione.

- Numero di artefatti: `0 to 2`
- Descrizione: come input, l'azione stack set accetta facoltativamente artefatti per i seguenti scopi:
  - Per fornire il file dei parametri da utilizzare. Consulta il parametro `ParameterOverrides`.
  - Per fornire il file degli account di destinazione da utilizzare. Consulta il parametro `DeploymentTargets`.

Artefatti di output

- Numero di artefatti: `0`
- Descrizione: gli artefatti di output non si applicano a questo tipo di azione.

Variabili di output

Quando è configurata, questa azione produce variabili che possono essere referenziate dalla configurazione dell'azione di un'azione downstream nella pipeline. È possibile configurare un'azione con uno spazio dei nomi per rendere tali variabili disponibili per la configurazione delle azioni downstream.

- StackSetId: L'ID dello stack set.
- OperationId: L'ID dell'operazione di set di stack.

Per ulteriori informazioni, consulta Variables.

Esempio di configurazione dell'operazione

Gli esempi seguenti mostrano la configurazione dell'CloudFormationStackInstancesazione.

Esempio di modello di autorizzazioni autogestite

L'esempio seguente mostra un'CloudFormationStackInstancesazione in cui l'obiettivo di distribuzione immesso è un Account AWS ID. 111111222222

YAML

```
Name: my-instances
ActionTypeId:
  Category: Deploy
  Owner: AWS
  Provider: CloudFormationStackInstances
  Version: '1'
RunOrder: 2
Configuration:
  DeploymentTargets: '111111222222'
  Regions: 'us-east-1,us-east-2,us-west-1,us-west-2'
  StackSetName: my-stackset
OutputArtifacts: []
InputArtifacts:
  - Name: SourceArtifact
Region: us-west-2
```

JSON

```
{
    "Name": "my-instances",
    "ActionTypeId": {
        "Category": "Deploy",
        "Owner": "AWS",
        "Provider": "CloudFormationStackInstances",
        "Version": "1"
    },
    "RunOrder": 2,
    "Configuration": {
        "DeploymentTargets": "111111222222",
        "Regions": "us-east-1,us-east-2,us-west-1,us-west-2",
        "StackSetName": "my-stackset"
    },
    "OutputArtifacts": [],
    "InputArtifacts": [
        {
            "Name": "SourceArtifact"
```

```
        }
    ],
    "Region": "us-west-2"
}
```

Esempio del modello di autorizzazioni gestite dal servizio

L'esempio seguente mostra un'CloudFormationStackInstancesazione per il modello di autorizzazioni gestite dal servizio in cui l'obiettivo di distribuzione è un ID di unità organizzativa AWS Organizations. `ou-1111-1example`

YAML

```
Name: Instances
ActionTypeId:
  Category: Deploy
  Owner: AWS
  Provider: CloudFormationStackInstances
  Version: '1'
RunOrder: 2
Configuration:
  DeploymentTargets: ou-1111-1example
  Regions: us-east-1
  StackSetName: my-stackset
OutputArtifacts: []
InputArtifacts:
  - Name: SourceArtifact
Region: eu-central-1
```

JSON

```
{
    "Name": "Instances",
    "ActionTypeId": {
        "Category": "Deploy",
        "Owner": "AWS",
        "Provider": "CloudFormationStackInstances",
        "Version": "1"
    },
    "RunOrder": 2,
    "Configuration": {
        "DeploymentTargets": "ou-1111-1example",
```

```
        "Regions": "us-east-1",
        "StackSetName": "my-stackset"
    },
    "OutputArtifacts": [],
    "InputArtifacts": [
        {
            "Name": "SourceArtifact"
        }
    ],
    "Region": "eu-central-1"
}
```

Modelli di autorizzazioni per le operazioni relative agli stack set

Poiché AWS CloudFormation StackSets esegue operazioni su più account, è necessario definire le autorizzazioni necessarie in tali account prima di poter creare lo stack set. È possibile definire le autorizzazioni tramite autorizzazioni autogestite o autorizzazioni gestite dal servizio.

Con le autorizzazioni autogestite, puoi creare i due ruoli IAM richiesti da StackSets : un ruolo di amministratore, ad esempio nell'account AWSCloudFormationStackSetAdministrationRole in cui definisci il set di stack, e un ruolo di esecuzione come quello AWSCloudFormationStackSetExecutionRole in ciascuno degli account in cui distribuisci le istanze dello stack set. Utilizzando questo modello di autorizzazioni, StackSets puoi eseguire la distribuzione su qualsiasi AWS account in cui l'utente dispone delle autorizzazioni per creare un ruolo IAM. Per ulteriori informazioni, consulta Concedere autorizzazioni autogestite nella Guida per l'utente.AWS CloudFormation

> **Note**
>
> Poiché AWS CloudFormation StackSets esegue operazioni su più account, è necessario definire le autorizzazioni necessarie in tali account prima di poter creare lo stack set.

Con le autorizzazioni gestite dal servizio, puoi distribuire istanze stack su account gestiti da Organizations. AWS Utilizzando questo modello di autorizzazioni, non è necessario creare i ruoli IAM necessari, in quanto StackSets crea i ruoli IAM per conto dell'utente. Con questo modello, puoi anche abilitare le distribuzioni automatiche agli account che verranno aggiunti all'organizzazione in futuro. Vedi Enable trusted access with AWS Organizations nella Guida AWS CloudFormation per l'utente.

Tipi di dati dei parametri del modello

I parametri del modello utilizzati nelle operazioni di stack set includono i seguenti tipi di dati. Per ulteriori informazioni, vedere DescribeStackSet.

ParameterKey

- Descrizione: la chiave associata al parametro. Se non specificate una chiave e un valore per un particolare parametro, AWS CloudFormation utilizza il valore predefinito specificato nel modello.
- Esempio:

```
"ParameterKey=BucketName,ParameterValue=my-bucket"
```

ParameterValue

- Descrizione: il valore di input associato al parametro.
- Esempio:

```
"ParameterKey=BucketName,ParameterValue=my-bucket"
```

UsePreviousValue

- Durante un aggiornamento dello stack, utilizzate il valore del parametro esistente utilizzato dallo stack per una determinata chiave di parametro. Se lo specificate`true`, non specificate il valore di un parametro.
- Esempio:

```
"ParameterKey=Asset1,UsePreviousValue=true"
```

Ogni set di stack ha un modello e un set di parametri di modello. Quando aggiorni il modello o i parametri del modello, li aggiorni per l'intero set. Quindi tutti gli stati delle istanze vengono impostati su OBSOLETO fino a quando le modifiche non vengono distribuite a quell'istanza.

Per sovrascrivere i valori dei parametri su istanze specifiche, ad esempio, se il modello contiene un parametro per `stage` con un valore di`prod`, è possibile sovrascrivere il valore di quel parametro impostando or. `beta` `gamma`

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Tipi di parametri: questo capitolo di riferimento della Guida per l'AWS CloudFormation utente fornisce ulteriori descrizioni ed esempi per CloudFormation i parametri del modello.
- Procedure ottimali: per ulteriori informazioni sulle migliori pratiche per la distribuzione di set di stack, https://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/stacksets-bestpractices.html consulta la Guida per l'AWS CloudFormation utente.
- AWS CloudFormation Riferimento alle API: puoi fare riferimento CloudFormation alle seguenti azioni nell'AWS CloudFormation API Reference per ulteriori informazioni sui parametri utilizzati nelle operazioni relative agli stack set:

  - L'CreateStackSetazione crea un set di stack.
  - L'UpdateStackSetazione aggiorna il set di stack e le istanze di stack associate negli account e nelle regioni specificati. Anche se l'operazione di stack set creata aggiornando lo stack set fallisce (completamente o parzialmente, al di sotto o al di sopra di una tolleranza di errore specificata), il set di stack viene aggiornato con queste modifiche. Le CreateStackInstances chiamate successive sul set di stack specificato utilizzano il set di stack aggiornato.
  - L'CreateStackInstancesazione crea un'istanza stack per tutte le regioni specificate all'interno di tutti gli account specificati su un modello di autorizzazione autogestito o all'interno di tutti gli obiettivi di distribuzione specificati su un modello di autorizzazione gestito dal servizio. È possibile sovrascrivere i parametri per le istanze create da questa azione. Se le istanze esistono già, CreateStackInstances chiamate UpdateStackInstances con gli stessi parametri di input. Quando utilizzate questa azione per creare istanze, non modifica lo stato delle altre istanze dello stack.
  - L'UpdateStackInstancesazione aggiorna le istanze dello stack con lo stack impostato per tutte le regioni specificate all'interno di tutti gli account specificati su un modello di autorizzazione autogestito o entro tutti gli obiettivi di distribuzione specificati su un modello di autorizzazione gestito dal servizio. È possibile sovrascrivere i parametri per le istanze aggiornate da questa azione. Quando si utilizza questa azione per aggiornare un sottoinsieme di istanze, non modifica lo stato delle altre istanze dello stack.
  - L'DescribeStackSetOperationazione restituisce la descrizione dell'operazione di stack set specificata.
  - L'DescribeStackSetazione restituisce la descrizione del set di stack specificato.

AWS CodeBuild

Consente di eseguire compilazioni e test come parte della pipeline. Quando si esegue un'azione di CodeBuild compilazione o test, i comandi specificati nelle specifiche di compilazione vengono eseguiti all'interno di un CodeBuild contenitore. Tutti gli artefatti specificati come artefatti di input per un' CodeBuild azione sono disponibili all'interno del contenitore che esegue i comandi. CodeBuild può fornire un'azione di compilazione o di test. Per ulteriori informazioni, consulta la Guida per l'utente AWS CodeBuild.

Quando si utilizza la CodePipeline procedura guidata nella console per creare un progetto di compilazione, il progetto di CodeBuild compilazione mostra che il provider di origine lo è CodePipeline. Quando crei un progetto di compilazione nella CodeBuild console, non puoi specificarlo CodePipeline come fornitore di origine, ma l'aggiunta dell'azione di compilazione alla pipeline modifica il codice sorgente nella console. CodeBuild Per ulteriori informazioni, consulta l'AWS CodeBuild API ProjectSourceReference.

Argomenti

Tipo di operazione

- Categoria: `Build` o `Test`
- Proprietario: `AWS`
- Provider: `CodeBuild`
- Versione: `1`

Parametri di configurazione

ProjectName

Campo obbligatorio: sì

`ProjectName`è il nome del progetto di compilazione in CodeBuild.

PrimarySource

Obbligatorio: condizionale

Il valore del `PrimarySource` parametro deve essere il nome di uno degli elementi di input dell'azione. CodeBuild cerca il file build spec ed esegue i comandi build spec nella directory che contiene la versione decompressa di questo artefatto.

Questo parametro è obbligatorio se per un'azione vengono specificati più artefatti di input. CodeBuild Quando è presente un solo artefatto di origine per l'operazione, l'impostazione predefinita per l'artefatto `PrimarySource` è tale artefatto.

BatchEnabled

Campo obbligatorio: no

Il valore booleano del `BatchEnabled` parametro consente all'azione di eseguire più build nella stessa esecuzione di build.

Quando questa opzione è abilitata, l'`CombineArtifacts`opzione è disponibile.

Per esempi di pipeline con compilazioni in batch abilitate, consulta CodePipeline Integrazione con CodeBuild e compilazioni in batch.

CombineArtifacts

Campo obbligatorio: no

Il valore booleano del `CombineArtifacts` parametro combina tutti gli elementi di compilazione di una compilazione in batch in un unico file di artefatti per l'azione di compilazione.

Per utilizzare questa opzione, il parametro deve essere abilitato. `BatchEnabled`

EnvironmentVariables

Campo obbligatorio: no

Il valore di questo parametro viene utilizzato per impostare le variabili di ambiente per l' CodeBuild azione nella pipeline. Il valore per il parametro `EnvironmentVariables` assume la forma di un array JSON di oggetti variabili di ambiente. Vedi il parametro di esempio in Dichiarazione di operazione (esempio CodeBuild).

Ogni oggetto ha tre parti, tutte costituite da stringhe:

- `name`: il nome o la chiave della variabile di ambiente.
- `value`: il valore della variabile di ambiente. Quando si utilizza il `SECRETS_MANAGER` tipo `PARAMETER_STORE` o, questo valore deve essere il nome di un parametro già archiviato in AWS Systems Manager Parameter Store o un segreto già archiviato in AWS Secrets Manager, rispettivamente.

  > **Note**
  >
  > Sconsigliamo vivamente l'uso di variabili di ambiente per archiviare valori sensibili, in particolare AWS le credenziali. Quando si utilizza la CodeBuild console o la AWS CLI, le variabili di ambiente vengono visualizzate in testo semplice. Per i valori sensibili, si consiglia di utilizzare invece il tipo `SECRETS_MANAGER`.

- `type`: (facoltativo) il tipo di variabile di ambiente. I valori validi sono `PARAMETER_STORE`, `SECRETS_MANAGER` o `PLAINTEXT`. Se il valore non viene specificato, viene usato il valore predefinito `PLAINTEXT`.

> **Note**
>
> Quando inserite la `name` configurazione e `type` per le variabili di ambiente, specialmente se la variabile di ambiente contiene la sintassi della variabile di CodePipeline output, non superate il limite di 1000 caratteri per il campo del valore della configurazione. `value` Quando questo limite viene superato, viene restituito un errore di convalida.

Per ulteriori informazioni, consulta l'API EnvironmentVariableReference. AWS CodeBuild Per un esempio di CodeBuild azione con una variabile di ambiente che si risolve nel nome del GitHub ramo, vedi. Esempio: utilizzare una BranchName variabile con variabili di CodeBuild ambiente

Input artifact (Artefatti di input)

- Numero di artefatti: `1 to 5`
- Descrizione: CodeBuild cerca il file build spec ed esegue i comandi build spec dalla directory dell'artefatto sorgente primario. Quando viene specificata più di una fonte di input per l' CodeBuild azione, questo artefatto deve essere impostato utilizzando il parametro di configurazione dell'azione in. `PrimarySource` CodePipeline

  Ogni artefatto di input viene estratto nella propria directory, le cui posizioni sono archiviate in variabili di ambiente. La directory per l'artefatto di origine principale viene resa disponibile con `$CODEBUILD_SRC_DIR`. Le directory per tutti gli altri artefatti di input sono rese disponibili con `$CODEBUILD_SRC_DIR_yourInputArtifactName`.

> **Note**
>
> L'artefatto configurato nel CodeBuild progetto diventa l'artefatto di input utilizzato dall'azione nella pipeline. CodeBuild

Artefatti di output

- Numero di artefatti: `0 to 5`
- Descrizione: possono essere usati per rendere gli artefatti definiti nel file delle specifiche di CodeBuild build disponibili per le azioni successive nella pipeline. Quando viene definito un solo artefatto di output, questo può essere definito direttamente nella sezione `artifacts` del file delle specifiche di compilazione. Quando viene specificato più di un artefatto di output, tutti gli artefatti a cui si fa riferimento devono essere definiti come artefatti secondari nel file di specifiche di compilazione. I nomi degli artefatti di output in CodePipeline devono corrispondere agli identificatori degli artefatti nel file delle specifiche di build.

> **Note**
>
> L'artefatto configurato nel CodeBuild progetto diventa l'artefatto di input nell'azione della pipeline. CodePipeline

Se il `CombineArtifacts` parametro è selezionato per le build in batch, la posizione dell'artefatto di output contiene gli artefatti combinati di più build eseguite nella stessa esecuzione.

Variabili di output

Questa azione produrrà come variabili tutte le variabili di ambiente che sono state esportate come parte della build. Per maggiori dettagli su come esportare le variabili di ambiente, consulta la Guida API. EnvironmentVariableAWS CodeBuild

Per ulteriori informazioni sull'utilizzo delle variabili di CodeBuild ambiente in CodePipeline, consulta gli esempi inCodeBuild azioni (variabili di output). Per un elenco delle variabili di ambiente in cui è possibile utilizzare CodeBuild, consulta Variabili di ambiente negli ambienti di compilazione nella Guida per l'AWS CodeBuild utente.

Dichiarazione di operazione (esempio CodeBuild)

YAML

```
Name: Build
Actions:
  - Name: PackageExport
    ActionTypeId:
      Category: Build
      Owner: AWS
      Provider: CodeBuild
      Version: '1'
    RunOrder: 1
    Configuration:
      BatchEnabled: 'true'
      CombineArtifacts: 'true'
      ProjectName: my-build-project
      PrimarySource: MyApplicationSource1
      EnvironmentVariables:
 '[{"name":"TEST_VARIABLE","value":"TEST_VALUE","type":"PLAINTEXT"},
{"name":"ParamStoreTest","value":"PARAMETER_NAME","type":"PARAMETER_STORE"}]'
    OutputArtifacts:
      - Name: MyPipeline-BuildArtifact
    InputArtifacts:
      - Name: MyApplicationSource1
```

```
      - Name: MyApplicationSource2
```

JSON

```
{
    "Name": "Build",
    "Actions": [
        {
            "Name": "PackageExport",
            "ActionTypeId": {
                "Category": "Build",
                "Owner": "AWS",
                "Provider": "CodeBuild",
                "Version": "1"
            },
            "RunOrder": 1,
            "Configuration": {
                "BatchEnabled": "true",
                "CombineArtifacts": "true",
                "ProjectName": "my-build-project",
                "PrimarySource": "MyApplicationSource1",
                "EnvironmentVariables": "[{\"name\":\"TEST_VARIABLE\",\"value\":
\"TEST_VALUE\",\"type\":\"PLAINTEXT\"},{\"name\":\"ParamStoreTest\",\"value\":
\"PARAMETER_NAME\",\"type\":\"PARAMETER_STORE\"}]"
            },
            "OutputArtifacts": [
                {
                    "Name": "MyPipeline-BuildArtifact"
                }
            ],
            "InputArtifacts": [
                {
                    "Name": "MyApplicationSource1"
                },
                {
                    "Name": "MyApplicationSource2"
                }
            ]
        }
    ]
}
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- AWS CodeBuild Guida per l'utente: per un esempio di pipeline con un' CodeBuild azione, consulta Use CodePipeline with CodeBuild to Test Code e Run Builds. Per esempi di progetti con più CodeBuild artefatti di input e output, consulta CodePipelineIntegration with and Multiple Input Sources CodeBuild and Output Artifacts Sample e Multiple Input Sources and Output Artifacts Sample.
- Tutorial: crea una pipeline con cui creare e testare la tua app Android AWS Device Farm— Questo tutorial fornisce un esempio di file di specifiche di build e un'applicazione di esempio per creare una pipeline con un GitHub sorgente che crea e testa un'app Android con e. CodeBuild AWS Device Farm
- Riferimento alle specifiche di compilazione per CodeBuild : questo argomento di riferimento fornisce definizioni ed esempi per comprendere i file delle specifiche di CodeBuild compilazione. Per un elenco delle variabili di ambiente in cui è possibile utilizzare CodeBuild, consulta Variabili di ambiente negli ambienti di compilazione nella Guida per l'AWS CodeBuild utente.

CodeCommit

Avvia la pipeline quando viene effettuato un nuovo commit sul CodeCommit repository e sul ramo configurati.

Se utilizzi la console per creare o modificare la pipeline, CodePipeline crea una regola CodeCommit CloudWatch Events che avvia la pipeline quando si verifica una modifica nel repository.

È necessario aver già creato un CodeCommit repository prima di connettere la pipeline tramite un'azione. CodeCommit

Una volta rilevata una modifica del codice, sono disponibili le seguenti opzioni per passare il codice alle operazioni successive:

- Predefinito: configura l'azione di CodeCommit origine per generare un file ZIP con una copia superficiale del commit.
- Clone completo: configura l'azione di origine per inviare un riferimento URL Git al repository per le azioni successive.

Attualmente, il riferimento all'URL Git può essere utilizzato solo dalle CodeBuild azioni a valle per clonare il repository e i metadati Git associati. Il tentativo di passare un riferimento URL Git a non CodeBuild azioni genera un errore.

Argomenti

Tipo di operazione

- Categoria: `Source`
- Proprietario: `AWS`
- Provider: `CodeCommit`
- Versione: `1`

Parametri di configurazione

RepositoryName

Campo obbligatorio: sì

Il nome del repository in cui devono essere rilevate le modifiche di origine.

BranchName

Campo obbligatorio: sì

Il nome del ramo in cui devono essere rilevate le modifiche di origine.

PollForSourceChanges

Campo obbligatorio: no

`PollForSourceChanges`controlla se interroga CodePipeline il CodeCommit repository per verificare la presenza di modifiche all'origine. Ti consigliamo invece di utilizzare CloudWatch Events per rilevare le modifiche all'origine. Per ulteriori informazioni sulla configurazione CloudWatch degli eventi, consulta Migrazione delle pipeline di polling (CodeCommit origine) (CLI) oMigra le pipeline di polling (CodeCommit source) (modello)AWS CloudFormation.

> **Important**
>
> Se intendi configurare una regola CloudWatch Events, devi impostarla su `PollForSourceChanges` per `false` evitare esecuzioni duplicate della pipeline.

I valori validi per questo parametro sono:

- `true`: Se impostata, analizza il repository per CodePipeline verificare se sono state apportate modifiche all'origine.

  > **Note**
  >
  > Se si omette`PollForSourceChanges`, per CodePipeline impostazione predefinita esegue il polling del repository per verificare la presenza di modifiche all'origine. Questo comportamento è lo stesso se `PollForSourceChanges` è incluso e impostato su `true`.

- `false`: se impostata, CodePipeline non esegue il polling del repository per verificare la presenza di modifiche all'origine. Utilizzate questa impostazione se intendete configurare una regola CloudWatch Events per rilevare le modifiche all'origine.

OutputArtifactFormat

Campo obbligatorio: no

Il formato dell'artefatto di output. I valori possono essere uno o due. `CODEBUILD_CLONE_REF` `CODE_ZIP` Se non altrimenti specificato, l'impostazione predefinita è `CODE_ZIP`.

> **Important**
>
> L'`CODEBUILD_CLONE_REF`opzione può essere utilizzata solo dalle azioni a CodeBuild valle.
> Se scegli questa opzione, devi aggiungere l'`codecommit:GitPull`autorizzazione al tuo ruolo di CodeBuild servizio, come mostrato inAggiungi le CodeBuild GitClone

autorizzazioni per le azioni di origine CodeCommit. È inoltre necessario aggiungere l'`codecommit:GetRepository`autorizzazione al proprio ruolo CodePipeline di servizio, come mostrato inAggiunta delle autorizzazioni dal ruolo di servizio CodePipeline. Per un tutorial che mostra come usare l'opzione Full clone, vediTutorial: usa il clone completo con una sorgente di CodeCommit pipeline.

Input artifact (Artefatti di input)

- Numero di artefatti: `0`
- Descrizione: gli artefatti di input non si applicano a questo tipo di azione.

Artefatti di output

- Numero di artefatti: `1`
- Descrizione: l'artefatto di output di questa azione è un file ZIP che contiene il contenuto del repository configurato e del ramo al commit specificato come revisione di origine per l'esecuzione della pipeline. Gli artefatti generati dal repository sono gli artefatti di output dell'azione. CodeCommit L'ID di commit del codice sorgente viene visualizzato CodePipeline come revisione del codice sorgente per l'esecuzione della pipeline attivata.

Variabili di output

Quando è configurata, questa azione produce variabili che possono essere referenziate dalla configurazione dell'azione di un'azione downstream nella pipeline. Questa azione produce variabili che possono essere viste come variabili di output, anche se l'azione non ha uno spazio dei nomi. È possibile configurare un'azione con uno spazio dei nomi per rendere tali variabili disponibili per la configurazione delle azioni downstream.

Per ulteriori informazioni, consulta Variables.

CommitId

L'ID di CodeCommit commit che ha attivato l'esecuzione della pipeline. Gli ID di commit sono l'SHA completo del commit.

CommitMessage

Il messaggio di descrizione, se presente, associato al commit che ha attivato l'esecuzione della pipeline.

RepositoryName

Il nome del CodeCommit repository in cui è stato effettuato il commit che ha attivato la pipeline.

BranchName

Il nome del ramo del CodeCommit repository in cui è stata apportata la modifica all'origine.

AuthorDate

La data in cui il commit è stato creato, in formato timestamp.

Per ulteriori informazioni sulla differenza tra un autore e un committer in Git, vedere Visualizzazione della cronologia del commit in Pro Git di Scott Chacon e Ben Straub.

CommitterDate

La data in cui è stato eseguito il commit, in formato timestamp.

Per ulteriori informazioni sulla differenza tra un autore e un committer in Git, vedere Visualizzazione della cronologia del commit in Pro Git di Scott Chacon e Ben Straub.

Esempio di configurazione dell'operazione

Esempio di formato predefinito degli artefatti di output

YAML

```
Actions:
  - OutputArtifacts:
      - Name: Artifact_MyWebsiteStack
    InputArtifacts: []
    Name: source
    Configuration:
      RepositoryName: MyWebsite
      BranchName: main
      PollForSourceChanges: 'false'
    RunOrder: 1
```

```
    ActionTypeId:
      Version: '1'
      Provider: CodeCommit
      Category: Source
      Owner: AWS
  Name: Source
```

JSON

```
{
    "Actions": [
        {
            "OutputArtifacts": [
                {
                    "Name": "Artifact_MyWebsiteStack"
                }
            ],
            "InputArtifacts": [],
            "Name": "source",
            "Configuration": {
                "RepositoryName": "MyWebsite",
                "BranchName": "main",
                "PollForSourceChanges": "false"
            },
            "RunOrder": 1,
            "ActionTypeId": {
                "Version": "1",
                "Provider": "CodeCommit",
                "Category": "Source",
                "Owner": "AWS"
            }
        }
    ],
    "Name": "Source"
},
```

Esempio di formato degli artefatti di output del clone completo

YAML

```
name: Source
actionTypeId:
```

```
  category: Source
  owner: AWS
  provider: CodeCommit
  version: '1'
runOrder: 1
configuration:
  BranchName: main
  OutputArtifactFormat: CODEBUILD_CLONE_REF
  PollForSourceChanges: 'false'
  RepositoryName: MyWebsite
outputArtifacts:
  - name: SourceArtifact
inputArtifacts: []
region: us-west-2
namespace: SourceVariables
```

JSON

```
{
    "name": "Source",
    "actionTypeId": {
        "category": "Source",
        "owner": "AWS",
        "provider": "CodeCommit",
        "version": "1"
    },
    "runOrder": 1,
    "configuration": {
        "BranchName": "main",
        "OutputArtifactFormat": "CODEBUILD_CLONE_REF",
        "PollForSourceChanges": "false",
        "RepositoryName": "MyWebsite"
    },
    "outputArtifacts": [
        {
            "name": "SourceArtifact"
        }
    ],
    "inputArtifacts": [],
    "region": "us-west-2",
    "namespace": "SourceVariables"
}
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Tutorial: crea una pipeline semplice (CodeCommitrepository)— Questo tutorial fornisce un esempio di file di specifiche dell'app e un esempio di CodeDeploy applicazione e gruppo di distribuzione. Usa questo tutorial per creare una pipeline con un CodeCommit sorgente da distribuire su istanze Amazon EC2.

AWS CodeDeploy

Utilizzi un' AWS CodeDeploy azione per distribuire il codice dell'applicazione nella tua flotta di distribuzione. La tua flotta di distribuzione può essere composta da istanze Amazon EC2, istanze locali o entrambe.

> **Note**
>
> Questo argomento di riferimento descrive l'azione di CodeDeploy distribuzione per i paesi CodePipeline in cui la piattaforma di distribuzione è Amazon EC2. Per informazioni di riferimento sulle azioni di distribuzione di Amazon Elastic Container Service to CodeDeploy blue/green in CodePipeline, consulta. Amazon Elastic Container Service e CodeDeploy blu-verde

Argomenti

Tipo di operazione

- Categoria: `Deploy`

- Proprietario: `AWS`
- Provider: `CodeDeploy`
- Versione: `1`

Parametri di configurazione

ApplicationName

Campo obbligatorio: sì

Il nome dell'applicazione in cui hai creato. CodeDeploy

DeploymentGroupName

Campo obbligatorio: sì

Il gruppo di distribuzione in cui hai creato CodeDeploy.

Input artifact (Artefatti di input)

- Numero di artefatti: `1`
- Descrizione: Il AppSpec file CodeDeploy utilizzato per determinare:
  - Cosa installare sulle istanze dalla revisione dell'applicazione in Amazon S3 oppure. GitHub
  - Quali hook di eventi del ciclo di vita eseguire in risposta agli eventi del ciclo di vita della distribuzione.

Per ulteriori informazioni sul AppSpec file, consulta la sezione File Reference. CodeDeploy AppSpec

Artefatti di output

- Numero di artefatti: `0`
- Descrizione: gli artefatti di output non si applicano a questo tipo di azione.

Dichiarazione dell'operazione

YAML

```
Name: Deploy
Actions:
  - Name: Deploy
    ActionTypeId:
      Category: Deploy
      Owner: AWS
      Provider: CodeDeploy
      Version: '1'
    RunOrder: 1
    Configuration:
      ApplicationName: my-application
      DeploymentGroupName: my-deployment-group
    OutputArtifacts: []
    InputArtifacts:
      - Name: SourceArtifact
    Region: us-west-2
    Namespace: DeployVariables
```

JSON

```
{
    "Name": "Deploy",
    "Actions": [
        {
            "Name": "Deploy",
            "ActionTypeId": {
                "Category": "Deploy",
                "Owner": "AWS",
                "Provider": "CodeDeploy",
                "Version": "1"
            },
            "RunOrder": 1,
            "Configuration": {
                "ApplicationName": "my-application",
                "DeploymentGroupName": "my-deployment-group"
            },
            "OutputArtifacts": [],
            "InputArtifacts": [
                {
```

```
                    "Name": "SourceArtifact"
                }
            ],
            "Region": "us-west-2",
            "Namespace": "DeployVariables"
        }
    ]
},
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Tutorial: creazione di una semplice pipeline (bucket S3)— Questo tutorial illustra la creazione di un bucket di sorgenti, istanze EC2 e CodeDeploy risorse per distribuire un'applicazione di esempio. Quindi costruisci la tua pipeline con un'azione di CodeDeploy distribuzione che distribuisce il codice mantenuto nel tuo bucket S3 sulla tua istanza Amazon EC2.
- Tutorial: crea una pipeline semplice (CodeCommitrepository)— Questo tutorial ti guida attraverso la creazione del tuo repository di CodeCommit sorgenti, delle istanze EC2 e delle risorse per distribuire un'applicazione di esempio. CodeDeploy Quindi costruisci la tua pipeline con un'azione di CodeDeploy distribuzione che distribuisce il codice dal tuo CodeCommit repository alla tua istanza Amazon EC2.
- CodeDeploy AppSpec Riferimento ai file: questo capitolo di riferimento della Guida per l'AWS CodeDeploy utente fornisce informazioni di riferimento ed esempi di file. CodeDeploy AppSpec

CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite

Le azioni di origine per le connessioni sono supportate da. AWS CodeConnections CodeConnections consente di creare e gestire connessioni tra AWS risorse e repository di terze parti come GitHub. Avvia una pipeline quando viene effettuato un nuovo commit su un repository di codice sorgente di terze parti. L'azione source recupera le modifiche al codice quando una pipeline viene eseguita manualmente o quando un evento webhook viene inviato dal provider di origine.

Puoi configurare le azioni nella tua pipeline per utilizzare una configurazione Git che ti consenta di avviare la pipeline con i trigger. Per configurare la configurazione dei trigger della pipeline per filtrare con i trigger, vedi maggiori dettagli in. Filtra i trigger nelle richieste push o pull di codice

> **Note**
>
> Questa funzionalità non è disponibile nelle regioni Asia Pacifico (Hong Kong), Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Africa (Città del Capo), Medio Oriente (Bahrain), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna), Europa (Zurigo), Israele (Tel Aviv) o AWS GovCloud (Stati Uniti occidentali). Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

Le connessioni possono associare AWS le tue risorse ai seguenti repository di terze parti:

- Bitbucket Cloud (tramite l'opzione provider Bitbucket nella CodePipeline console o il provider `Bitbucket` nella CLI)

  > **Note**
  >
  > È possibile creare connessioni a un repository Bitbucket Cloud. I tipi di provider Bitbucket installati, ad esempio Bitbucket Server, non sono supportati.

- > **Note**
  >
  > Se utilizzi un'area di lavoro Bitbucket, devi disporre dell'accesso come amministratore per creare la connessione.

- GitHub ed GitHub Enterprise Cloud (tramite l'opzione provider GitHub (versione 2) nella CodePipeline console o il `GitHub` provider nella CLI)

  > **Note**
  >
  > Se il repository si trova in un' GitHub organizzazione, devi essere il proprietario dell'organizzazione per creare la connessione. Se utilizzi un repository che non fa parte di un'organizzazione, devi essere il proprietario del repository.

- GitHub Enterprise Server (tramite l'opzione provider GitHub Enterprise Server nella CodePipeline console o il `GitHub Enterprise Server` provider nella CLI)

- GitLab.com (tramite l'opzione GitLabprovider nella CodePipeline console o il `GitLab` provider nella CLI)

> **Note**
>
> È possibile creare connessioni a un repository in cui si ricopre il ruolo di proprietario e quindi la connessione può essere utilizzata con il repository con risorse come. GitLab CodePipeline Per i repository nei gruppi, non è necessario essere il proprietario del gruppo.

- Installazione gestita automaticamente per GitLab (Enterprise Edition o Community Edition) (tramite l'opzione provider GitLab autogestita nella CodePipeline console o il `GitLabSelfManaged` provider nella CLI)

> **Note**
>
> Ogni connessione supporta tutti gli archivi che hai con quel provider. Devi solo creare una nuova connessione per ogni tipo di provider.

Le connessioni consentono alla pipeline di rilevare le modifiche alla fonte tramite l'app di installazione del provider terzo. Ad esempio, i webhook vengono utilizzati per sottoscrivere tipi di GitHub eventi e possono essere installati su un'organizzazione, un repository o un'app. GitHub La tua connessione installa un webhook di archivio sull' GitHub app che si iscrive a eventi di tipo push. GitHub

Una volta rilevata una modifica del codice, sono disponibili le seguenti opzioni per passare il codice alle operazioni successive:

- Predefinito: come altre azioni di CodePipeline origine esistenti, `CodeStarSourceConnection` può generare un file ZIP con una copia superficiale del commit.
- Clone completo: `CodeStarSourceConnection` può anche essere configurato per inviare un riferimento URL al repository per le azioni successive.

  Attualmente, il riferimento all'URL Git può essere utilizzato solo dalle CodeBuild azioni a valle per clonare il repository e i metadati Git associati. Il tentativo di passare un riferimento URL Git a non CodeBuild azioni genera un errore.

CodePipeline ti chiede di aggiungere l'app di installazione AWS Connector al tuo account di terze parti quando crei una connessione. È necessario aver già creato l'account e l'archivio del provider di terze parti prima di poterti connettere tramite l'azione. `CodeStarSourceConnection`

> **Note**
>
> Per creare o allegare una policy al tuo ruolo con le autorizzazioni necessarie per utilizzare le AWS CodeStar connessioni, vedi Riferimento alle autorizzazioni di Connections. A seconda di quando è stato creato il ruolo di CodePipeline servizio, potrebbe essere necessario aggiornarne le autorizzazioni per supportare le connessioni. AWS CodeStar Per istruzioni, consulta Aggiunta delle autorizzazioni dal ruolo di servizio CodePipeline.

> **Note**
>
> Per utilizzare le connessioni in Europa (Milano) Regione AWS, devi:
>
> 1. Installare un'app specifica per la regione
> 2. Abilitare la regione
>
> Questa app specifica per la regione supporta i collegamenti nella regione Europa (Milano). È pubblicata sul sito del provider di terze parti ed è separata dall'app esistente che supporta le connessioni per altre regioni. Installando questa app, autorizzi i provider di terze parti a condividere i tuoi dati con il servizio solo per questa regione e puoi revocare le autorizzazioni in qualsiasi momento disinstallando l'app.
> Il servizio non elaborerà o memorizzerà i dati a meno che tu non abiliti la Regione. Abilitando questa regione, concedi al nostro servizio le autorizzazioni per elaborare e archiviare i dati. Anche se la regione non è abilitata, i provider di terze parti possono comunque condividere i tuoi dati con il nostro servizio se l'app specifica per la regione rimane installata, quindi assicurati di disinstallarla dopo aver disabilitato la regione. Per ulteriori informazioni, consulta Enabling a Region (Abilitare una regione).

Argomenti

- Input artifact (Artefatti di input)
- Artefatti di output
- Variabili di output
- Dichiarazione dell'operazione
- Installazione dell'app di installazione e creazione di una connessione
- Consulta anche

Tipo di operazione

- Categoria: `Source`
- Proprietario: `AWS`
- Provider: `CodeStarSourceConnection`
- Versione: `1`

Parametri di configurazione

ConnectionArn

Campo obbligatorio: sì

L'ARN della connessione configurato e autenticato per il provider di origine.

FullRepositoryId

Campo obbligatorio: sì

Il proprietario e il nome del repository in cui devono essere rilevate le modifiche di origine.

Esempio: `some-user/my-repo`

> **Important**
>
> È necessario utilizzare le maiuscole e minuscole corrette per il FullRepositoryIdvalore. Ad esempio, se il nome utente è `some-user` e il nome del repository è`My-Repo`, il valore consigliato di FullRepositoryIdè`some-user/My-Repo`.

BranchName

Campo obbligatorio: sì

Il nome del ramo in cui devono essere rilevate le modifiche di origine.

OutputArtifactFormat

Campo obbligatorio: no

Specifica il formato dell'artefatto di output. Può essere `CODEBUILD_CLONE_REF` o `CODE_ZIP`. Se non altrimenti specificato, l'impostazione predefinita è `CODE_ZIP`.

> **Important**
>
> L'`CODEBUILD_CLONE_REF`opzione può essere utilizzata solo dalle azioni a CodeBuild valle.
> Se scegli questa opzione, dovrai aggiornare le autorizzazioni per il tuo ruolo di CodeBuild Project Service come mostrato in. Aggiungi le autorizzazioni per le connessioni a Bitbucket, Enterprise Server o.com CodeBuild GitClone GitHub GitHub GitLab Per un tutorial che mostra come usare l'opzione Full clone, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline

DetectChanges

Campo obbligatorio: no

Controlla l'avvio automatico della pipeline quando viene effettuato un nuovo commit sul repository e sul ramo configurati. Se non specificato, il valore predefinito è `true` e il campo non viene visualizzato per impostazione predefinita. I valori validi per questo parametro sono:

- `true`: avvia CodePipeline automaticamente la pipeline con nuovi commit.
- `false`: CodePipeline non avvia la pipeline con nuovi commit.

Input artifact (Artefatti di input)

- Numero di artefatti: `0`
- Descrizione: gli artefatti di input non si applicano a questo tipo di azione.

Artefatti di output

- Numero di artefatti: 1
- Descrizione: gli artefatti generati dal repository sono gli artefatti di output per l'operazione `CodeStarSourceConnection`. L'ID di commit del codice sorgente viene visualizzato CodePipeline come revisione del codice sorgente per l'esecuzione della pipeline attivata. È possibile configurare l'artefatto di output di questa operazione:
  - Un file ZIP che contiene il contenuto del repository configurato e del ramo al commit specificato come revisione di origine per l'esecuzione della pipeline.
  - Un file JSON che contiene un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git.

> **Important**
>
> Questa opzione può essere utilizzata solo dalle CodeBuild azioni a valle.
> Se scegli questa opzione, dovrai aggiornare le autorizzazioni per il tuo ruolo di CodeBuild Project Service come mostrato in. Risoluzione dei problemi CodePipeline Per un tutorial che mostra come usare l'opzione Full clone, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline

Variabili di output

Quando è configurata, questa azione produce variabili che possono essere referenziate dalla configurazione dell'azione di un'azione downstream nella pipeline. Questa azione produce variabili che possono essere viste come variabili di output, anche se l'azione non ha uno spazio dei nomi. È possibile configurare un'azione con uno spazio dei nomi per rendere tali variabili disponibili per la configurazione delle azioni downstream.

Per ulteriori informazioni, consulta Variables.

AuthorDate

La data in cui il commit è stato creato, in formato timestamp.

BranchName

Il nome del ramo per il repository in cui è stata apportata la modifica dell'origine.

CommitId

L'ID di commit che ha attivato l'esecuzione della pipeline.

CommitMessage

Il messaggio di descrizione, se presente, associato al commit che ha attivato l'esecuzione della pipeline.

ConnectionArn

L'ARN della connessione configurato e autenticato per il provider di origine.

FullRepositoryName

Il nome del repository in cui è stato effettuato il commit che ha attivato la pipeline.

Dichiarazione dell'operazione

Nell'esempio seguente, l'elemento di output è impostato sul formato ZIP predefinito `CODE_ZIP` per la connessione con ARN. `arn:aws:codestar-connections:region:`*`account-id`*`:connection/`*`connection-id`*

YAML

```
Name: Source
Actions:
  - InputArtifacts: []
    ActionTypeId:
      Version: '1'
      Owner: AWS
      Category: Source
      Provider: CodeStarSourceConnection
    OutputArtifacts:
      - Name: SourceArtifact
    RunOrder: 1
    Configuration:
      ConnectionArn: "arn:aws:codestar-connections:region:account-
id:connection/connection-id"
      FullRepositoryId: "some-user/my-repo"
      BranchName: "main"
      OutputArtifactFormat: "CODE_ZIP"
    Name: ApplicationSource
```

JSON

```
{
    "Name": "Source",
    "Actions": [
        {
            "InputArtifacts": [],
            "ActionTypeId": {
                "Version": "1",
                "Owner": "AWS",
                "Category": "Source",
                "Provider": "CodeStarSourceConnection"
            },
            "OutputArtifacts": [
                {
                    "Name": "SourceArtifact"
                }
            ],
            "RunOrder": 1,
            "Configuration": {
                "ConnectionArn": "arn:aws:codestar-connections:region:account-
id:connection/connection-id",
                "FullRepositoryId": "some-user/my-repo",
                "BranchName": "main",
                "OutputArtifactFormat": "CODE_ZIP"
            },
            "Name": "ApplicationSource"
        }
    ]
},
```

Installazione dell'app di installazione e creazione di una connessione

La prima volta che usi la console per aggiungere una nuova connessione a un repository di terze parti, devi autorizzare CodePipeline l'accesso ai tuoi repository. È possibile selezionare o creare un'app di installazione che permette di connettersi all'account in cui è stato creato il repository di codice di terze parti.

Quando si utilizza AWS CLI o un AWS CloudFormation modello, è necessario fornire l'ARN di connessione di una connessione che è già stata sottoposta all'handshake di installazione. In caso contrario, la pipeline non viene attivata.

> **Note**
>
> Per un'azione di `CodeStarSourceConnection` origine, non è necessario impostare un webhook o impostare come impostazione predefinita il polling. L'azione connessioni gestisce automaticamente il rilevamento delle modifiche alla fonte.

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- AWS::CodeStarConnections::Connection— Il riferimento al AWS CloudFormation modello per la risorsa AWS CodeStar Connections fornisce parametri ed esempi per le connessioni nei AWS CloudFormation modelli.
- AWS CodeStarRiferimento all'API AWS CodeStar Connections: il riferimento all'API Connections fornisce informazioni di riferimento per le azioni di connessione disponibili.
- Per visualizzare i passaggi per la creazione di una pipeline con azioni di origine supportate dalle connessioni, consulta quanto segue:
  - Per Bitbucket Cloud, utilizza l'opzione Bitbucket nella console o l'azione nella CLI`CodestarSourceConnection`. Per informazioni, consulta Connessioni Bitbucket Cloud.
  - Per GitHub GitHub Enterprise Cloud, utilizza l'opzione GitHubprovider nella console o l'`CodestarSourceConnection`azione nella CLI. Per informazioni, consulta GitHub connessioni.
  - Per GitHub Enterprise Server, utilizzate l'opzione provider GitHub Enterprise Server nella console o l'`CodestarSourceConnection`azione nella CLI. Per informazioni, consulta GitHub Connessioni Enterprise Server.
  - Per GitLab .com, utilizza l'opzione GitLabprovider nella console o l'`CodestarSourceConnection`azione con il `GitLab` provider nella CLI. Per informazioni, consulta GitLabconnessioni.com.
- Per visualizzare un tutorial introduttivo che crea una pipeline con un'origine Bitbucket e un' CodeBuild azione, vedi Guida introduttiva alle connessioni.
- Per un tutorial che mostra come connettersi a un GitHub repository e utilizzare l'opzione Full clone con un'azione a valle, consulta. CodeBuild Tutorial: usa il clone completo con una sorgente di GitHub pipeline

AWS Device Farm

Nella tua pipeline, puoi configurare un'azione di test AWS Device Farm da utilizzare per eseguire e testare l'applicazione sui dispositivi. Device Farm utilizza pool di test di dispositivi e framework di test per testare le applicazioni su dispositivi specifici. Per informazioni sui tipi di framework di test supportati dall'azione Device Farm, vedere Working with Test Types in AWS Device Farm.

Argomenti

Tipo di operazione

- Categoria: `Test`
- Proprietario: `AWS`
- Provider: `DeviceFarm`
- Versione: `1`

Parametri di configurazione

AppType

Campo obbligatorio: sì

Il sistema operativo e il tipo di applicazione che stai testando. Di seguito è riportato un elenco di valori validi:

- `iOS`
- `Android`
- `Web`

ProjectId

Campo obbligatorio: sì

L'ID del progetto Device Farm.

Per trovare l'ID del progetto, nella console Device Farm, scegli il tuo progetto. Nel browser, copiare l'URL del nuovo progetto. L'URL contiene l'ID del progetto. L'ID del progetto è il valore nell'URL successivo`projects/`. Nell'esempio seguente, l'ID del progetto è`eec4905f-98f8-40aa-9afc-4c1cfexample`.

```
https://<region-URL>/devicefarm/home?region=us-west-2#/projects/
eec4905f-98f8-40aa-9afc-4c1cfexample/runs
```

App

Campo obbligatorio: sì

Il nome e la posizione del file dell'applicazione nell'elemento di input. Ad esempio: `s3-ios-test-1.ipa`

TestSpec

Condizionale: Sì

La posizione del file di definizione delle specifiche di test nell'artefatto di input. Questo è necessario per il test in modalità personalizzata.

DevicePoolArn

Campo obbligatorio: sì

L'ARN del pool di dispositivi Device Farm.

Per ottenere gli ARN del pool di dispositivi disponibili per il progetto, incluso l'ARN per i dispositivi principali, utilizza la AWS CLI per immettere il seguente comando:

```
aws devicefarm list-device-pools --arn arn:aws:devicefarm:us-
west-2:account_ID:project:project_ID
```

TestType

Campo obbligatorio: sì

Speciifica il framework di test supportato per il test. Di seguito è riportato un elenco di valori validi per`TestType`:

- APPIUM_JAVA_JUNIT
- APPIUM_JAVA_TESTNG
- NODO_APPIUM
- APPIUM_RUBY
- APPIUM_PYTHON
- APPIUM_WEB_JAVA_JUNIT
- APPIUM_WEB_JAVA_TESTNG
- NODO APPIUM_WEB
- APPIUM_WEB_RUBY
- APPIUM_WEB_PYTHON
- BUILTIN_FUZZ
- INSTRUMENTATION
- XCTEST
- XCTEST_UI

> **Note**
>
> I seguenti tipi di test non sono supportati dall'azione in CodePipeline:`WEB_PERFORMANCE_PROFILE`, e. `REMOTE_ACCESS_RECORD` `REMOTE_ACCESS_REPLAY`

Per informazioni sui tipi di test di Device Farm, vedere Working with Test Types in AWS Device Farm.

RadioBluetoothEnabled

Campo obbligatorio: no

Un valore booleano che indica se abilitare il Bluetooth all'inizio del test.

RecordAppPerformanceData

Campo obbligatorio: no

Un valore booleano che indica se registrare dati sulle prestazioni del dispositivo come CPU, FPS e prestazioni della memoria durante il test.

RecordVideo

Campo obbligatorio: no

Un valore booleano che indica se registrare video durante il test.

RadioWifiEnabled

Campo obbligatorio: no

Un valore booleano che indica se abilitare il Wi-Fi all'inizio del test.

RadioNfcEnabled

Campo obbligatorio: no

Un valore booleano che indica se abilitare NFC all'inizio del test.

RadioGpsEnabled

Campo obbligatorio: no

Un valore booleano che indica se abilitare il GPS all'inizio del test.

Test

Campo obbligatorio: no

Il nome e il percorso del file di definizione del test nella posizione di origine. Il percorso è relativo alla cartella principale dell'artefatto di input del test.

FuzzEventCount

Campo obbligatorio: no

Il numero di eventi dell'interfaccia utente che il fuzz test deve eseguire, compreso tra 1 e 10.000.

FuzzEventThrottle

Campo obbligatorio: no

Il numero di millisecondi di attesa del fuzz test prima di eseguire il successivo evento dell'interfaccia utente, compreso tra 1 e 1.000.

FuzzRandomizerSeed

Campo obbligatorio: no

Un seme per il fuzz test da utilizzare per randomizzare gli eventi dell'interfaccia utente. Utilizzando lo stesso numero per i fuzz test successivi si ottengono sequenze di eventi identiche.

CustomHostMachineArtifacts

Campo obbligatorio: no

La posizione sul computer host in cui verranno archiviati gli elementi personalizzati.

CustomDeviceArtifacts

Campo obbligatorio: no

La posizione sul dispositivo in cui verranno archiviati gli artefatti personalizzati.

UnmeteredDevicesOnly

Campo obbligatorio: no

Un valore booleano che indica se utilizzare solo i dispositivi illimitati durante l'esecuzione dei test in questa fase.

JobTimeoutMinutes

Campo obbligatorio: no

Il numero di minuti di esecuzione di un test per dispositivo prima del timeout.

Latitudine

Campo obbligatorio: no

La latitudine del dispositivo espressa in gradi del sistema di coordinate geografiche.

Longitude

Campo obbligatorio: no

La longitudine del dispositivo espressa in gradi del sistema di coordinate geografiche.

Input artifact (Artefatti di input)

- Numero di artefatti: 1
- Descrizione: L'insieme di artefatti da mettere a disposizione dell'azione di test. Device Farm cerca l'applicazione integrata e le definizioni di test da utilizzare.

Artefatti di output

- Numero di artefatti: `0`
- Descrizione: gli artefatti di output non si applicano a questo tipo di azione.

Dichiarazione dell'operazione

YAML

```
Name: Test
Actions:
  - Name: TestDeviceFarm
    ActionTypeId: null
    category: Test
    owner: AWS
    provider: DeviceFarm
    version: '1'
RunOrder: 1
Configuration:
  App: s3-ios-test-1.ipa
  AppType: iOS
  DevicePoolArn: >-
    arn:aws:devicefarm:us-west-2::devicepool:0EXAMPLE-d7d7-48a5-ba5c-b33d66efa1f5
  ProjectId: eec4905f-98f8-40aa-9afc-4c1cfEXAMPLE
  TestType: APPIUM_PYTHON
  TestSpec: example-spec.yml
OutputArtifacts: []
InputArtifacts:
  - Name: SourceArtifact
Region: us-west-2
```

JSON

```
{
    "Name": "Test",
    "Actions": [
        {
            "Name": "TestDeviceFarm",
            "ActionTypeId": null,
            "category": "Test",
            "owner": "AWS",
```

```
            "provider": "DeviceFarm",
            "version": "1"
        }
    ],
    "RunOrder": 1,
    "Configuration": {
        "App": "s3-ios-test-1.ipa",
        "AppType": "iOS",
        "DevicePoolArn": "arn:aws:devicefarm:us-west-2::devicepool:0EXAMPLE-
d7d7-48a5-ba5c-b33d66efa1f5",
        "ProjectId": "eec4905f-98f8-40aa-9afc-4c1cfEXAMPLE",
        "TestType": "APPIUM_PYTHON",
        "TestSpec": "example-spec.yml"
    },
    "OutputArtifacts": [],
    "InputArtifacts": [
        {
            "Name": "SourceArtifact"
        }
    ],
    "Region": "us-west-2"
},
```

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Utilizzo dei tipi di test in Device Farm: questo capitolo di riferimento della Device Farm Developer Guide fornisce ulteriori descrizioni sui framework di test per Android, iOS e applicazioni Web supportati da Device Farm.
- Azioni in Device Farm: le chiamate e i parametri API nel Device Farm API Reference possono aiutarti a lavorare con i progetti Device Farm.
- Tutorial: crea una pipeline con cui creare e testare la tua app Android AWS Device Farm— Questo tutorial fornisce un esempio di file di specifiche di build e un'applicazione di esempio per creare una pipeline con un GitHub sorgente che crea e testa un'app Android con Device CodeBuild Farm.
- Tutorial: crea una pipeline con cui testare la tua app iOS AWS Device Farm— Questo tutorial fornisce un'applicazione di esempio per creare una pipeline con un sorgente Amazon S3 che testa un'app iOS integrata con Device Farm.

AWS Lambda

Consente di eseguire una funzione Lambda come azione nella pipeline. Utilizzando l'oggetto evento che è un input per questa funzione, la funzione ha accesso alla configurazione dell'operazione, alle posizioni degli artefatti di input, alle posizioni degli artefatti di output e ad altre informazioni necessarie per accedere agli artefatti. Per un esempio di evento passato a una funzione di invoke Lambda, vedere. Evento JSON di esempio Come parte dell'implementazione della funzione Lambda, deve esserci una chiamata a `PutJobSuccessResult API` o `PutJobFailureResult API`. In caso contrario, l'esecuzione di questa operazione si blocca fino a che l'operazione non scade. Se specifichi artefatti di output per l'operazione, questi devono essere caricati nel bucket S3 come parte dell'implementazione della funzione.

> **Important**
>
> Non registrate l'evento JSON CodePipeline inviato a Lambda perché ciò può comportare la registrazione delle credenziali utente nei log. CloudWatch Il CodePipeline ruolo utilizza un evento JSON per passare credenziali temporanee a Lambda sul campo. `artifactCredentials` Per un evento di esempio, consultare Evento JSON di esempio.

Tipo di operazione

- Categoria: `Invoke`
- Proprietario: `AWS`
- Provider: `Lambda`
- Versione: `1`

Parametri di configurazione

FunctionName

Campo obbligatorio: sì

`FunctionName`è il nome della funzione creata in Lambda.

UserParameters

Campo obbligatorio: no

Una stringa che può essere elaborata come input dalla funzione Lambda.

Input artifact (Artefatti di input)

- Numero di artefatti: `0 to 5`
- Descrizione: il set di artefatti da rendere disponibili per la funzione Lambda.

Artefatti di output

- Numero di artefatti: `0 to 5`
- Descrizione: l'insieme di artefatti prodotti come output dalla funzione Lambda.

Variabili di output

Questa azione produrrà come variabili tutte le coppie chiave-valore incluse nella `outputVariables` sezione della richiesta API. PutJobSuccessResult

Per ulteriori informazioni sulle variabili in CodePipeline, vedere. Variables

Esempio di configurazione dell'operazione

YAML

```
Name: Lambda
Actions:
  - Name: Lambda
    ActionTypeId:
      Category: Invoke
      Owner: AWS
      Provider: Lambda
      Version: '1'
    RunOrder: 1
    Configuration:
      FunctionName: myLambdaFunction
      UserParameters: 'http://192.0.2.4'
    OutputArtifacts: []
    InputArtifacts: []
    Region: us-west-2
```

JSON

```
{
    "Name": "Lambda",
    "Actions": [
        {
            "Name": "Lambda",
            "ActionTypeId": {
                "Category": "Invoke",
                "Owner": "AWS",
                "Provider": "Lambda",
                "Version": "1"
            },
            "RunOrder": 1,
            "Configuration": {
                "FunctionName": "myLambdaFunction",
                "UserParameters": "http://192.0.2.4"
            },
            "OutputArtifacts": [],
            "InputArtifacts": [],
            "Region": "us-west-2"
        }
    ]
},
```

Evento JSON di esempio

L'azione Lambda invia un evento JSON che contiene l'ID del lavoro, la configurazione dell'azione della pipeline, le posizioni degli artefatti di input e output e qualsiasi informazione di crittografia per gli artefatti. Il job worker accede a questi dettagli per completare l'azione Lambda. Per ulteriori informazioni, consulta Dettagli del processo. Di seguito è riportato un esempio di evento.

```
{
    "CodePipeline.job": {
        "id": "11111111-abcd-1111-abcd-111111abcdef",
        "accountId": "111111111111",
        "data": {
            "actionConfiguration": {
                "configuration": {
                    "FunctionName": "MyLambdaFunction",
                    "UserParameters": "input_parameter"
```

```
                }
            },
            "inputArtifacts": [
                {
                    "location": {
                        "s3Location": {
                            "bucketName": "bucket_name",
                            "objectKey": "filename"
                        },
                        "type": "S3"
                    },
                    "revision": null,
                    "name": "ArtifactName"
                }
            ],
            "outputArtifacts": [],
            "artifactCredentials": {
                "secretAccessKey": "secret_key",
                "sessionToken": "session_token",
                "accessKeyId": "access_key_ID"
            },
            "continuationToken": "token_ID",
            "encryptionKey": {
              "id": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
              "type": "KMS"
            }
        }
    }
}
```

L'evento JSON fornisce i seguenti dettagli del lavoro per l'azione Lambda in: CodePipeline

- `id`: l'ID univoco generato dal sistema del processo.
- `accountId`: l'ID AWS dell'account associato al job.
- `data`: altre informazioni richieste da un esecutore del processo per completare il processo.
  - `actionConfiguration`: i parametri di operazione per l'operazione Lambda. Per le definizioni, consulta Parametri di configurazione .
  - `inputArtifacts`: l'artefatto fornito all'operazione.
    - `location`: la posizione dello store degli artefatti.
      - `s3Location`: le informazioni sulla posizione dell'artefatto di input per l'operazione.

        - `bucketName`: il nome dell'archivio di artefatti della pipeline per l'azione (ad esempio, un bucket Amazon S3 denominato -2-1234567890). codepipeline-us-east
        - `objectKey`: il nome dell'applicazione (ad esempio, `CodePipelineDemoApplication.zip`).
      - `type`: il tipo di artefatto nella posizione. Al momento, `S3` è l'unico tipo di artefatto valido.
    - `revision`: l'ID revisione dell'artefatto. A seconda del tipo di oggetto, può essere un ID di commit (GitHub) o un ID di revisione (Amazon Simple Storage Service). Per ulteriori informazioni, consulta ArtifactRevision.
    - `name`: il nome dell'artefatto da utilizzare, ad esempio `MyApp`.
- `outputArtifacts`: l'output dell'operazione.
    - `location`: la posizione dello store degli artefatti.
        - `s3Location`: le informazioni sulla posizione dell'artefatto di output per l'operazione.
            - `bucketName`: il nome dell'archivio di artefatti della pipeline per l'azione (ad esempio, un bucket Amazon S3 denominato -2-1234567890). codepipeline-us-east
            - `objectKey`: il nome dell'applicazione (ad esempio, `CodePipelineDemoApplication.zip`).
        - `type`: il tipo di artefatto nella posizione. Al momento, `S3` è l'unico tipo di artefatto valido.
    - `revision`: l'ID revisione dell'artefatto. A seconda del tipo di oggetto, può essere un ID di commit (GitHub) o un ID di revisione (Amazon Simple Storage Service). Per ulteriori informazioni, consulta ArtifactRevision.
    - `name`: il nome dell'output di un artefatto, ad esempio `MyApp`.
- `artifactCredentials`: le credenziali di AWS sessione utilizzate per accedere agli artefatti di input e output nel bucket Amazon S3. Queste credenziali sono credenziali temporanee emesse da AWS Security Token Service (AWS STS).
    - `secretAccessKey`: la chiave di accesso segreta per la sessione.
    - `sessionToken`: il token per la sessione.
    - `accessKeyId`: la chiave di accesso segreta per la sessione.
- `continuationToken`: un token generato dall'operazione. Le operazioni future utilizzano questo token per identificare l'istanza in esecuzione dell'operazione. Al termine dell'operazione, non è necessario fornire alcun token di continuazione.

- `encryptionKey`: La chiave di crittografia utilizzata per crittografare i dati nell'archivio degli artefatti, ad esempio una chiave. AWS KMS Se non è definito, viene utilizzata la chiave predefinita per Amazon Simple Storage Service.
  - `id`: l'ID utilizzato per identificare la chiave. Per una chiave AWS KMS , puoi utilizzare l'ID della chiave, l'ARN della chiave o l'ARN dell'alias.
  - `type`: il tipo di chiave di crittografia, ad esempio una chiave AWS KMS .

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- AWS CloudFormation Guida per l'utente: per ulteriori informazioni sulle azioni e gli AWS CloudFormation artefatti Lambda per le pipeline, vedere Utilizzo delle funzioni di sostituzione dei parametri con CodePipeline le pipeline, Automazione della distribuzione di applicazioni basate su Lambda e Artifacts.AWS CloudFormation
- Invoca una AWS Lambda funzione in una pipeline in CodePipeline— Questa procedura fornisce una funzione Lambda di esempio e mostra come utilizzare la console per creare una pipeline con un'azione di richiamo Lambda.

Riferimento alla struttura d'azione Snyk

L'azione Snyk CodePipeline automatizza il rilevamento e la correzione delle vulnerabilità di sicurezza nel codice open source. Puoi usare Snyk con il codice sorgente dell'applicazione nel tuo repository di terze parti, come Bitbucket Cloud, GitHub o con immagini per applicazioni container. La tua azione analizzerà e segnalerà i livelli di vulnerabilità e gli avvisi che configurerai.

> **Note**

Argomenti

ID del tipo di azione

- Categoria: `Invoke`
- Proprietario: `ThirdParty`
- Provider: `Snyk`
- Versione: `1`

Esempio:

```
{
    "Category": "Invoke",
    "Owner": "ThirdParty",
    "Provider": "Snyk",
    "Version": "1"
},
```

Input artifact (Artefatti di input)

- Numero di artefatti: `1`
- Descrizione: i file che costituiscono l'artefatto di input per l'azione di richiamo.

Artefatti di output

- Numero di artefatti: `1`
- Descrizione: i file che costituiscono l'artefatto di output per l'azione di richiamo.

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Per ulteriori informazioni sull'utilizzo delle azioni Snyk in CodePipeline, consulta Automatizzare la scansione delle vulnerabilità con Snyk. CodePipeline

AWS Step Functions

Un' AWS CodePipeline azione che esegue le seguenti operazioni:

- Avvia l'esecuzione di una macchina a AWS Step Functions stati dalla pipeline.
- Fornisce uno stato iniziale alla macchina a stati tramite una proprietà nella configurazione dell'operazione o un file contenuto in un artefatto della pipeline da passare come input.
- Facoltativamente, imposta un prefisso dell'ID di esecuzione per identificare le esecuzioni che provengono dall'operazione.
- Supporta macchine a stati Standard ed Express .

> **Note**
>
> L'azione Step Functions viene eseguita su Lambda e pertanto ha quote di dimensione degli artefatti uguali alle quote di dimensione degli artefatti per le funzioni Lambda. Per ulteriori informazioni, consulta le quote Lambda nella Lambda Developer Guide.

Tipo di operazione

- Categoria: `Invoke`
- Proprietario: `AWS`
- Provider: `StepFunctions`
- Versione: `1`

Parametri di configurazione

StateMachineArn

Campo obbligatorio: sì

L' Amazon Resource Name (ARN) per la macchina a stati da richiamare.

ExecutionNamePrefix

Campo obbligatorio: no

Per impostazione predefinita, l'ID di esecuzione dell'operazione viene utilizzato come nome di esecuzione della macchina a stati. Se viene fornito un prefisso, viene anteposto all'ID di esecuzione dell'operazione con un trattino e utilizzato insieme come nome di esecuzione della macchina a stati.

```
myPrefix-1624a1d1-3699-43f0-8e1e-6bafd7fde791
```

Per una macchina a stati di tipo Express, il nome deve contenere solo caratteri 0-9, A-Z, a-z, - e _.

InputType

Campo obbligatorio: no

- Literal (Letterale) (impostazione predefinita): se specificato, il valore nel campo Input viene passato direttamente all'input della macchina a stati.

  Esempio di voce per il campo Input quando è selezionato Literal (Letterale) :

  ```
  {"action": "test"}
  ```

- FilePath: Il contenuto di un file nell'elemento di input specificato dal campo Input viene utilizzato come input per l'esecuzione della macchina a stati. Un artefatto di input è richiesto quando InputTypeè impostato su. FilePath

  Esempio di immissione per il campo Input quando FilePathè selezionato:

  ```
  assets/input.json
  ```

Input

Obbligatorio: condizionale

- Letterale: quando InputTypeè impostato su Literal (impostazione predefinita), questo campo è facoltativo.

  Se fornito, il campo Input viene utilizzato direttamente come input per l'esecuzione della macchina a stati. In caso contrario, la macchina a stati viene richiamata con un oggetto JSON vuoto `{}`.

- FilePath: Quando InputTypeè impostato su FilePath, questo campo è obbligatorio.

  Un artefatto di input è richiesto anche quando InputTypeè impostato su. FilePath

Il contenuto del file nell'artefatto di input specificato viene utilizzato come input per l'esecuzione della macchina a stati.

Input artifact (Artefatti di input)

- Numero di artefatti: `0 to 1`
- Descrizione: se InputTypeimpostato su FilePath, questo artefatto è obbligatorio e viene utilizzato per generare l'input per l'esecuzione della macchina a stati.

Artefatti di output

- Numero di artefatti: `0 to 1`
- Descrizione:
  - Macchine a stati Standard: se fornito, l'artefatto di output viene popolato con l'output della macchina a stati. Questo viene ottenuto dalla `output` proprietà della risposta dell' DescribeExecution API Step Functions dopo che l'esecuzione della macchina a stati è stata completata correttamente.
  - Macchine a stati Express: non supportate.

Variabili di output

Questa operazione produce variabili di output che possono essere referenziate mediante configurazione dell'operazione di un'operazione a valle nella pipeline.

Per ulteriori informazioni, consulta Variables.

StateMachineArn

ARN della macchina a stati.

ExecutionArn

L'ARN di esecuzione della macchina a stati. Solo macchine a stati Standard.

Esempio di configurazione dell'operazione

Esempio di input predefinito

YAML

```
Name: ActionName
ActionTypeId:
  Category: Invoke
  Owner: AWS
  Version: 1
  Provider: StepFunctions
OutputArtifacts:
  - Name: myOutputArtifact
Configuration:
  StateMachineArn: arn:aws:states:us-east-1:111122223333:stateMachine:HelloWorld-
StateMachine
  ExecutionNamePrefix: my-prefix
```

JSON

```
{
    "Name": "ActionName",
    "ActionTypeId": {
        "Category": "Invoke",
        "Owner": "AWS",
        "Version": 1,
        "Provider": "StepFunctions"
    },
    "OutputArtifacts": [
        {
            "Name": "myOutputArtifact"
        }
    ],
    "Configuration": {
        "StateMachineArn": "arn:aws:states:us-
east-1:111122223333:stateMachine:HelloWorld-StateMachine",
        "ExecutionNamePrefix": "my-prefix"
    }
}
```

Esempio di input letterale

YAML

```
Name: ActionName
ActionTypeId:
  Category: Invoke
  Owner: AWS
  Version: 1
  Provider: StepFunctions
OutputArtifacts:
  - Name: myOutputArtifact
Configuration:
  StateMachineArn: arn:aws:states:us-east-1:111122223333:stateMachine:HelloWorld-
StateMachine
  ExecutionNamePrefix: my-prefix
  Input: '{"action": "test"}'
```

JSON

```
{
    "Name": "ActionName",
    "ActionTypeId": {
        "Category": "Invoke",
        "Owner": "AWS",
        "Version": 1,
        "Provider": "StepFunctions"
    },
    "OutputArtifacts": [
        {
            "Name": "myOutputArtifact"
        }
    ],
    "Configuration": {
        "StateMachineArn": "arn:aws:states:us-
east-1:111122223333:stateMachine:HelloWorld-StateMachine",
        "ExecutionNamePrefix": "my-prefix",
        "Input": "{\"action\": \"test\"}"
    }
}
```

Esempio di file di input

YAML

```
Name: ActionName
InputArtifacts:
  - Name: myInputArtifact
ActionTypeId:
  Category: Invoke
  Owner: AWS
  Version: 1
  Provider: StepFunctions
OutputArtifacts:
  - Name: myOutputArtifact
Configuration:
  StateMachineArn: 'arn:aws:states:us-east-1:111122223333:stateMachine:HelloWorld-
StateMachine'
  ExecutionNamePrefix: my-prefix
  InputType: FilePath
  Input: assets/input.json
```

JSON

```
{
    "Name": "ActionName",
    "InputArtifacts": [
        {
            "Name": "myInputArtifact"
        }
    ],
    "ActionTypeId": {
        "Category": "Invoke",
        "Owner": "AWS",
        "Version": 1,
        "Provider": "StepFunctions"
    },
    "OutputArtifacts": [
        {
            "Name": "myOutputArtifact"
        }
    ],
    "Configuration": {
```

```
        "StateMachineArn": "arn:aws:states:us-
east-1:111122223333:stateMachine:HelloWorld-StateMachine",
        "ExecutionNamePrefix": "my-prefix",
        "InputType": "FilePath",
        "Input": "assets/input.json"
    }
}
```

Comportamento

Durante una versione, CodePipeline esegue la macchina a stati configurata utilizzando l'input specificato nella configurazione dell'azione.

Quando InputTypeè impostato su Literal, il contenuto del campo di configurazione dell'azione di input viene utilizzato come input per la macchina a stati. Quando l'input letterale non viene fornito, l'esecuzione della macchina a stati utilizza un oggetto JSON vuoto `{}`. Per ulteriori informazioni sull'esecuzione di un'esecuzione di una macchina a stati senza input, consulta l' StartExecutionAPI Step Functions.

Quando InputTypeè impostata su FilePath, l'azione decomprime l'elemento di input e utilizza il contenuto del file specificato nel campo di configurazione dell'azione di input come input per la macchina a stati. Quando FilePathè specificato, il campo di input è obbligatorio e deve esistere un elemento di input; in caso contrario, l'azione ha esito negativo.

Dopo l'esecuzione di un avvio riuscito, il comportamento divergerà per i due tipi di macchina a stati, Standard ed Express.

Macchine a stati Standard

Se l'esecuzione della macchina a stati standard è stata avviata correttamente, esegue il CodePipeline polling dell'`DescribeExecution`API finché l'esecuzione non raggiunge lo stato di terminale. Se l'esecuzione viene completata correttamente, l'operazione ha esito positivo; in caso contrario, ha esito negativo.

Se è configurato un artefatto di output, l'artefatto conterrà il valore restituito dalla macchina a stati. Questo viene ottenuto dalla `output` proprietà della risposta dell' DescribeExecution API Step Functions dopo che l'esecuzione della macchina a stati è stata completata correttamente. Si noti che vi sono dei vincoli di lunghezza di output applicati a questa API.

Gestione degli errori

- Se l'operazione non riesce ad avviare un'esecuzione di una macchina a stati, l'esecuzione dell'operazione ha esito negativo.
- Se l'esecuzione della macchina a stati non riesce a raggiungere lo stato di terminale prima che l'azione CodePipeline Step Functions raggiunga il suo timeout (impostazione predefinita di 7 giorni), l'esecuzione dell'azione fallisce. La macchina a stati potrebbe continuare nonostante questo errore. Per ulteriori informazioni sui timeout di esecuzione delle macchine a stati in Step Functions, consulta Flussi di lavoro Standard e Express.

  > **Note**
  >
  > È possibile richiedere un aumento della quota per il timeout dell'azione di richiamo per l'account con l'operazione. Tuttavia, l'aumento delle quote si applica a tutte le operazioni di questo tipo in tutte le regioni per tale account.

- Se l'esecuzione della macchina a stati raggiunge uno stato terminale di FAILED, TIMED_OUT o ABORTED, l'esecuzione dell'operazione ha esito negativo.

Macchine a stati Express

Se l'esecuzione della macchina a stati Express è stata avviata correttamente, l'esecuzione dell'operazione di richiamo viene completata correttamente.

Considerazioni relative alle operazioni configurate per le macchine a stati Express:

- Non è possibile designare un artefatto di output.
- L'operazione non attende il completamento dell'esecuzione della macchina a stati.
- Dopo l'avvio dell'esecuzione dell'azione CodePipeline, l'esecuzione dell'azione riesce anche se l'esecuzione della macchina a stati fallisce.

Gestione degli errori

- Se CodePipeline non riesce ad avviare l'esecuzione di una macchina a stati, l'esecuzione dell'azione fallisce. Altrimenti, l'operazione viene eseguita immediatamente con esito positivo. L'azione riesce CodePipeline indipendentemente dal tempo impiegato dall'esecuzione della macchina a stati per essere completata o dal relativo risultato.

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- AWS Step Functions Guida per gli sviluppatori: per informazioni sulle macchine a stati, le esecuzioni e gli input per le macchine a stati, consulta la Guida per gli AWS Step Functions sviluppatori.
- Tutorial: utilizzare un'azione di AWS Step Functions richiamo in una pipeline— Questo tutorial ti consente di iniziare con una macchina a stati standard di esempio e mostra come utilizzare la console per aggiornare una pipeline aggiungendo un'azione di invoca Step Functions.

Riferimento al modello di integrazione

Esistono diverse integrazioni predefinite per servizi di terze parti che aiutano a integrare gli strumenti esistenti per i clienti nel processo di rilascio della pipeline. I partner o i fornitori di servizi terzi utilizzano un modello di integrazione per implementare i tipi di azioni da utilizzare in. CodePipeline

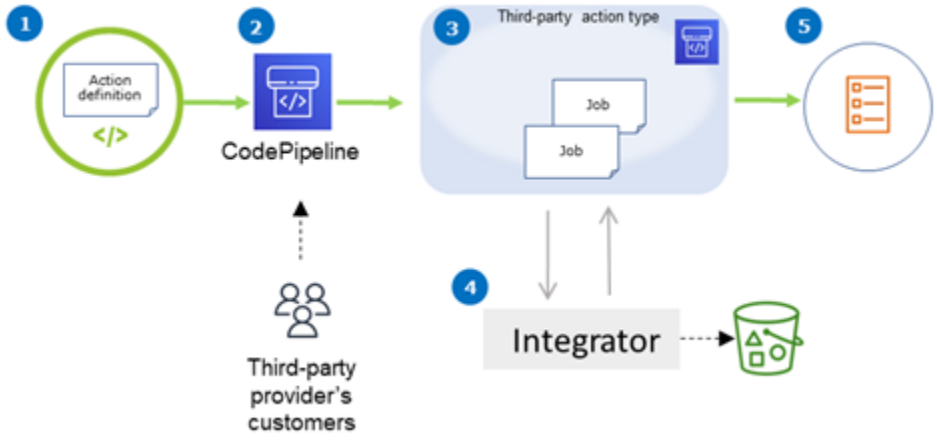
Utilizza questo riferimento quando pianifichi o lavori con tipi di azioni gestiti con un modello di integrazione supportato in CodePipeline.

Per certificare il tipo di azione di terze parti come integrazione con i partner CodePipeline, fai riferimento al AWS Partner Network (APN). Queste informazioni sono un supplemento al Reference.AWS CLI

Argomenti

Come funzionano i tipi di azioni di terze parti con l'integratore

Puoi aggiungere tipi di azioni di terze parti alle pipeline dei clienti per completare le attività relative alle risorse dei clienti. L'integratore gestisce le richieste di lavoro ed esegue l'azione con. CodePipeline Il diagramma seguente mostra un tipo di azione di terze parti creato per essere utilizzato dai clienti nella loro pipeline. Dopo che il cliente ha configurato l'azione, l'azione viene eseguita e crea richieste di lavoro che vengono gestite dal motore di azione dell'integratore.



Il diagramma mostra i seguenti passaggi:

1. La definizione dell'azione viene registrata e resa disponibile in CodePipeline. L'azione di terze parti è disponibile per i clienti del fornitore terzo.
2. Il cliente del provider sceglie e configura l'azione in. CodePipeline
3. L'azione viene eseguita e i lavori vengono messi in coda. CodePipeline Quando il lavoro è pronto CodePipeline, invia una richiesta di lavoro.
4. L'integratore (il job worker per le API di polling di terze parti o la funzione Lambda) raccoglie la richiesta di lavoro, restituisce una conferma e lavora sugli artefatti per le azioni.
5. L'integratore restituisce un output di successo/fallimento (il job worker utilizza API di successo/fallimento o la funzione Lambda invia un output di successo/fallimento) con un risultato del lavoro e un token di continuazione.

Per informazioni sui passaggi che è possibile utilizzare per richiedere, visualizzare e aggiornare un tipo di azione, vedere. Lavorare con i tipi di azione

Concetti

Questa sezione utilizza i seguenti termini per i tipi di azioni di terze parti:

Tipo di operazione

Un processo ripetibile che può essere riutilizzato in pipeline che eseguono gli stessi carichi di lavoro di distribuzione continua. I tipi di azione sono identificati da un`Owner`,, e`Category`. `Provider` `Version` Per esempio:

```
            {

                "Category": "Deploy",
                "Owner": "AWS",
                "Provider": "CodeDeploy",
                "Version": "1"
            },
```

Tutte le azioni dello stesso tipo condividono la stessa implementazione.

Azione

Una singola istanza di un tipo di azione, uno dei processi discreti che avvengono all'interno di una fase di una pipeline. Ciò include in genere i valori utente specifici della pipeline in cui viene eseguita l'azione.

Definizione dell'azione

Lo schema per un tipo di azione che definisce le proprietà richieste per configurare l'azione e gli artefatti di input/output.

Esecuzione dell'operazione

Una raccolta di lavori che sono stati eseguiti per determinare se l'azione sulla pipeline del cliente ha avuto successo o meno.

Motore di esecuzione delle azioni

Una proprietà della configurazione di esecuzione dell'azione che definisce il tipo di integrazione utilizzato da un tipo di azione. I valori validi sono `JobWorker` e `Lambda`.

Integrazione

Descrive un software eseguito da un integratore per implementare un tipo di azione. CodePipeline supporta due tipi di integrazione corrispondenti ai due motori `JobWorker` di azione supportati e. `Lambda`

Integratore

La persona che possiede l'implementazione di un tipo di azione.

Processo

Un lavoro basato sulla pipeline e sul contesto del cliente per eseguire un'integrazione. L'esecuzione di un'azione è composta da uno o più lavori.

Job worker

Il servizio che elabora l'input del cliente e gestisce un lavoro.

Modelli di integrazione supportati

CodePipeline dispone di due modelli di integrazione:

- Modello di integrazione Lambda: questo modello di integrazione è il modo preferito per lavorare con i tipi di azione. CodePipeline Il modello di integrazione Lambda utilizza una funzione Lambda per elaborare le richieste di lavoro durante l'esecuzione dell'azione.
- Modello di integrazione Job Worker: Il modello di integrazione Job Worker è il modello utilizzato in precedenza per le integrazioni di terze parti. Il modello di integrazione dei job worker utilizza un job worker configurato per contattare le CodePipeline API per elaborare le richieste di lavoro al momento dell'esecuzione dell'azione.

A titolo di confronto, la tabella seguente descrive le caratteristiche dei due modelli:

	Modello di integrazione Lambda	Modello di integrazione Job Worker
Descrizione	L'integratore scrive l'integrazione come funzione Lambda, che viene richiamata CodePipeline ogni volta che è disponibile un lavoro per l'azione. La funzione Lambda non esegue un sondaggio per i lavori disponibili, ma attende la ricezione della richiesta di lavoro successiv a.	L'integratore scrive l'integrazione come job worker che analizza costantemente i posti di lavoro disponibili nelle pipeline del cliente. Il job worker esegue quindi il lavoro e invia i risultati del lavoro utilizzando le API. CodePipeline CodePipeline
Infrastruttura	AWS Lambda	Implementa il codice Job Worker nell'infrastruttura dell'integratore, come le istanze Amazon EC2.

	Modello di integrazione Lambda	Modello di integrazione Job Worker
Impegno di sviluppo	L'integrazione contiene solo la logica aziendale.	L'integrazione deve interagire con le CodePipeline API oltre a contenere la logica aziendale.
Operazioni e sforzi	Minore impegno operativo poiché l'infrastruttura è costituita solo da AWS risorse.	Maggiore impegno operativo perché il lavoratore necessita di un hardware autonomo.
Durata massima di esecuzione del Job	Se l'integrazione deve essere eseguita attivamente per più di 15 minuti, questo modello non può essere utilizzato. Questa azione è destinata agli integratori che devono avviare un processo (ad esempio, avviare una build sull'elemento di codice del cliente) e restituire un risultato al termine. Non è consigliabile che l'integra tore continui ad aspettare il completamento della build. Invece, restituisci una continuazione. CodePipelinecrea un nuovo lavoro in altri 30 secondi se viene ricevuta una continuazione dal codice dell'integratore per controllare il lavoro fino al termine.	Utilizzando questo modello è possibile sostenere lavori di esecuzione molto lunga (ore/gior ni).

Modello di integrazione Lambda

Il modello di integrazione Lambda supportato include la creazione della funzione Lambda e la definizione dell'output per il tipo di azione di terze parti.

Aggiorna la tua funzione Lambda per gestire l'input da CodePipeline

È possibile creare una nuova funzione Lambda. Puoi aggiungere una logica aziendale alla tua funzione Lambda che viene eseguita ogni volta che c'è un lavoro disponibile nella pipeline per il tuo tipo di azione. Ad esempio, dato il contesto del cliente e della pipeline, potresti voler iniziare a integrare il tuo servizio per il cliente.

Usa i seguenti parametri per aggiornare la funzione Lambda per gestire l'input da. CodePipeline

Formato:

- `jobId`:
  - L'ID univoco del lavoro generato dal sistema.
  - ▬Tipo: stringa
  - Schema: [0-9a-f] {8} - [0-9a-f] {4} - [0-9a-f] {4} - [0-9a-f] {4} - [0-9a-f] {12}
- `accountId`:
  - L'ID AWS dell'account del cliente da utilizzare durante l'esecuzione del lavoro.
  - ▬Tipo: stringa
  - Modello: [0-9] {12}
- `data`:
  - Altre informazioni su un processo utilizzato da un'integrazione per completare il lavoro.
  - Contiene una mappa di quanto segue:
    - `actionConfiguration`:
      - I dati di configurazione per l'azione. I campi di configurazione dell'azione sono una mappatura di coppie chiave-valore per consentire al cliente di inserire valori. Le chiavi sono determinate dai parametri chiave nel file di definizione del tipo di azione al momento della configurazione dell'azione. In questo esempio, i valori sono determinati dall'utente dell'azione specificando le informazioni nei `Password` campi `Username` and.
      - Tipo: mappa da stringa a stringa, facoltativamente presente

        Esempio:

```
"configuration": {
    "Username": "MyUser",
    "Password": "MyPassword"
```

```
    },
```

- `encryptionKey`:
  - Rappresenta le informazioni sulla chiave utilizzata per crittografare i dati nell'archivio degli artefatti, ad esempio una chiave. AWS KMS
  - Contenuto: Tipo di tipo di dati, presente opzionalmente `encryptionKey`
- `inputArtifacts`:
  - Elenco di informazioni su un artefatto su cui lavorare, ad esempio un artefatto di test o di costruzione.
  - Contenuto: Elenco del tipo di dati, presente opzionalmente `Artifact`
- `outputArtifacts`:
  - Elenco di informazioni sull'output di un'azione.
  - Contenuto: Elenco del tipo di dati`Artifact`, presente opzionalmente
- `actionCredentials`:
  - Rappresenta un oggetto di credenziali di AWS sessione. Queste credenziali sono credenziali temporanee emesse da. AWS STS Possono essere utilizzate per accedere agli artefatti di input e output nel bucket S3 utilizzato per archiviare gli artefatti per la pipeline. CodePipeline

    Queste credenziali dispongono inoltre delle stesse autorizzazioni del modello di istruzioni politiche specificato nel file di definizione del tipo di azione.
  - Contenuto: tipo di dati`AWSSessionCredentials`, facoltativamente presente
- `actionExecutionId`:
  - L'ID esterno dell'esecuzione dell'azione.
  - Tipo: stringa
- `continuationToken`:
  - Un token generato dal sistema, ad esempio un ID di distribuzione, richiesto da un processo per continuare il lavoro in modo asincrono.
  - Tipo: stringa, facoltativamente presente

Tipi di dati:

- `encryptionKey`:
  - `id`:

    - ID utilizzato per identificare la chiave. Per ogni AWS KMS chiave, è possibile utilizzare l'ID chiave, la chiave ARN o l'alias ARN.
    - ▬Tipo: stringa
  - `type`:
    - Il tipo di chiave di crittografia, ad esempio una chiave. AWS KMS
    - ▬Tipo: stringa
    - Valori validi: `KMS`
- `Artifact`:
  - `name`:
    - Il nome dell'artefatto.
    - Tipo: String, facoltativamente presente
  - `revision`:
    - L'ID di revisione dell'artefatto. A seconda del tipo di oggetto, potrebbe essere un ID di commit (GitHub) o un ID di revisione (Amazon S3).
    - Tipo: stringa, presente opzionalmente
  - `location`:
    - La posizione di un artefatto.
    - Contenuto: Tipo di tipo di dati`ArtifactLocation`, facoltativamente presente
- `ArtifactLocation`:
  - `type`:
    - Il tipo di artefatto presente nel luogo.
    - Tipo: stringa, facoltativamente presente
    - Valori validi: `S3`
  - `s3Location`:
    - La posizione del bucket S3 che contiene una revisione.
    - Contenuto: Tipo di tipo di dati, facoltativamente presente `S3Location`
- `S3Location`:
  - `bucketName`:
    - Nome del bucket S3.
    - ▬Tipo: stringa
  - `objectKey`:

    - La chiave dell'oggetto nel bucket S3, che identifica in modo univoco l'oggetto nel bucket.
    - ▬Tipo: stringa
- `AWSSessionCredentials`:
  - `accessKeyId`:
    - La chiave di accesso per la sessione.
    - ▬Tipo: stringa
  - `secretAccessKey`:
    - La chiave di accesso segreta per la sessione.
    - ▬Tipo: stringa
  - `sessionToken`:
    - Il token per la sessione.
    - ▬Tipo: stringa

Esempio:

```
{
    "jobId": "01234567-abcd-abcd-abcd-012345678910",
    "accountId": "012345678910",
    "data": {
        "actionConfiguration": {
            "key1": "value1",
            "key2": "value2"
        },
        "encryptionKey": {
            "id": "123-abc",
            "type": "KMS"
        },
        "inputArtifacts": [
            {
                "name": "input-art-name",
                "location": {
                    "type": "S3",
                    "s3Location": {
                        "bucketName": "inputBucket",
                        "objectKey": "inputKey"
                    }
                }
            }
```

```
        ],
        "outputArtifacts": [
            {
                "name": "output-art-name",
                "location": {
                    "type": "S3",
                    "s3Location": {
                        "bucketName": "outputBucket",
                        "objectKey": "outputKey"
                    }
                }
            }
        ],
        "actionExecutionId": "actionExecutionId",
        "actionCredentials": {
            "accessKeyId": "access-id",
            "secretAccessKey": "secret-id",
            "sessionToken": "session-id"
        },
        "continuationToken": "continueId-xxyyzz"
    }
}
```

Restituisci i risultati della tua funzione Lambda a CodePipeline

La risorsa job worker dell'integratore deve restituire un payload valido in caso di successo, fallimento o continuazione.

Formato:

- `result`: Il risultato del lavoro.
  - Richiesto
  - Valori validi (senza distinzione tra maiuscole e minuscole):
    - `Success`: indica che un lavoro è stato completato correttamente e che è terminato.
    - `Continue`: Indica che un lavoro ha esito positivo e deve continuare, ad esempio se il job worker viene nuovamente richiamato per l'esecuzione della stessa azione.
    - `Fail`: Indica che un lavoro non è riuscito ed è terminale.
- `failureType`: tipo di errore da associare a un processo non riuscito.

La `failureType` categoria relativa alle azioni dei partner descrive il tipo di errore riscontrato durante l'esecuzione del processo. Gli integratori impostano il tipo insieme al messaggio di errore quando restituiscono il risultato di un errore di lavoro a. CodePipeline

- Facoltativo. Obbligatorio se il risultato è. `Fail`
- Deve essere nullo se `result` è `Success` o `Continue`
- Valori validi:
  - ConfigurationError
  - JobFailed
  - PermissionsError
  - RevisionOutOfSync
  - RevisionUnavailable
  - SystemUnavailable

- `continuation`: stato di continuazione da passare al processo successivo nell'ambito dell'esecuzione dell'azione corrente.
  - Facoltativo. Obbligatorio se il risultato è`Continue`.
  - Deve essere nullo se `result` è `Success` o`Fail`.
  - Proprietà:
    - `State`: Un hash dello stato da passare.
- `status`: Stato dell'esecuzione dell'azione.
  - Facoltativo.
  - Proprietà:
    - `ExternalExecutionId`: un ID di esecuzione esterno o un ID di commit opzionale da associare al job.
    - `Summary`: un riepilogo facoltativo di ciò che è accaduto. Negli scenari di errore, questo diventa il messaggio di errore visualizzato dall'utente.
- `outputVariables`: Un insieme di coppie chiave/valore da passare alla successiva esecuzione dell'azione.
  - Facoltativo.
  - Deve essere nullo se `result` è o. `Continue` `Fail`

Esempio:

```
{
    "result": "success",
    "failureType": null,
    "continuation": null,
    "status": {
        "externalExecutionId": "my-commit-id-123",
        "summary": "everything is dandy"
    },
    "outputVariables": {
        "FirstOne": "Nice",
        "SecondOne": "Nicest",
        ...
    }
}
```

Utilizza i token di continuazione per attendere i risultati di un processo asincrono

Il `continuation` token fa parte del payload e il risultato della funzione Lambda. È un modo per passare lo stato del lavoro a CodePipeline e indicare che il lavoro deve essere continuato. Ad esempio, dopo che un integratore ha avviato una build per il cliente sulla propria risorsa, non attende il completamento della build, ma indica CodePipeline che non ha un risultato terminale restituendo `result` as `continue` e restituendo l'ID univoco della build al CodePipeline token as. `continuation`

> **Note**
>
> Le funzioni Lambda possono essere eseguite solo per un massimo di 15 minuti. Se il processo deve durare più a lungo, puoi utilizzare i token di continuazione.

Il CodePipeline team richiama l'integratore dopo 30 secondi con lo stesso `continuation` token nel payload in modo che possa verificarne il completamento. Se la build viene completata, l'integratore restituisce il risultato di successo/fallimento del terminale, altrimenti continua.

Fornisci CodePipeline le autorizzazioni per richiamare la funzione Lambda dell'integratore in fase di esecuzione

Aggiungi le autorizzazioni alla funzione Lambda dell'integratore per fornire al servizio CodePipeline le autorizzazioni per richiamarlo utilizzando il service principal:. CodePipeline `codepipeline.amazonaws.com` È possibile aggiungere le autorizzazioni utilizzando o la riga di comando. AWS CloudFormation Per vedere un esempio, consulta Lavorare con i tipi di azione.

Modello di integrazione Job Worker

Dopo aver progettato il flusso di lavoro di alto livello, puoi creare il tuo job worker. Sebbene le specifiche dell'azione di terze parti determinino ciò che è necessario per il lavoratore, la maggior parte dei job worker per le azioni di terze parti include le seguenti funzionalità:

- Sondaggio dei lavori relativi all'utilizzo. CodePipeline `PollForThirdPartyJobs`
- Riconoscimento delle offerte di lavoro e restituzione dei risultati all' CodePipeline utilizzo di `AcknowledgeThirdPartyJobPutThirdPartyJobSuccessResult`, e. `PutThirdPartyJobFailureResult`
- Recupero e/o inserimento di artefatti nel bucket Amazon S3 per la pipeline. Per scaricare artefatti dal bucket Amazon S3, devi creare un client Amazon S3 che utilizzi la firma Signature Version 4 (Sig V4). Sig V4 è richiesto per. AWS KMS

  Per caricare artefatti nel bucket Amazon S3, devi anche configurare la richiesta Amazon S3 per utilizzare la crittografia tramite (`PutObject`). AWS Key Management Service AWS KMS AWS KMS usi. AWS KMS keys Per sapere se utilizzare la chiave gestita dal cliente Chiave gestita da AWS o una chiave gestita dal cliente per caricare gli artefatti, il lavoratore deve esaminare i dati del lavoro e verificare la proprietà della chiave di crittografia. Se la proprietà è impostata, è necessario utilizzare l'ID della chiave gestita dal cliente durante la configurazione. AWS KMS Se la proprietà della chiave è nulla, si utilizza la. Chiave gestita da AWS CodePipeline utilizza il, Chiave gestita da AWS a meno che non sia configurato diversamente.

  Per un esempio che mostra come creare i AWS KMS parametri in Java o.NET, consulta Specificare AWS Key Management Service in Amazon S3 l'uso AWS degli SDK. Per ulteriori informazioni sul bucket Amazon S3 per CodePipeline, consulta. CodePipeline concetti

Scelta e configurazione di una strategia di gestione delle autorizzazioni per l'esecutore del processo

Per sviluppare un job worker su cui affidare l'intervento di terzi CodePipeline, è necessaria una strategia per l'integrazione della gestione degli utenti e delle autorizzazioni.

La strategia più semplice consiste nell'aggiungere l'infrastruttura necessaria per il lavoratore creando istanze Amazon EC2 con un ruolo di istanza AWS Identity and Access Management (IAM), che consentono di scalare facilmente le risorse necessarie per l'integrazione. Puoi utilizzare l'integrazione integrata con AWS per semplificare l'interazione tra il tuo lavoratore e. CodePipeline

Scopri di più su Amazon EC2 e determina se è la scelta giusta per la tua integrazione. Per informazioni, consulta Amazon EC2 - Virtual Server Hosting. Per informazioni sulla configurazione di un'istanza Amazon EC2, consulta Getting Started with Amazon EC2 Linux Instances.

Un'altra strategia da considerare consiste nell'utilizzare la federazione delle identità con IAM per integrare il sistema e le risorse esistenti del provider di identità. Questa strategia è utile se disponi già di un provider di identità aziendale o se sei già configurato per supportare gli utenti che utilizzano provider di identità Web. La federazione delle identità consente di concedere un accesso sicuro alle AWS risorse CodePipeline, anche senza dover creare o gestire utenti IAM. Puoi utilizzare le caratteristiche e le policy per requisiti di sicurezza delle password e rotazione delle credenziali. Puoi utilizzare applicazioni di esempio come modelli per il tuo progetto. Per informazioni, consulta Gestione della federazione.

Per fornire l'accesso, aggiungi autorizzazioni ai tuoi utenti, gruppi o ruoli:

- Utenti e gruppi in AWS IAM Identity Center:

  Crea un set di autorizzazioni. Segui le istruzioni riportate nella pagina Create a permission set (Creazione di un set di autorizzazioni) nella Guida per l'utente di AWS IAM Identity Center .
- Utenti gestiti in IAM tramite un provider di identità:

  Crea un ruolo per la federazione delle identità. Segui le istruzioni riportate nella pagina Creating a role for a third-party identity provider (federation) (Creazione di un ruolo per un provider di identità di terze parti [federazione]) nella Guida per l'utente di IAM.
- Utenti IAM:
  - Crea un ruolo che l'utente possa assumere. Per istruzioni, consulta la pagina Creating a role for an IAM user (Creazione di un ruolo per un utente IAM) nella Guida per l'utente di IAM.

- (Non consigliato) Collega una policy direttamente a un utente o aggiungi un utente a un gruppo di utenti. Segui le istruzioni riportate nella pagina Aggiunta di autorizzazioni a un utente (console) nella Guida per l'utente di IAM.

Di seguito è riportato un esempio di politica che potresti creare per l'utilizzo con il tuo collaboratore esterno. Questa policy è intesa solo come un esempio ed è fornita senza modifiche.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codepipeline:PollForThirdPartyJobs",
        "codepipeline:AcknowledgeThirdPartyJob",
        "codepipeline:GetThirdPartyJobDetails",
        "codepipeline:PutThirdPartyJobSuccessResult",
        "codepipeline:PutThirdPartyJobFailureResult"
      ],
      "Resource": [
        "arn:aws:codepipeline:us-east-2::actionType:ThirdParty/Build/Provider/1/"
      ]
    }
  ]
}
```

Riferimento per il file di definizioni delle immagini

Questa sezione è solo per riferimento. Per ulteriori informazioni su come creare una pipeline con l'origine o distribuire operazioni di distribuzione per container, consulta Creare una pipeline in CodePipeline.

AWS CodePipeline i job worker per le azioni relative ai container, come un'azione sorgente Amazon ECR o le azioni di distribuzione di Amazon ECS, utilizzano i file delle definizioni per mappare l'URI dell'immagine e il nome del contenitore alla definizione dell'attività. Ogni file di definizioni è un file in formato JSON utilizzato dal provider di operazioni come segue:

- Le distribuzioni standard di Amazon ECS richiedono un `imagedefinitions.json` file come input per l'azione di distribuzione.
- Le distribuzioni blu/green di Amazon ECS richiedono un `imageDetail.json` file come input per l'azione di distribuzione.
  - Le operazioni di origine Amazon ECR generano un file `imageDetail.json` file che viene fornito come un output dall'operazione di origine.

Argomenti

file imagedefinitions.json per le azioni di distribuzione standard di Amazon ECS

Un documento di definizioni delle immagini è un file JSON che descrive il nome del contenitore Amazon ECS, l'immagine e il tag. Se stai distribuendo applicazioni basate su container, devi generare un file di definizioni delle immagini per fornire al CodePipeline job worker il contenitore Amazon ECS e l'identificazione dell'immagine da recuperare dal repository di immagini, come Amazon ECR.

> **Note**
>
> Il nome file predefinito è `imagedefinitions.json`. Se scegli di utilizzare un nome di file diverso, è necessario fornirlo quando la fase di distribuzione della pipeline.

Crea il file `imagedefinitions.json` con le seguenti considerazioni:

- Il file deve utilizzare la codifica UTF-8.
- Il limite massimo delle dimensioni del file di definizioni delle immagini è di 100 KB.
- È necessario creare il file come un artefatto di origine o di compilazione in modo che sia un artefatto di input per l'operazione di distribuzione. In altre parole, assicuratevi che il file sia caricato nella posizione di origine, ad esempio nel CodeCommit repository, o generato come elemento di output integrato.

Il file `imagedefinitions.json` fornisce il nome del container e l'URI dell'immagine. Deve essere costruito con il seguente set di coppie chiave-valore.

Chiave	Valore
nome	*container_name*
imageUri	*imageUri*

Di seguito è riportata la struttura JSON, in cui il nome del container è `sample-app`, l'URI dell'immagine è `ecs-repo` e il tag è `latest`:

```
[
  {
    "name": "sample-app",
    "imageUri": "11111EXAMPLE.dkr.ecr.us-west-2.amazonaws.com/ecs-repo:latest"
  }
]
```

È anche possibile costruire il file per elencare più coppie container-immagine.

Struttura JSON:

```
[
  {
    "name": "simple-app",
    "imageUri": "httpd:2.4"
  },
  {
    "name": "simple-app-1",
    "imageUri": "mysql"
  },
  {
    "name": "simple-app-2",
    "imageUri": "java1.8"
  }
]
```

Prima di creare la pipeline, utilizza la procedura descritta di seguito per impostare il file `imagedefinitions.json`.

1. Come parte della pianificazione della distribuzione dell'applicazione basata su container per la pipeline, pianifica la fase di origine e la fase di compilazione, se applicabili.
2. Seleziona una delle seguenti opzioni:

   a. Se la pipeline è stata creata in modo da saltare la fase di compilazione, è necessario creare manualmente il file JSON e caricarlo nel repository di origine in modo che l'azione di origine possa fornire l'artefatto. Crea il file utilizzando un editor di testo e assegna un nome al file o utilizza il nome file predefinito `imagedefinitions.json`. Esegui il push del file di definizioni delle immagini nel repository di origine.

   > **Note**
   >
   > Se il tuo repository di origine è un bucket Amazon S3, ricordati di comprimere il file JSON.

   b. Se la pipeline dispone di una fase di compilazione, aggiungi un comando al file delle specifiche di compilazione che genera il file di definizioni delle immagini nel repository di origine durante la fase di compilazione. L'esempio seguente usa il comando printf per creare un file `imagedefinitions.json`. Elencare il comando nella sezione `post_build` del file `buildspec.yml`:

```
printf '[{"name":"container_name","imageUri":"image_URI"}]' >
imagedefinitions.json
```

Il file di definizioni delle immagini deve essere incluso come un artefatto di output nel file `buildspec.yml`.

3. Quando si crea la pipeline nella console, nella pagina Deploy (Distribuzione) della procedura guidata Create Pipeline (Creazione pipeline) in Image Filename (Nome file immagini) immettere il nome del file di definizioni delle immagini.

Per un step-by-step tutorial sulla creazione di una pipeline che utilizza Amazon ECS come provider di distribuzione, consulta Tutorial: Continuous Deployment with. CodePipeline

File ImageDetail.json per le azioni di distribuzione blu/verde di Amazon ECS

Un `imageDetail.json` documento è un file JSON che descrive l'URI dell'immagine Amazon ECS. Se stai distribuendo applicazioni basate su container per una distribuzione blu/verde, devi generare il file per fornire `imageDetail.json` ad Amazon ECS e al CodeDeploy job worker l'identificazione dell'immagine da recuperare dall'archivio di immagini, come Amazon ECR.

> **Note**
>
> Il nome del file deve essere `imageDetail.json`.

Per una descrizione dell'azione e dei relativi parametri, consulta. Amazon Elastic Container Service e CodeDeploy blu-verde

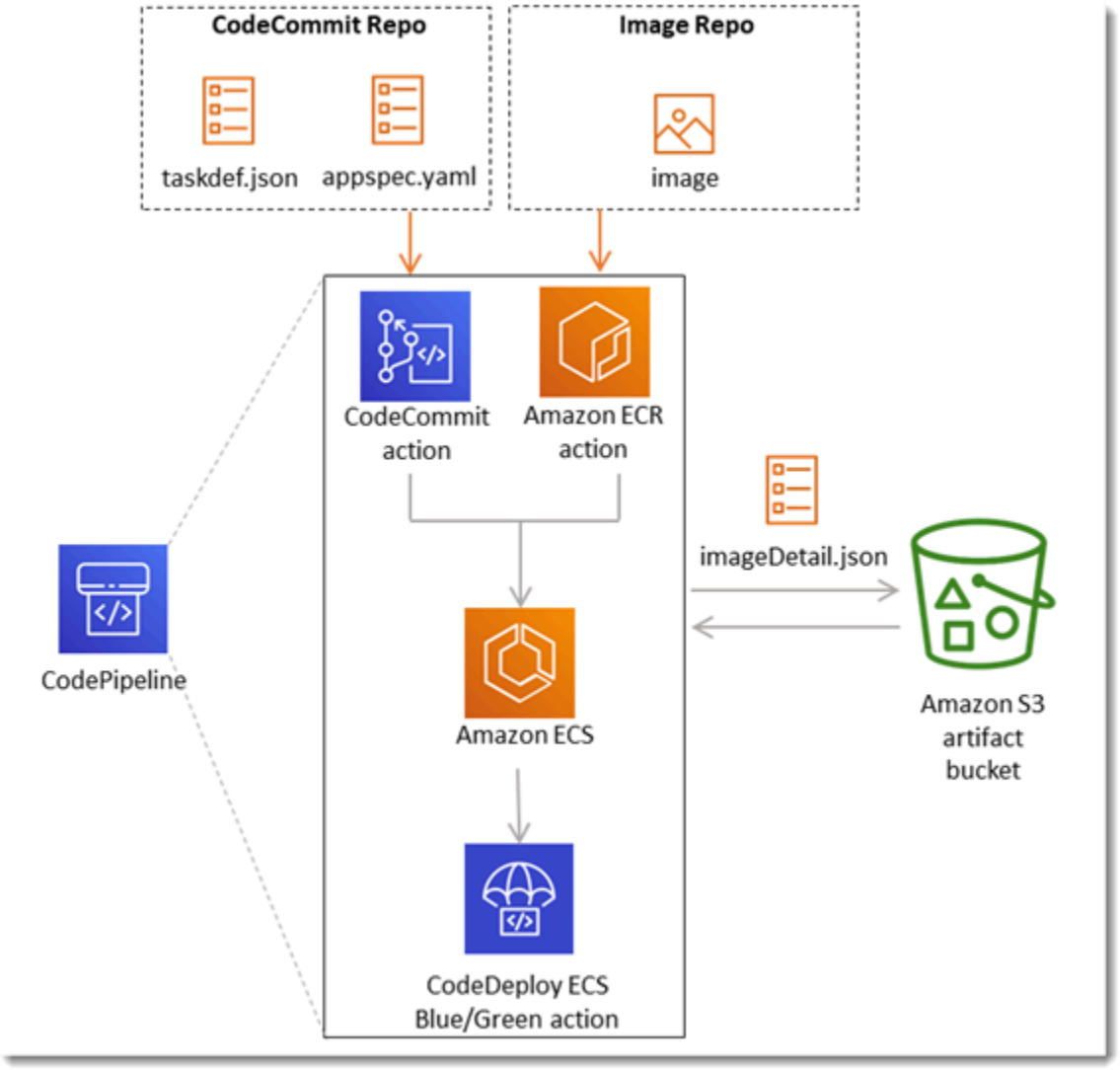
È necessario creare il file `imageDetail.json` come un artefatto di origine o di compilazione in modo che sia un artefatto di input per l'operazione di distribuzione. Puoi utilizzare uno di questi metodi per fornire il file `imageDetail.json` nella pipeline:

- Includi il `imageDetail.json` file nella posizione di origine in modo che venga fornito nella pipeline come input per la tua azione di distribuzione blu/verde di Amazon ECS.

> **Note**
>
> Se il tuo repository di origine è un bucket Amazon S3, ricordati di comprimere il file JSON.

- Le azioni sorgente di Amazon ECR generano automaticamente un `imageDetail.json` file come elemento di input per l'azione successiva.

> **Note**
>
> Poiché l'azione di origine di Amazon ECR crea questo file, le pipeline con un'azione di origine Amazon ECR non devono fornire manualmente un file. `imageDetail.json` Per un tutorial sulla creazione di una pipeline che includa una fase sorgente di Amazon ECR, consulta. Tutorial: crea una pipeline con una sorgente Amazon ECR e una distribuzione da ECS a CodeDeploy



Il file `imageDetail.json` fornisce l'URI dell'immagine. Deve essere costruito con la seguente coppia chiave-valore.

Chiave	Valore
ImageURI	*image_URI*

imageDetail.json

Questa è la struttura JSON, che contiene l'URI dell'immagine `ACCOUNTID.dkr.ecr.us-west-2.amazonaws.com/dk-image-repo@sha256:example3`:

```
{
```

```
"ImageURI": "ACCOUNTID.dkr.ecr.us-west-2.amazonaws.com/dk-image-
repo@sha256:example3"
}
```

imageDetail.json (generated by ECR)

Un `imageDetail.json` file viene generato automaticamente dall'azione sorgente di Amazon ECR ogni volta che una modifica viene inserita nell'archivio di immagini. Le azioni di origine `imageDetail.json` generate da Amazon ECR vengono fornite come artefatto di output dall'azione di origine all'azione successiva nella pipeline.

Questa è la struttura JSON, dove il nome del repository è `dk-image-repo`, l'URI dell'immagine è `ecs-repo` e il tag dell'immagine è `latest`:

```
{
    "ImageSizeInBytes": "44728918",
    "ImageDigest":
 "sha256:EXAMPLE1122334455667788990bfea42ea2d3b8a1ee8329ba7e68694950afd3",
    "Version": "1.0",
    "ImagePushedAt": "Mon Jan 21 20:04:00 UTC 2019",
    "RegistryId": "EXAMPLE12233",
    "RepositoryName": "dk-image-repo",
    "ImageURI": "ACCOUNTID.dkr.ecr.us-west-2.amazonaws.com/dk-image-
repo@sha256:example3",
    "ImageTags": [
        "latest"
    ]
}
```

Il `imageDetail.json` file mappa l'URI dell'immagine e il nome del contenitore alla definizione del task Amazon ECS come segue:

- `ImageSizeInBytes`: le dimensioni in byte dell'immagine nel repository.
- `ImageDigest`: il file digest `sha256` del manifest delle immagini.
- `Version`: la versione dell'immagine.
- `ImagePushedAt`: la data e l'ora in cui l'immagine più recente è stata inviata al repository.
- `RegistryId`: l'ID AWS dell'account associato al registro che contiene il repository.
- `RepositoryName`: il nome del repository Amazon ECR in cui è stata inserita l'immagine.
- `ImageURI`: l'URI dell'immagine.

- `ImageTags`: il tag utilizzato per l'immagine.

Prima di creare la pipeline, utilizza la procedura descritta di seguito per impostare il file `imageDetail.json`.

1. Come parte della pianificazione della distribuzione blu/verde dell'applicazione basata su container per la pipeline, pianifica la fase di origine e la fase di compilazione, se applicabili.
2. Seleziona una delle seguenti opzioni:

   a. Se la tua pipeline ha saltato la fase di compilazione, devi creare manualmente il file JSON e caricarlo nel tuo repository di origine, ad esempio in modo che l'azione del codice sorgente possa fornire l' CodeCommitartefatto. Crea il file utilizzando un editor di testo e assegna un nome al file o utilizza il nome file predefinito `imageDetail.json`. Invia il file `imageDetail.json` al repository di origine.

   b. Se la pipeline ha una fase di compilazione, esegui questa procedura:

      i. Aggiungi un comando al file delle specifiche di compilazione che genera il file di definizioni delle immagini nel repository di origine durante la fase di compilazione. L'esempio seguente usa il comando printf per creare un file `imageDetail.json`. Elenca questo comando nella sezione `post_build` del file buildspec.yml:

      ```
      printf '{"ImageURI":"image_URI"}' > imageDetail.json
      ```

      È necessario includere il file `imageDetail.json` come un artefatto di output nel file`buildspec.yml`.

      ii. Aggiungi il file `imageDetail.json` come file dell'artefatto nel file `buildspec.yml`.

      ```
      artifacts:
        files:
          - imageDetail.json
      ```

Variables

Questa sezione è solo per riferimento. Per informazioni sulla creazione di variabili, vedere Utilizzo delle variabili.

Le variabili consentono di configurare le azioni della pipeline con valori determinati al momento dell'esecuzione della pipeline o dell'esecuzione dell'azione.

Alcuni provider di azioni producono un insieme definito di variabili. È possibile scegliere tra le chiavi variabili predefinite per tale provider di azioni, ad esempio l'ID di commit.

> **Important**
>
> Quando passate parametri segreti, non inserite direttamente il valore. Il valore viene reso come testo in chiaro ed è quindi leggibile. Per motivi di sicurezza, non utilizzate testo semplice con segreti. Ti consigliamo vivamente di AWS Secrets Manager utilizzarlo per memorizzare i segreti.

Per vedere step-by-step esempi di utilizzo delle variabili:

- Per un tutorial con una variabile a livello di pipeline che viene passata al momento dell'esecuzione della pipeline, vedi. Tutorial: utilizzare le variabili a livello di pipeline
- Per un tutorial con un'azione Lambda che utilizza le variabili di un'azione upstream (CodeCommit) e genera variabili di output, consulta. Tutorial: Utilizzo delle variabili con le azioni di richiamo Lambda
- Per un tutorial con un' AWS CloudFormation azione che fa riferimento alle variabili di output impilate di un'azione upstream CloudFormation , consulta. Tutorial: crea una pipeline che utilizza le variabili delle azioni di AWS CloudFormation distribuzione
- Per un esempio di azione di approvazione manuale con testo del messaggio che fa riferimento a variabili di output che si CodeCommit risolvono nell'ID di commit e nel messaggio di commit, vedi. Esempio: utilizzo delle variabili nelle approvazioni manuali
- Per un esempio di CodeBuild azione con una variabile di ambiente che si risolve nel nome del GitHub ramo, vedi. Esempio: utilizzare una BranchName variabile con variabili di CodeBuild ambiente
- CodeBuild le azioni producono come variabili tutte le variabili di ambiente che sono state esportate come parte della build. Per ulteriori informazioni, consulta CodeBuild azioni (variabili di output).

Limiti variabili

Per informazioni sui limiti, vedere Quote in AWS CodePipeline.

> **Note**
>
> Quando inserite la sintassi delle variabili nei campi di configurazione delle azioni, non superate il limite di 1000 caratteri per i campi di configurazione. Quando questo limite viene superato, viene restituito un errore di convalida.

Argomenti

Concetti

Questa sezione elenca i termini e i concetti chiave relativi alle variabili e agli spazi dei nomi.

Variables

Le variabili sono coppie chiave-valore che possono essere utilizzate per configurare dinamicamente le azioni nella pipeline. Attualmente esistono tre modi in cui queste variabili vengono rese disponibili:

- Esiste un insieme di variabili che sono implicitamente disponibili all'inizio di ogni esecuzione della pipeline. Questo set include attualmente `PipelineExecutionId`, l'ID dell'esecuzione della pipeline corrente.
- Le variabili a livello di pipeline vengono definite al momento della creazione della pipeline e risolte in fase di esecuzione della pipeline.

  Le variabili a livello di pipeline vengono specificate al momento della creazione della pipeline e possono fornire valori al momento dell'esecuzione della pipeline.

- Ci sono tipi di azione che producono insiemi di variabili quando vengono eseguiti. È possibile visualizzare le variabili prodotte da un'azione esaminando il `outputVariables` campo che fa parte dell'API. ListActionExecutions Per un elenco dei nomi delle chiavi disponibili per provider di azioni, vedere Variabili disponibili per le operazioni della pipeline. Per vedere quali variabili produce ogni tipo di azione, consulta CodePipeline Riferimento per la struttura delle operazioni.

Per fare riferimento a queste variabili nella configurazione dell'azione, è necessario utilizzare la sintassi di riferimento variabile con lo spazio dei nomi corretto.

Per un flusso di lavoro variabile di esempio, vedere Configurazione delle variabili .

Spazi dei nomi

Per garantire che le variabili possano essere referenziate in modo univoco, devono essere assegnate a uno spazio dei nomi. Dopo aver assegnato un set di variabili a uno spazio dei nomi, è possibile fare riferimento a queste variabili in una configurazione di azione utilizzando lo spazio dei nomi e la chiave variabile con la seguente sintassi:

```
#{namespace.variable_key}
```

Esistono tre tipi di namespace in base ai quali è possibile assegnare le variabili:

- Lo spazio dei nomi riservato codepipeline

  Questo è lo spazio dei nomi assegnato al set di variabili implicite disponibili all'inizio di ogni esecuzione della pipeline. Questo spazio dei nomi è `codepipeline`. Esempio di riferimento variabile:

  ```
  #{codepipeline.PipelineExecutionId}
  ```

- Lo spazio dei nomi delle variabili a livello di pipeline

  Questo è lo spazio dei nomi assegnato alle variabili a livello di pipeline. Lo spazio dei nomi per tutte le variabili a livello di pipeline è. `variables` Esempio di riferimento variabile:

  ```
  #{variables.variable_name}
  ```

- Spazio dei nomi assegnato all'azione

Si tratta di uno spazio dei nomi assegnato a un'azione. Tutte le variabili prodotte dall'azione rientrano in questo spazio dei nomi. Per rendere le variabili prodotte da un'azione disponibili per l'uso in una configurazione di azione downstream, è necessario configurare l'azione di produzione con uno spazio dei nomi. Gli spazi dei nomi devono essere univoci nella definizione della pipeline e non possono entrare in conflitto con i nomi degli artefatti. Ecco un riferimento di variabile di esempio per un'azione configurata con uno spazio dei nomi di `SourceVariables`.

```
#{SourceVariables.VersionId}
```

Casi d'uso per le variabili

Di seguito sono riportati alcuni dei casi d'uso più comuni per le variabili a livello di pipeline, che consentono di determinare come utilizzare le variabili per esigenze specifiche.

- Le variabili a livello di pipeline sono destinate CodePipeline ai clienti che desiderano utilizzare la stessa pipeline ogni volta con variazioni minori negli input per la configurazione dell'azione. Qualsiasi sviluppatore che avvia una pipeline aggiunge il valore della variabile nell'interfaccia utente all'avvio della pipeline. Con questa configurazione, si passano i parametri solo per quell'esecuzione.
- Con le variabili a livello di pipeline, è possibile passare input dinamici alle azioni nella pipeline. È possibile migrare le tubazioni parametrizzate verso CodePipeline senza dover mantenere versioni diverse della stessa pipeline o creare tubazioni complesse.
- È possibile utilizzare variabili a livello di pipeline per passare parametri di input che consentono di riutilizzare una pipeline ad ogni esecuzione, ad esempio quando si desidera specificare quale versione si desidera distribuire in un ambiente di produzione, in modo da non dover duplicare le pipeline.
- È possibile utilizzare una singola pipeline per distribuire risorse in più ambienti di compilazione e distribuzione. Ad esempio, per una pipeline con un CodeCommit repository, è possibile eseguire la distribuzione da una filiale specifica e da un ambiente di distribuzione di destinazione e passare CodeDeploy i parametri a livello CodeBuild di pipeline.

Configurazione delle variabili

È possibile configurare le variabili a livello di pipeline o a livello di azione nella struttura della pipeline.

Configurazione delle variabili a livello di pipeline

È possibile aggiungere una o più variabili a livello di pipeline. È possibile fare riferimento a questo valore nella configurazione delle CodePipeline azioni. È possibile aggiungere i nomi delle variabili, i valori predefiniti e le descrizioni quando si crea la pipeline. Le variabili vengono risolte al momento dell'esecuzione.

> **Note**
>
> Se non è definito un valore predefinito per una variabile a livello di pipeline, la variabile viene considerata obbligatoria. È necessario specificare le sostituzioni per tutte le variabili obbligatorie quando si avvia una pipeline, altrimenti l'esecuzione della pipeline fallirà con un errore di convalida.

Le variabili vengono fornite a livello di pipeline utilizzando l'attributo variables nella struttura della pipeline. Nell'esempio seguente, la variabile `Variable1` ha un valore di. `Value1`

```
"variables": [
     {
         "name": "Variable1",
         "defaultValue": "Value1",
         "description": "description"
     }
 ]
```

Per un esempio della struttura JSON della pipeline, vedere. Creare una pipeline in CodePipeline

Per un tutorial con una variabile a livello di pipeline che viene passata al momento dell'esecuzione della pipeline, vedi. Tutorial: utilizzare le variabili a livello di pipeline

Nota che l'utilizzo di variabili a livello di pipeline in qualsiasi tipo di azione Source non è supportato.

> **Note**
>
> Se lo spazio dei `variables` nomi è già utilizzato in alcune azioni all'interno della pipeline, è necessario aggiornare la definizione dell'azione e scegliere un altro spazio dei nomi per l'azione in conflitto.

Configurazione delle variabili a livello di azione

È possibile configurare un'azione per produrre variabili dichiarando uno spazio dei nomi per l'azione. L'azione deve essere già uno dei provider di azioni che genera variabili. In caso contrario, le variabili disponibili sono variabili a livello di pipeline.

Dichiari lo spazio dei nomi da:

- Nella pagina Modifica azione della console, immettere uno spazio dei nomi nello spazio dei nomi variabile.
- Immissione di uno spazio dei nomi nel campo parametro `namespace` nella struttura della pipeline JSON.

In questo esempio, si aggiunge il `namespace` parametro all'azione di CodeCommit origine con il nome`SourceVariables`. In questo modo viene configurata l'azione per produrre le variabili disponibili per tale provider di azioni, ad esempio `CommitId`.

```
{
    "name": "Source",
    "actions": [
        {
            "outputArtifacts": [
                {
                    "name": "SourceArtifact"
                }
            ],
            "name": "Source",
            "namespace": "SourceVariables",
            "configuration": {
                "RepositoryName": "MyRepo",
                "BranchName": "mainline",
                "PollForSourceChanges": "false"
            },
            "inputArtifacts": [],
            "region": "us-west-2",
            "actionTypeId": {
                "provider": "CodeCommit",
                "category": "Source",
                "version": "1",
                "owner": "AWS"
            },
```

```
            "runOrder": 1
        }
    ]
},
```

Successivamente, è possibile configurare l'azione downstream per utilizzare le variabili prodotte dall'azione precedente. Lo puoi fare procedendo come segue:

- Nella pagina Modifica azione della console, immettere la sintassi della variabile (per l'azione downstream) nei campi di configurazione delle azioni.
- Immissione della sintassi della variabile (per l'azione downstream) nei campi di configurazione delle azioni nella struttura della pipeline JSON

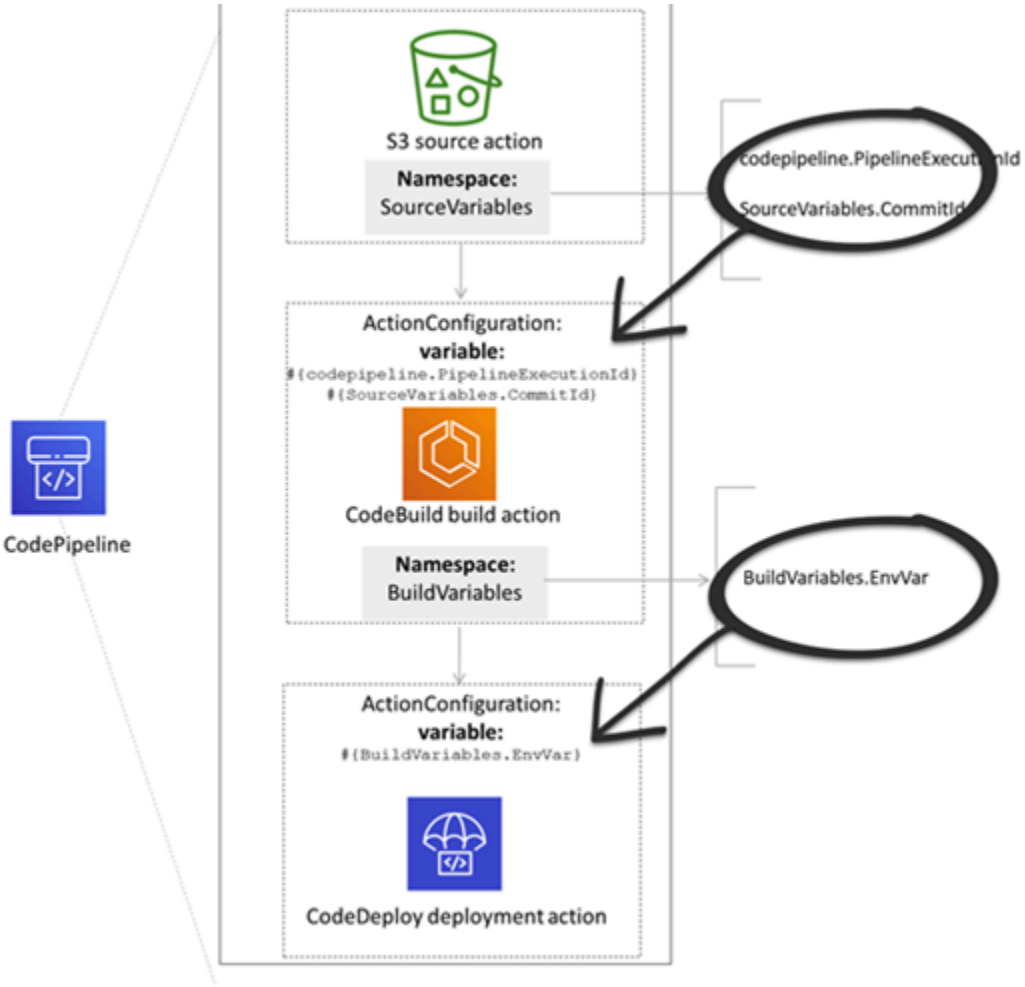
In questo esempio, il campo di configurazione dell'azione di compilazione mostra le variabili di ambiente che vengono aggiornate all'esecuzione dell'azione. L'esempio specifica lo spazio dei nomi e la variabile per l'ID di esecuzione con `#{codepipeline.PipelineExecutionId}` e lo spazio dei nomi e la variabile per l'ID commit con **`#{SourceVariables.CommitId}`**.

```
{
    "name": "Build",
    "actions": [
        {
            "outputArtifacts": [
                {
                    "name": "BuildArtifact"
                }
            ],
            "name": "Build",
            "configuration": {
                "EnvironmentVariables": "[{\"name\":\"Release_ID\",\"value\":
\"#{codepipeline.PipelineExecutionId}\",\"type\":\"PLAINTEXT\"},{\"name\":\"Commit_ID
\",\"value\":\"#{SourceVariables.CommitId}\",\"type\":\"PLAINTEXT\"}]",
                "ProjectName": "env-var-test"
            },
            "inputArtifacts": [
                {
                    "name": "SourceArtifact"
                }
            ],
            "region": "us-west-2",
            "actionTypeId": {
```

```
                "provider": "CodeBuild",
                "category": "Build",
                "version": "1",
                "owner": "AWS"
            },
            "runOrder": 1
        }
    ]
},
```

Risoluzione delle variabili

Ogni volta che un'azione viene eseguita come parte di un'esecuzione di una pipeline, le variabili che produce sono disponibili per l'uso in qualsiasi azione che è garantita che si verifichi dopo l'azione di produzione. Per utilizzare queste variabili in un'azione consuming, è possibile aggiungerle alla configurazione dell'azione consuming utilizzando la sintassi mostrata nell'esempio precedente. Prima di eseguire un'azione di consumo, CodePipeline risolve tutti i riferimenti alle variabili presenti nella configurazione prima di iniziare l'esecuzione dell'azione.



Regole per le variabili

Le seguenti regole consentono di configurare le variabili:

- Specificare lo spazio dei nomi e la variabile per un'azione tramite una nuova proprietà azione o modificando un'azione.
- Quando si utilizza la creazione guidata pipeline, la console genera uno spazio dei nomi per ogni azione creata con la procedura guidata.
- Se lo spazio dei nomi non è specificato, le variabili prodotte da tale azione non possono essere referenziate in alcuna configurazione di azione.
- Per fare riferimento alle variabili prodotte da un'azione, l'azione di riferimento deve avvenire dopo l'azione che produce le variabili. Ciò significa che è in una fase successiva rispetto all'azione che produce le variabili, o nella stessa fase, ma in un ordine di esecuzione superiore.

Variabili disponibili per le operazioni della pipeline

Il provider di azioni determina quali variabili possono essere generate dall'azione.

Per step-by-step le procedure per la gestione delle variabili, vedere. Utilizzo delle variabili

Azioni con chiavi variabili definite

A differenza di uno spazio dei nomi che puoi scegliere, le azioni seguenti utilizzano chiavi variabili che non possono essere modificate. Ad esempio, per il provider di azioni Amazon S3, sono disponibili solo le chiavi `ETag` e `VersionId` variabili.

Ogni esecuzione ha anche un set di variabili CodePipeline di pipeline generate che contengono dati sull'esecuzione, come l'ID di rilascio della pipeline. Queste variabili possono essere consumate da qualsiasi azione nella pipeline.

Argomenti

- CodePipelinevariabile ID di esecuzione
- Variabili di output delle azioni Amazon ECR
- AWS CloudFormation StackSets variabili di output delle azioni
- CodeCommit variabili di output dell'azione
- CodeStarSourceConnection variabili di output delle azioni
- GitHub variabili di output delle azioni (GitHub azione versione 1)
- Variabili di output delle azioni S3

CodePipelinevariabile ID di esecuzione

CodePipelinevariabile ID di esecuzione

Provider	Chiave variabile	Valore di esempio	Esempio di sintassi variabile
codepipeline	`PipelineExecutionId`	8abc75f0-fbf8-4f4c-bfEXAMPLE	`#{codepipeline.PipelineExecutionId}`

Variabili di output delle azioni Amazon ECR

Variabili Amazon ECR

Chiave variabile	Valore di esempio	Esempio di sintassi variabile
`ImageDigest`	sha256:EXAMPLE1122334455	`#{SourceVariables.ImageDigest}`
`ImageTag`	più recente	`#{SourceVariables.ImageTag}`
`ImageURI`	11111EXAMPLE.dkr.ecr.us-west-2.amazonaws.com/ecs-repo:latest	`#{SourceVariables.ImageURI}`
`RegistryId`	EXAMPLE12233	`#{SourceVariables.RegistryId}`
`RepositoryName`	my-image-repo	`#{SourceVariables.RepositoryName}`

AWS CloudFormation StackSets variabili di output delle azioni

AWS CloudFormation StackSets variabili

Chiave variabile	Valore di esempio	Esempio di sintassi variabile
`OperationId`	Esempio di ribuzione-2bbb-111-2bbb-11111	`#{DeployVariables.OperationId}`
`StackSetId`	my-stackset: 1111aaaa-1111-2222-2bbb-11111 esempio	`#{DeployVariables.StackSetId}`

CodeCommit variabili di output dell'azione

CodeCommit variabili

Chiave variabile	Valore di esempio	Esempio di sintassi variabile
`AuthorDate`	2019-10-29T03:32:21Z	`#{SourceVariables.AuthorDate}`
`BranchName`	sviluppo	`#{SourceVariables.BranchName}`
`CommitId`	exampleb01f91b31	`#{SourceVariables.CommitId}`
`CommitMessage`	Corretto un bug (100 KB di dimensione massima)	`#{SourceVariables.CommitMessage}`
`CommitterDate`	2019-10-29T03:32:21Z	`#{SourceVariables.CommitterDate}`
`RepositoryName`	myCodeCommitRepo	`#{SourceVariables.RepositoryName}`

CodeStarSourceConnection variabili di output delle azioni

`CodeStarSourceConnection`variabili (Bitbucket Cloud GitHub, GitHub Enterprise Repository e.com) GitLab

Chiave variabile	Valore di esempio	Esempio di sintassi variabile
`AuthorDate`	2019-10-29T03:32:21Z	`#{SourceVariables.AuthorDate}`
`BranchName`	sviluppo	`#{SourceVariables.BranchName}`
`CommitId`	exampleb01f91b31	`#{SourceVariables.CommitId}`

Chiave variabile	Valore di esempio	Esempio di sintassi variabile
`CommitMessage`	Corretto un bug (100 KB di dimensione massima)	`#{SourceVariables.CommitMessage}`
`ConnectionArn`	*arn:aws:codestar-connections:region:account-id: connection/connection-id*	`#{SourceVariables.ConnectionArn}`
`FullRepositoryName`	nome utente/ GitHubRepo	`#{SourceVariables.FullRepositoryName}`

GitHub variabili di output delle azioni (GitHub azione versione 1)

GitHub variabili (GitHub azione versione 1)

Chiave variabile	Valore di esempio	Esempio di sintassi variabile
`AuthorDate`	2019-10-29T03:32:21Z	`#{SourceVariables.AuthorDate}`
`BranchName`	principale	`#{SourceVariables.BranchName}`
`CommitId`	exampleb01f91b31	`#{SourceVariables.CommitId}`
`CommitMessage`	Corretto un bug (100 KB di dimensione massima)	`#{SourceVariables.CommitMessage}`
`CommitterDate`	2019-10-29T03:32:21Z	`#{SourceVariables.CommitterDate}`
`CommitUrl`		`#{SourceVariables.CommitUrl}`
`RepositoryName`	myGitHubpronti contro termine	`#{SourceVariables.RepositoryName}`

Variabili di output delle azioni S3

Variabili S3

Chiave variabile	Valore di esempio	Esempio di sintassi variabile
`ETag`	example28be1c3	`#{SourceVariables.ETag}`
`VersionId`	exampleta_IUQCv	`#{SourceVariables.VersionId}`

Azioni con chiavi variabili configurate dall'utente

Per CodeBuild le AWS CloudFormation azioni e Lambda, le chiavi variabili sono configurate dall'utente.

Argomenti

- CloudFormation azioni (variabili di output)
- CodeBuild azioni (variabili di output)
- Variabili di output dell'azione Lambda

CloudFormation azioni (variabili di output)

AWS CloudFormation variabili

Chiave variabile	Esempio di sintassi variabile
Per AWS CloudFormation le azioni, le variabili vengono prodotte a partire da qualsiasi valore indicato nella `Outputs` sezione di un modello di pila. Tieni presente che le uniche modalità di CloudFormation azione che generano output sono quelle che comportano la creazione o l'aggiornamento di uno stack, come la creazione dello stack, gli aggiornamenti dello stack e l'esecuzione dei set di modifiche. Le modalità di operazione corrispondenti che generano variabili sono: • CREATE_UPDATE	`#{DeployVariables.StackName}`

Chiave variabile	Esempio di sintassi variabile
• CHANGE_SET_EXECUTE • CHANGE_SET_REPLACE • REPLACE_ON_FAILURE Per ulteriori informazioni su queste modalità di azione, vedere. AWS CloudFormation Per un tutorial che mostra come creare una pipeline con un'azione di AWS CloudFormation distribuzione in una pipeline che utilizza variabili di AWS CloudFormation output, consulta. Tutorial: crea una pipeline che utilizza le variabili delle azioni di AWS CloudFormation distribuzione	

CodeBuild azioni (variabili di output)

CodeBuild variabili

Chiave variabile	Esempio di sintassi variabile
Per CodeBuild le azioni, le variabili vengono prodotte a partire da valori generati da variabili di ambiente esportate. Imposta una variabile di CodeBuild ambiente modificando l' CodeBuild azione in CodePipeline o aggiungendo la variabile di ambiente alle specifiche di build. Aggiungi le istruzioni alle specifiche di CodeBuild compilazione per aggiungere la variabile di ambiente nella sezione delle variabili esportate. Vedi env/exported-variables nella Guida per l'utente.AWS CodeBuild	`#{BuildVariables.EnvVar}`

Variabili di output dell'azione Lambda

Variabili Lambda

Chiave variabile	Esempio di sintassi variabile
L'azione Lambda produrrà come variabili tutte le coppie chiave-valore incluse nella `outputVariables` sezione della richiesta API. PutJobSuccessResult Per un tutorial con un'azione Lambda che utilizza le variabili di un'azione upstream (CodeCommit) e genera variabili di output, consulta. Tutorial: Utilizzo delle variabili con le azioni di richiamo Lambda	`#{TestVariables.testRunId}`

Lavorare con i modelli a globo nella sintassi

Quando specificate i file o i percorsi utilizzati negli artefatti della pipeline o nelle posizioni di origine, potete specificare l'artefatto in base al tipo di azione. Ad esempio, per l'azione S3, si specifica la chiave dell'oggetto S3.

Per i trigger, puoi specificare filtri. È possibile utilizzare modelli a globo per specificare i filtri. Di seguito vengono mostrati gli esempi.

Quando la sintassi è «glob», la rappresentazione String del percorso viene abbinata utilizzando un linguaggio di pattern limitato con una sintassi simile alle espressioni regolari. Per esempio:

- `*.java`Speciifica un percorso che rappresenta un nome di file che termina con .java
- `*.*`Specifica i nomi di file contenenti un punto
- `*.{java,class}`Specificate i nomi di file che terminano con .java o .class
- `foo.?`Specifica i nomi di file che iniziano con foo. e un'estensione a carattere singolo

Le seguenti regole vengono utilizzate per interpretare i pattern globulari:

- Per specificare zero o più caratteri di un componente del nome nei confini della directory, usa`*`.
- Per specificare zero o più caratteri di un componente del nome che attraversa i confini della directory, usa`**`.
- Per specificare un carattere di un componente del nome, usa`?`.
- Per evitare caratteri che altrimenti verrebbero interpretati come caratteri speciali, usate il carattere barra rovesciata ()`\`.
- Per specificare un singolo carattere da un set di caratteri, usate. `[ ]`
- Per specificare un singolo file che si trova nella radice della posizione di creazione o della posizione del repository di origine, usa`my-file.jar`.
- Per specificare un singolo file in una sottodirectory, usa `directory/my-file.jar` o. `directory/subdirectory/my-file.jar`
- Per specificare tutti i file, utilizzare`"**"`. Il pattern a `**` glob indica che deve corrispondere a un numero qualsiasi di sottodirectory.
- Per specificare tutti i file e le directory in una directory denominata, usa. `directory` `"directory/**"` Il pattern a `**` glob indica che deve corrispondere a un numero qualsiasi di sottodirectory.

- Per specificare tutti i file in una directory denominata`directory`, ma non nessuna delle sue sottodirectory, usa. `"directory/*"`
- All'interno di un'espressione tra parentesi i `\` caratteri`*`, `?` e corrispondono a se stessi. Il carattere (-) corrisponde a se stesso se è il primo carattere tra parentesi o il primo carattere dopo la negazione `!` if.
- I `{ }` caratteri sono un gruppo di modelli secondari, in cui il gruppo corrisponde se uno qualsiasi dei sottomodelli del gruppo corrisponde. Il `","` carattere viene utilizzato per separare i sottomodelli. I gruppi non possono essere annidati.

Aggiornamento delle pipeline di polling nel metodo di rilevamento delle modifiche consigliato

Se disponi di una pipeline che utilizza il polling per reagire alle modifiche all'origine, puoi aggiornarla per utilizzare il metodo di rilevamento consigliato. Per una guida alla migrazione con istruzioni per aggiornare i sondaggi in modo da utilizzare il metodo di rilevamento delle modifiche consigliato basato sugli eventi, consulta. Esegui la migrazione delle pipeline di sondaggi per utilizzare il rilevamento delle modifiche basato sugli eventi

Aggiornare un'azione di origine della GitHub versione 1 a un'azione di origine della GitHub versione 2

In AWS CodePipeline, sono supportate due versioni dell'azione di GitHub origine:

- Consigliato: l'azione della GitHub versione 2 utilizza l'autenticazione basata sull'app Github supportata da una risorsa. CodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite Installa un'applicazione AWS CodeStar Connections nell' GitHub organizzazione in modo da poter gestire l'accesso in. GitHub
- Non consigliato: l'azione della GitHub versione 1 utilizza i token OAuth per l'autenticazione GitHub e utilizza un webhook separato per rilevare le modifiche. Questo non è più il metodo consigliato.

> **Note**
>
> Le connessioni non sono disponibili nelle regioni Asia Pacifico (Hong Kong), Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Africa (Città del Capo), Medio Oriente (Bahrein), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna), Europa (Zurigo), Israele (Tel Aviv) o (Stati Uniti occidentali). AWS GovCloud Per fare riferimento ad altre azioni disponibili, consulta. Integrazioni di prodotti e servizi con CodePipeline Per considerazioni su questa azione nella regione Europa (Milano), si veda la nota inCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.

L'utilizzo dell'azione della GitHub versione 2 anziché dell'azione della GitHub versione 1 presenta alcuni importanti vantaggi:

- Con le connessioni, CodePipeline non sono più necessarie app OAuth o token di accesso personali per accedere al tuo repository. Quando crei una connessione, installi un' GitHub app che gestisce l'autenticazione nel tuo GitHub repository e consente le autorizzazioni a livello di organizzazione. È necessario autorizzare i token OAuth come utente per accedere al repository. Per ulteriori informazioni sull'accesso basato su OAuth rispetto GitHub all'accesso basato su app, consulta. GitHub https://docs.github.com/en/developers/apps/differences-between-github-apps-and-oauth-apps

- Quando gestisci le azioni della GitHub versione 2 nella CLI o CloudFormation, non devi più archiviare il tuo token di accesso personale come segreto in Secrets Manager. Non è più necessario fare riferimento dinamicamente al segreto memorizzato nella configurazione dell' CodePipeline azione. Si aggiunge invece l'ARN della connessione alla configurazione dell'azione. Per un esempio di configurazione dell'azione, vediCodeStarSourceConnection per Bitbucket Cloud, GitHub Enterprise Server GitHub, GitLab .com e GitLab azioni autogestite.
- Quando si crea una risorsa di connessione da utilizzare con l'azione della GitHub versione 2 in CodePipeline, è possibile utilizzare la stessa risorsa di connessione per associare altri servizi supportati, come CodeGuru Reviewer, al repository.
- Nella versione 2 di Github, puoi clonare i repository per accedere ai metadati git nelle CodeBuild azioni successive, mentre nella versione 1 di Github puoi solo scaricare il codice sorgente.
- Un amministratore installa l'app per gli archivi della tua organizzazione. Non è più necessario tenere traccia dei token OAuth che dipendono dalla persona che ha creato il token.

Tutte le app installate in un'organizzazione hanno accesso allo stesso set di repository. Per cambiare chi può accedere a ciascun repository, modifica la policy IAM per ogni connessione. Per un esempio, vedi Esempio: una politica ristretta per l'utilizzo di connessioni con un repository specifico.

È possibile utilizzare i passaggi descritti in questo argomento per eliminare l'azione di origine della GitHub versione 1 e aggiungere un'azione di origine della GitHub versione 2 dalla console. CodePipeline

Argomenti

- Passaggio 1: Sostituisci l' GitHub azione della versione 1
- Passaggio 2: Creare una connessione a GitHub
- Passaggio 3: Salva l'azione GitHub sorgente

Passaggio 1: Sostituisci l' GitHub azione della versione 1

Utilizza la pagina di modifica della pipeline per sostituire l'azione della versione 1 con GitHub un'azione della versione 2 GitHub .

Per sostituire l'azione della versione 1 GitHub

1. Accedi alla CodePipeline console.

2. Scegli la tua pipeline e scegli Modifica. Scegli Modifica fase nella fase di origine. Viene visualizzato un messaggio che consiglia di aggiornare l'azione.
3. Nel provider Action, scegli GitHub (Versione 2).
4. Esegui una di queste operazioni:

   - In Connessione, se non hai già creato una connessione con il tuo provider, scegli Connetti a GitHub. Procedi al Passaggio 2: Crea una connessione a GitHub.
   - In Connessione, se hai già creato una connessione al tuo provider, scegli la connessione. Procedi al passaggio 3: Salva l'azione di origine per la tua connessione.

Passaggio 2: Creare una connessione a GitHub

Dopo aver scelto di creare la connessione, viene GitHub visualizzata la pagina Connetti a.

Per creare una connessione a GitHub

1. Nelle impostazioni di GitHub connessione, il nome della connessione viene visualizzato in Nome connessione.

   In GitHub App, scegli l'installazione di un'app o scegli Installa una nuova app per crearne una.

   > **Note**
   >
   > È sufficiente installare una sola app per tutte le connessioni a un provider specifico. Se hai già installato l' GitHub app, sceglila e salta questo passaggio.

2. Se GitHub viene visualizzata la pagina di autorizzazione, accedi con le tue credenziali e scegli di continuare.
3. Nella pagina di installazione dell'app, un messaggio indica che l' AWS CodeStar app sta tentando di connettersi al tuo GitHub account.

   > **Note**
   >
   > L'app viene installata una sola volta per ogni GitHub account. Se hai già installato l'app, puoi scegliere Configure (Configura) per passare a una pagina di modifica per l'installazione dell'app oppure è possibile utilizzare il pulsante Indietro per tornare alla console.

4. Nella AWS CodeStar pagina di installazione, scegli Installa.
5. Nella GitHub pagina Connect to, viene visualizzato l'ID di connessione per la nuova installazione. Scegli Connetti.

Passaggio 3: Salva l'azione GitHub sorgente

Completa gli aggiornamenti nella pagina Modifica azione per salvare la nuova azione sorgente.

Per salvare l'azione GitHub sorgente

1. In Repository, inserisci il nome del tuo repository di terze parti. In Branch, inserisci il ramo in cui desideri che la pipeline rilevi le modifiche all'origine.

   > **Note**
   >
   > In Repository, digita `owner-name/repository-name` come mostrato in questo esempio:
   >
   > ```
   > my-account/my-repository
   > ```

2. In Formato di output degli artefatti, scegliete il formato per gli artefatti.

   - Per memorizzare gli artefatti di output derivanti dall' GitHub azione utilizzando il metodo predefinito, scegliete predefinito. CodePipeline L'azione accede ai file dal GitHub repository e archivia gli artefatti in un file ZIP nell'archivio degli artefatti della pipeline.
   - Per archiviare un file JSON contenente un riferimento URL al repository in modo che le operazioni downstream possano eseguire direttamente comandi Git, scegliere Full clone (Clone completo). Questa opzione può essere utilizzata solo dalle azioni a valle. CodeBuild

     Se scegli questa opzione, dovrai aggiornare le autorizzazioni per il tuo ruolo di CodeBuild Project Service come mostrato in. Aggiungi le autorizzazioni per le connessioni a Bitbucket, Enterprise Server o.com CodeBuild GitClone GitHub GitHub GitLab Per un tutorial che mostra come usare l'opzione Full clone, consulta. Tutorial: usa il clone completo con una sorgente di GitHub pipeline

3. In Output Artifacts, puoi mantenere il nome dell'artefatto di output per questa azione, ad esempio. `SourceArtifact` Scegli Fine per chiudere la pagina Modifica azione.

4. Scegliete Fine per chiudere la pagina di modifica dello stage. Scegliete Salva per chiudere la pagina di modifica della pipeline.

Quote in AWS CodePipeline

CodePipeline prevede quote per il numero di pipeline, fasi, azioni e webhook che un account può avere in ogni regione. AWS AWS Queste quote si applicano per regione e possono essere aumentate. Per richiedere un aumento, utilizza la console Support Center.

L'elaborazione delle richieste di aumento delle quote può richiedere fino a due settimane.

Risorsa	Di default
Periodo di tempo prima della scadenza di un'operazione (Si tratta di timeout configurabili. Vedi la tabella seguente per i timeout non configurabili)	AWS CloudFormation azione di implement azione: 3 giorni CodeDeploy e azioni di implementazione CodeDeploy ECS (blu/verde): 5 giorni AWS Lambda azione di richiamo: 24 ore **Note** Mentre l'azione è in esecuzione, contatta CodePipeline periodica mente Lambda per uno stato. La funzione Lambda risponde con uno stato in cui l'esecuzione dell'azione è riuscita, non riuscita o in corso. Se la funzione Lambda non ha inviato alcuna risposta dopo 20 minuti, l'azione scade. Se, durante i 20 minuti, la funzione Lambda ha risposto che l'azione è ancora in corso, CodePipeline riavvia il timer da 20 minuti e riprova. Se non riesce dopo 24 ore, CodePipeline imposta lo stato di azione Lambda invoke su Failed. Lambda ha un timeout separato per le funzioni Lambda che non

Risorsa	Di default
	è correlato al timeout dell'azione. CodePipeline Azione di distribuzione di Amazon S3:90 minuti **Note** Se il caricamento su S3 scade durante la distribuzione di un file ZIP di grandi dimensioni, l'azione ha esito negativo e viene generato un errore di timeout. Prova a suddividere il file ZIP in file più piccoli. Azione di approvazione manuale. Timeout predefinito a livello di account: 7 giorni. **Note** Il timeout predefinito per l'azione di approvazione manuale può essere sostituito per un'azione specifica nella pipeline ed è configurabile fino a 86400 minuti (60 giorni) con un valore minimo di 5 minuti. Per ulteriori informazioni, consulta l'API Reference. ActionDeclarationCodePipeline Se configurato, questo timeout viene applicato all'azione. In caso contrario, viene utilizzato il valore predefinito a livello di account.

Risorsa	Di default
	Tutte le altre operazioni: 1 ora **Note** Il timeout dell'azione di distribuzione di Amazon ECS è configurabile fino a un'ora (il timeout predefinito).
Numero massimo di pipeline totali per regione in un account AWS	1000 **Note** Le pipeline configurate per il polling o il rilevamento delle modifiche basato su eventi vengono conteggiate per questa quota.
Numero massimo di condotte impostate per il sondaggio delle modifiche alle fonti, per regione AWS	300 **Note** Questa quota è fissa e non può essere modificata. Se raggiungi il limite per il polling delle pipeline, puoi comunque configurare pipeline aggiuntive che utilizzano il rilevamento delle modifiche basato sugli eventi. Per ulteriori informazioni, consulta Azioni all'origine e metodi di rilevamento delle modifiche.[1]
Numero massimo di webhook per regione in un account AWS	300

Risorsa	Di default
Numero di azioni personalizzate per regione in un account AWS	50

[1]In base al provider di origine, utilizzare le istruzioni seguenti per aggiornare le pipeline di polling e utilizzare il rilevamento delle modifiche basato su eventi:

- Per aggiornare un'azione CodeCommit sorgente, consultaMigrazione delle pipeline di polling (o sorgente Amazon CodeCommit S3) (console).
- Per aggiornare un'azione sorgente di Amazon S3, consulta. Migrazione delle pipeline di polling (o sorgente Amazon CodeCommit S3) (console)
- Per aggiornare un'azione di GitHub origine, consultaMigrazione delle pipeline di polling ai webhook (azioni di origine della GitHub versione 1) (console).

Le seguenti quote AWS CodePipeline si applicano alla disponibilità della regione, ai vincoli di denominazione e alle dimensioni consentite degli artefatti. Queste quote sono fisse e non possono essere modificate.

Per un elenco degli endpoint di CodePipeline servizio per ogni regione, consulta AWS CodePipeline Endpoint e quote nel Riferimento generale.AWS

Per informazioni sui requisiti strutturali, consulta CodePipeline riferimento alla struttura della tubazione.

AWS Regioni in cui è possibile creare una pipeline	Stati Uniti orientali (Ohio) Stati Uniti orientali (Virginia settentrionale) Stati Uniti occidentali (California settentrionale) US West (Oregon) Canada (Centrale) Europa (Francoforte) Europa (Zurigo) *

	Israele (Tel Aviv)
	Europa (Irlanda)
	Europa (Londra)
	Europa (Milano) *
	Europa (Parigi)
	Europa (Spagna)
	Europa (Stoccolma)
	Africa (Città del Capo) *
	Asia Pacifico (Hong Kong) *
	Asia Pacific (Hyderabad)
	Asia Pacifico (Mumbai)
	Asia Pacifico (Tokyo)
	Asia Pacifico (Seoul)
	Asia Pacifico (Osaka-Locale)
	Asia Pacifico (Singapore)
	Asia Pacifico (Sydney)
	Asia Pacifico (Giacarta)
	Asia Pacifico (Melbourne)
	Sud America (San Paolo)
	Medio Oriente (Bahrein) *
	Medio Oriente (Emirati Arabi Uniti)
	AWS GovCloud (Stati Uniti occidentali)

	AWS GovCloud (Stati Uniti orientali)
Caratteri consentiti in un nome operazione	I nomi di operazioni non possono superare i 100 caratteri. I caratteri consentiti includono: Lettere minuscole dalla a alla z, incluse. Lettere maiuscole dalla A alla Z, incluse. Numeri da 0 a 9 inclusi. Caratteri speciali . (punto), @ (chiocciola), - (segno meno) e _ (carattere di sottolineatura). Tutti gli altri caratteri, ad esempio spazi, non sono consentiti.
Caratteri consentiti nei tipi di operazione	I nomi dei tipi di operazione non possono superare i 25 caratteri. I caratteri consentiti includono: Lettere minuscole dalla a alla z, incluse. Lettere maiuscole dalla A alla Z, incluse. Numeri da 0 a 9, inclusi. Caratteri speciali . (punto), @ (chiocciola), - (segno meno) e _ (carattere di sottolineatura). Tutti gli altri caratteri, ad esempio spazi, non sono consentiti.

Caratteri consentiti nei nomi degli artefatti	I nomi degli Artifact non possono superare i 100 caratteri. I caratteri consentiti includono: Lettere minuscole dalla a alla z, incluse. Lettere maiuscole dalla A alla Z, incluse. Numeri da 0 a 9 inclusi. Caratteri speciali - (segno meno) e _ (trattino basso). Tutti gli altri caratteri, ad esempio spazi, non sono consentiti.
Caratteri consentiti nei nomi di operazioni partener	I nomi delle azioni dei partner devono seguire le stesse convenzioni e restrizioni di denominaz ione dei nomi delle altre azioni. CodePipeline Nello specifico, non possono superare 100 caratteri. I caratteri consentiti includono: Lettere minuscole dalla a alla z, incluse. Lettere maiuscole dalla A alla Z, incluse. Numeri da 0 a 9, inclusi. Caratteri speciali . (punto), @ (chiocciola), - (segno meno) e _ (carattere di sottolineatura). Tutti gli altri caratteri, ad esempio spazi, non sono consentiti.

Caratteri consentiti in un nome della pipeline	I nomi delle pipeline non possono superare i 100 caratteri. I caratteri consentiti includono: Lettere minuscole dalla a alla z, incluse. Lettere maiuscole dalla A alla Z, incluse. Numeri da 0 a 9, inclusi. Caratteri speciali . (punto), @ (chiocciola), - (segno meno) e _ (carattere di sottolineatura). Tutti gli altri caratteri, ad esempio spazi, non sono consentiti.
Caratteri consentiti in un nome fase	I nomi di fase non possono superare i 100 caratteri. I caratteri consentiti includono: Lettere minuscole dalla a alla z, incluse. Lettere maiuscole dalla A alla Z, incluse. Numeri da 0 a 9, inclusi. Caratteri speciali . (punto), @ (chiocciola), - (segno meno) e _ (carattere di sottolineatura). Tutti gli altri caratteri, ad esempio spazi, non sono consentiti.
Periodo di tempo prima della scadenza di un'operazione	CodeBuild azione di costruzione e azione di test: 8 ore Azioni personalizzate: 24 ore Azione di invocazione Step Functions: 7 giorni

Lunghezza massima della chiave di configurazione dell'azione (ad esempio, le chiavi di CodeBuild configurazione sono `ProjectName` `PrimarySource` , `eEnvironmentVariables`)	50 caratteri
Lunghezza massima del valore di configurazione dell'azione (ad esempio, il valore della `RepositoryName` configurazione nella configurazione dell' CodeCommit azione deve essere inferiore a 1000 caratteri): `"RepositoryName": "my-repo-name-less-than-1000-characters"`)	1000 caratteri
Numero massimo di azioni per pipeline	500
Numero massimo di esecuzioni simultanee di pipeline per pipeline (modalità QUEUED|PARALLEL)	50
Numero massimo di esecuzioni di azioni simultanee per esecuzione della pipeline in modalità PARALLEL	5
Numero massimo di file per un oggetto Amazon S3	100.000
Numero massimo di mesi durante i quali vengono conservate le informazioni sulla cronologia di esecuzione della pipeline	12
Numero massimo di azioni parallele in una fase	50
Numero massimo di azioni sequenziali in una fase	50

Dimensione massima degli artefatti in una fase di origine	Artefatti archiviati nei bucket Amazon S3:7 GB Artefatti archiviati nei nostri repository: 1 GB CodeCommit GitHub Eccezione: se si utilizza AWS Elastic Beanstalk per distribuire applicazioni, la dimensione massima degli artefatti è sempre di 512 MB. Eccezione: se si utilizza AWS CloudForm ation per distribuire applicazioni, la dimensione massima degli artefatti è sempre di 256 MB. Eccezione: se utilizzi l'operazione `CodeDeplo yToECS` per la distribuzione delle applicazioni, la dimensione massima degli artefatti è sempre di 3 MB.
Dimensione massima del file JSON delle definizioni delle immagini utilizzato nelle pipeline che distribuiscono contenitori e immagini Amazon ECS	100 KB
Dimensione massima degli artefatti di input per le azioni AWS CloudFormation	256 MB
Dimensione massima degli artefatti di input per l'operazione `CodeDeployToECS`	3 MB
Dimensione massima degli artefatti di input per l'operazione `Step Functions`	L'azione Step Functions viene eseguita su Lambda e pertanto ha quote di dimensione degli artefatti uguali alle quote di dimension e degli artefatti per le funzioni Lambda. Per ulteriori informazioni, consulta le quote Lambda nella Lambda Developer Guide.

La dimensione massima dell'oggetto JSON che può essere archiviato nella proprietà `ParameterOverrides`	Per un'azione di CodePipeline distribuzione con AWS CloudFormation come provider, la `ParameterOverrides` proprietà viene utilizzata per memorizzare un oggetto JSON che specifica i valori per il file di configurazione del modello. AWS CloudFormation L'oggetto JSON che può essere archiviato nella proprietà `ParameterOverrides` deve avere una dimensione massima di 1 kilobyte.
Numero di azioni in una fase	Minimo 1, massimo 50
Numero di artefatti consentiti per ogni azione	Per il numero di artefatti di input e output consentiti per ogni azione, vedere la sezione Numero di artefatti di input e output per ogni tipo di operazione
Numero di fasi in una pipeline	Minimo 2, massimo 50
Tag della pipeline	I tag rispettano la distinzione tra maiuscole e minuscole. Numero massimo di 50 per ogni risorsa.

Nomi delle chiavi di tag della pipeline	Qualsiasi combinazione di lettere Unicode, numeri, spazi e caratteri consentiti in UTF-8 con una lunghezza compresa tra 1 e 128 caratteri. I caratteri consentiti sono + - = . _ : / @ I nomi delle chiavi di tag devono essere univoci e ogni chiave può avere un solo valore. Un tag non può: • inizia con: AWS • contenere solo spazi • terminare con uno spazio • contenere emojis o uno dei seguenti caratteri : ? ^ * [\ ~ ! # $ % & * ( ) > < | " '
Valori dei tag della pipeline	Qualsiasi combinazione di lettere Unicode, numeri, spazi e caratteri consentiti in UTF-8 con una lunghezza compresa tra 1 e 256 caratteri. I caratteri consentiti sono + - = . _ : / @ Una chiave può avere un solo valore, ma molte chiavi possono avere lo stesso valore. Un tag non può: • inizia con AWS: • contenere solo spazi • terminare con uno spazio • contenere emojis o uno dei seguenti caratteri : ? ^ * [\ ~ ! # $ % & * ( ) > < | " '

Trigger	Esiste un massimo di 50 trigger in una definizione di pipeline nella configurazione `push` e `pull request`. Sono disponibili al massimo tre filtri per trigger push e pull request. **Note** I duplicati per i filtri nello stesso array di tipi di eventi non sono consentiti. Puoi aggiungere fino a 8 modelli di inclusione e 8 di esclusione, rami e percorsi di file per ogni tipo di evento (push, pull request). I caratteri consentiti nei valori dei modelli includono tutti i tipi di caratteri. Per i modelli di inclusione ed esclusione, la lunghezza massima è di 255 caratteri. Per i nomi dei tag, la lunghezza massima è di 255 caratteri. La dimensione massima dell'`triggers`array non deve superare i 200 KB

Filtri Trigger	Percorsi dei file: • Numero di motivi: puoi aggiungere fino a 8 modelli di inclusione e 8 di esclusione. • Dimensione del motivo: la dimensione di ogni motivo di inclusione o esclusione può contenere fino a 255 caratteri. Filiali: • Numero di motivi: puoi aggiungere fino a 8 modelli di inclusione e 8 di esclusione. • Dimensione del motivo: la dimensione di ogni motivo di inclusione o esclusione può contenere fino a 255 caratteri. Richieste pull: Filiali: • Numero di motivi: puoi aggiungere fino a 8 modelli di inclusione e 8 di esclusione. • Dimensione del motivo: la dimensione di ogni motivo di inclusione o esclusione può contenere fino a 255 caratteri.
Unicità dei nomi	All'interno di un singolo AWS account, ogni pipeline creata in una AWS regione deve avere un nome univoco. Puoi riutilizzare i nomi per le pipeline in diverse regioni. AWS I nomi delle fasi devono essere univoci all'interno di una pipeline. I nomi delle operazioni devono essere univoci all'interno di una fase.

Quote per variabili di output e namespace	Esiste un limite massimo di 122880 byte per tutte le variabili di output combinate per una particolare azione. Esiste un limite massimo di 100 KB per la configurazione totale dell'azione risolta per una determinata azione. I nomi delle variabili di output sono maiuscole e minuscole. Gli spazi dei nomi fanno distinzione tra maiuscole e minuscole. I caratteri consentiti includono: • Lettere minuscole dalla a alla z, incluse. • Lettere maiuscole dalla A alla Z, incluse. • Numeri da 0 a 9, inclusi. • Caratteri speciali ^ (accento circonflesso), @ (chiocciola), - (segno meno), _ (caratter e di sottolineatura), [(parentesi sinistra),] (parentesi destra), * (asterisco), $ (simbolo del dollaro). Tutti gli altri caratteri, ad esempio spazi, non sono consentiti.

Quote per le variabili a livello di pipeline	È previsto un massimo di 50 variabili a livello di pipeline per pipeline. I nomi delle variabili per le variabili a livello di pipeline devono essere: • Lunghezza massima di 128 caratteri • Lettere minuscole dalla a alla z, incluse. • Lettere maiuscole dalla A alla Z, incluse. • Numeri da 0 a 9, inclusi. • Caratteri speciali `@\-_]+` Tutti gli altri caratteri, ad esempio spazi, non sono consentiti. Per i valori variabili, la lunghezza massima è di 1000 caratteri Per i valori variabili, sono consentiti tutti i caratteri. Per le descrizioni delle variabili, la lunghezza massima è di 200 caratteri.

* È necessario abilitare questa regione prima di poterla utilizzare.

Appendice A: azioni di origine della GitHub versione 1

Questa appendice fornisce informazioni sulla versione 1 dell'azione in. GitHub CodePipeline

> **Note**
>
> Sebbene non sia consigliabile utilizzare l'azione della GitHub versione 1, le pipeline esistenti con l'azione della GitHub versione 1 continueranno a funzionare senza alcun impatto. Per una pipeline con un'azione della GitHub versione 1, CodePipeline utilizza token basati su OAuth per connettersi al tuo repository. GitHub Al contrario, l' GitHub azione (versione 2) utilizza una risorsa di connessione per associare AWS risorse al repository. GitHub La risorsa di connessione utilizza token basati su app per connettersi. Per ulteriori informazioni sull'aggiornamento della pipeline all' GitHub azione consigliata che utilizza una connessione, consulta. Aggiornare un'azione di origine della GitHub versione 1 a un'azione di origine della GitHub versione 2 Per ulteriori informazioni sull'accesso basato su OAuth rispetto GitHub all'accesso basato su app GitHub , consulta. https://docs.github.com/en/developers/apps/differences-between-github-apps-and-oauth-apps

Per l'integrazione con GitHub, CodePipeline utilizza un'applicazione GitHub OAuth per la pipeline. CodePipelineutilizza i webhook per gestire il rilevamento delle modifiche per la pipeline con l'azione di origine della versione 1. GitHub

> **Note**
>
> Quando configuri un'azione di origine della GitHub versione 2 in AWS CloudFormation, non includi alcuna informazione sul GitHub token né aggiungi una risorsa webhook. Si configura una risorsa di connessione come illustrato AWS::CodeStarConnections::Connectionnella Guida per l' AWS CloudFormation utente.

Questo riferimento contiene le seguenti sezioni relative all'azione della GitHub versione 1:

- Per informazioni su come aggiungere un'azione di origine e un webhook della GitHub versione 1 a una pipeline, vedere. Aggiungere un'azione di origine della versione 1 GitHub

- Per informazioni sui parametri di configurazione e sugli snippet YAML/JSON di esempio per un'azione di origine della versione 1, consulta. GitHub GitHub riferimento alla struttura delle azioni di origine della versione 1

Argomenti

Aggiungere un'azione di origine della versione 1 GitHub

Aggiungete le azioni di origine della GitHub versione 1 nei seguenti CodePipeline modi:

- Utilizzando la CodePipeline console Create pipeline wizard (Creazione di una pipeline (console)) o la pagina Modifica azione per scegliere l'opzione del GitHubprovider. La console crea un webhook che avvia la pipeline quando cambia la fonte.
- Utilizzo della CLI per aggiungere la configurazione dell'azione per l'`GitHub`azione e creare risorse aggiuntive come segue:
  - Utilizzo dell'`GitHub`esempio di configurazione dell'azione in GitHub riferimento alla struttura delle azioni di origine della versione 1 per creare l'azione come mostrato inCreazione di una pipeline (CLI).
  - Disattivazione dei controlli periodici e creazione manuale del rilevamento delle modifiche, poiché per impostazione predefinita il metodo di rilevamento delle modifiche prevede l'avvio della pipeline mediante il polling della fonte. Per le azioni della versione 1, si esegue la migrazione della pipeline di polling verso i webhook. GitHub

GitHub riferimento alla struttura delle azioni di origine della versione 1

> **Note**
>
> Sebbene non sia consigliabile utilizzare l'azione della GitHub versione 1, le pipeline esistenti con l'azione della GitHub versione 1 continueranno a funzionare senza alcun impatto.
> Per una pipeline con un'azione sorgente della GitHub GitHub versione 1, CodePipeline utilizza token basati su OAuth per connettersi al tuo repository. GitHub Al contrario, la nuova

GitHub azione (versione 2) utilizza una risorsa di connessione per associare AWS risorse al repository. GitHub La risorsa di connessione utilizza token basati su app per connettersi. Per ulteriori informazioni sull'aggiornamento della pipeline all' GitHub azione consigliata che utilizza una connessione, consulta. Aggiornare un'azione di origine della GitHub versione 1 a un'azione di origine della GitHub versione 2

Attiva la pipeline quando viene effettuato un nuovo commit nel GitHub repository e nel ramo configurati.

Per l'integrazione con GitHub, CodePipeline utilizza un'applicazione OAuth o un token di accesso personale per la pipeline. Se utilizzi la console per creare o modificare la pipeline, CodePipeline crea un GitHub webhook che avvia la pipeline quando si verifica una modifica nel repository.

È necessario aver già creato un GitHub account e un repository prima di connettere la pipeline tramite un'azione. GitHub

Se desideri limitare l'accesso ai CodePipeline repository, crea un GitHub account e concedi all'account l'accesso solo ai repository con cui desideri effettuare l'integrazione. CodePipeline Usa quell'account quando configuri l'uso dei GitHub repository CodePipeline per le fasi di origine nelle pipeline.

Per ulteriori informazioni, consulta la documentazione per gli GitHub sviluppatori sul GitHub sito Web.

Argomenti

Tipo di operazione

- Categoria: `Source`

- Proprietario: `ThirdParty`
- Provider: `GitHub`
- Versione: `1`

Parametri di configurazione

Owner

Campo obbligatorio: sì

Il nome dell' GitHub utente o dell'organizzazione proprietaria del GitHub repository.

Repo

Campo obbligatorio: sì

Il nome del repository in cui devono essere rilevate le modifiche di origine.

Ramo

Campo obbligatorio: sì

Il nome del ramo in cui devono essere rilevate le modifiche di origine.

O AuthToken

Campo obbligatorio: sì

Rappresenta il token di GitHub autenticazione che CodePipeline consente di eseguire operazioni sul GitHub repository. La voce viene sempre visualizzata come una maschera di quattro asterischi. Rappresenta uno dei seguenti valori:

- Quando si utilizza la console per creare la pipeline, CodePipeline utilizza un token OAuth per registrare la connessione. GitHub
- Quando usi il AWS CLI per creare la pipeline, puoi passare il tuo token di accesso GitHub personale in questo campo. Sostituisci gli asterischi (****) con il token di accesso personale copiato da. GitHub Quando si esegue `get-pipeline` per visualizzare la configurazione dell'operazione, per questo valore viene visualizzata la maschera con quattro asterischi.
- Quando si utilizza un AWS CloudFormation modello per creare la pipeline, è necessario innanzitutto archiviare il token come indirizzo segreto. AWS Secrets Manager Il valore di questo campo viene incluso come riferimento dinamico al segreto memorizzato in Secrets Manager, ad esempio`{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}`.

Per ulteriori informazioni sugli GitHub ambiti, consulta lo GitHub Developer API Reference sul GitHub sito Web.

PollForSourceChanges

Campo obbligatorio: no

`PollForSourceChanges`controlla se interroga CodePipeline il GitHub repository per verificare la presenza di modifiche all'origine. Si consiglia di utilizzare i webhook per rilevare le modifiche all'origine. Per ulteriori informazioni sulla configurazione dei webhook, vedere Migrazione delle pipeline di polling ai webhook (azioni sorgente GitHub versione 1) (CLI) o Pipeline di aggiornamento per gli eventi push (azioni di origine della GitHub versione 1) (modello)AWS CloudFormation.

> **Important**
>
> Se si intende configurare webhook, è necessario impostare `PollForSourceChanges` su `false` per evitare esecuzioni di pipeline duplicate.

I valori validi per questo parametro sono:

- `True`: Se impostato, analizza il repository per CodePipeline verificare se sono state apportate modifiche all'origine.

  > **Note**
  >
  > Se si omette`PollForSourceChanges`, per CodePipeline impostazione predefinita esegue il polling del repository per verificare la presenza di modifiche all'origine. Questo comportamento è lo stesso se `PollForSourceChanges` è impostato su `true`.

- `False`: se impostata, CodePipeline non esegue il polling del repository per verificare la presenza di modifiche all'origine. Utilizzare questa impostazione se si intende configurare un webhook per rilevare le modifiche all'origine.

Input artifact (Artefatti di input)

- Numero di artefatti: `0`
- Descrizione: gli artefatti di input non si applicano a questo tipo di azione.

Artefatti di output

- Numero di artefatti: 1
- Descrizione: l'artefatto di output di questa azione è un file ZIP che contiene il contenuto del repository configurato e del ramo al commit specificato come revisione di origine per l'esecuzione della pipeline. Gli artefatti generati dal repository sono gli artefatti di output dell'azione. GitHub L'ID di commit del codice sorgente viene visualizzato CodePipeline come revisione del codice sorgente per l'esecuzione della pipeline attivata.

Variabili di output

Quando è configurata, questa azione produce variabili che possono essere referenziate dalla configurazione dell'azione di un'azione downstream nella pipeline. Questa azione produce variabili che possono essere viste come variabili di output, anche se l'azione non ha uno spazio dei nomi. È possibile configurare un'azione con uno spazio dei nomi per rendere tali variabili disponibili per la configurazione delle azioni downstream.

Per ulteriori informazioni sulle variabili in CodePipeline, vedere. Variables

CommitId

L'ID di GitHub commit che ha attivato l'esecuzione della pipeline. Gli ID di commit sono l'SHA completo del commit.

CommitMessage

Il messaggio di descrizione, se presente, associato al commit che ha attivato l'esecuzione della pipeline.

CommitUrl

L'indirizzo URL per il commit che ha attivato la pipeline.

RepositoryName

Il nome del GitHub repository in cui è stato effettuato il commit che ha attivato la pipeline.

BranchName

Il nome del ramo del GitHub repository in cui è stata apportata la modifica all'origine.

AuthorDate

La data in cui il commit è stato creato, in formato timestamp.

Per ulteriori informazioni sulla differenza tra un autore e un committer in Git, vedere Visualizzazione della cronologia del commit in Pro Git di Scott Chacon e Ben Straub.

CommitterDate

La data in cui è stato eseguito il commit, in formato timestamp.

Per ulteriori informazioni sulla differenza tra un autore e un committer in Git, vedere Visualizzazione della cronologia del commit in Pro Git di Scott Chacon e Ben Straub.

Dichiarazione di operazione (esempio GitHub)

YAML

```
Name: Source
Actions:
  - InputArtifacts: []
    ActionTypeId:
      Version: '1'
      Owner: ThirdParty
      Category: Source
      Provider: GitHub
    OutputArtifacts:
      - Name: SourceArtifact
    RunOrder: 1
    Configuration:
      Owner: MyGitHubAccountName
      Repo: MyGitHubRepositoryName
      PollForSourceChanges: 'false'
      Branch: main
      OAuthToken: '{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}'
    Name: ApplicationSource
```

JSON

```
{
    "Name": "Source",
    "Actions": [
        {
            "InputArtifacts": [],
            "ActionTypeId": {
                "Version": "1",
```

```
                "Owner": "ThirdParty",
                "Category": "Source",
                "Provider": "GitHub"
            },
            "OutputArtifacts": [
                {
                    "Name": "SourceArtifact"
                }
            ],
            "RunOrder": 1,
            "Configuration": {
                "Owner": "MyGitHubAccountName",
                "Repo": "MyGitHubRepositoryName",
                "PollForSourceChanges": "false",
                "Branch": "main",
                "OAuthToken":
 "{{resolve:secretsmanager:MyGitHubSecret:SecretString:token}}"
            },
            "Name": "ApplicationSource"
        }
    ]
},
```

Connessione a GitHub (OAuth)

La prima volta che usi la console per aggiungere un GitHub repository a una pipeline, ti viene chiesto di autorizzare CodePipeline l'accesso ai tuoi repository. Il token richiede i seguenti ambiti: GitHub

- L'ambito `repo`, utilizzato per il controllo completo per leggere ed estrarre artefatti da repository pubblici e privati in una pipeline.
- L'ambito `admin:repo_hook`, utilizzato per il controllo completo di hook di repository.

Quando si utilizza la CLI o un AWS CloudFormation modello, è necessario fornire il valore per un token di accesso personale in cui è già stato creato. GitHub

Consulta anche

Le risorse correlate seguenti possono essere utili durante l'utilizzo di questa operazione.

- Risorsa di riferimento per la Guida per AWS CloudFormation l'utente AWS::CodePipeline::Webhook: include definizioni di campo, esempi e frammenti per la risorsa in. AWS CloudFormation
- Risorsa di riferimento per l'AWS CloudFormation User Guide AWS::CodeStar::GitHub Repository: include definizioni di campo, esempi e frammenti per la risorsa in. AWS CloudFormation
- Tutorial: crea una pipeline con cui creare e testare la tua app Android AWS Device Farm— Questo tutorial fornisce un esempio di file di specifiche di build e un'applicazione di esempio per creare una pipeline con un sorgente. GitHub Crea e testa un'app Android con e. CodeBuild AWS Device Farm

AWS CodePipeline Cronologia dei documenti della Guida dell'utente

La tabella seguente descrive le modifiche importanti in ogni versione della Guida per l' CodePipeline utente. Per ricevere notifiche sugli aggiornamenti di questa documentazione, puoi abbonarti a un feed RSS.

- Versione API: 09-07-2015
- Ultimo aggiornamento della documentazione: 7 maggio 2024

Modifica	Descrizione	Data
Aggiornamenti all'azione S3 source per aggiungere una nuova opzione per le sostituzioni dei sorgenti	Una nuova opzione per le sostituzioni all'origine denominata `S3_OBJECT_KEY` è disponibile per l'azione di origine. `S3` È stato aggiunto un nuovo `AllowOverrideForS3ObjectKey` parametro per l'azione di origine di S3. Consulta la pagina di riferimento sull'azione del codice sorgente di Amazon S3 e avvia una pipeline con un override della revisione del codice sorgente.	7 giugno 2024
Aggiornamenti all'azione S3 di origine per aggiungere nuove variabili di output	Nuove variabili di output `ObjectKey` denominate `BucketName` e sono disponibili per l'azione `S3` di origine. Consulta la pagina di riferimento sull'azione dei sorgenti di Amazon S3.	5 giugno 2024

Support per l'analisi dei costi per lo spostamento di tubazioni di tipo V1 in pipeline di tipo V2	Lo `PipelineCostAnalyzer.py` script è stato aggiunto per eseguire l'analisi dei costi di spostamento delle tubazioni di tipo V1 esistenti in tubazioni di tipo V2. Vedi Qual è il tipo di pipeline giusto per me? .	30 maggio 2024
Aggiornamenti alle azioni `CloudFormationStackSet` e `CloudFormationStackInstances`	Il `CallAs` parametro è stato aggiunto per l'`CloudFormationStackInstances` azione `CloudFormationStackSet` and. Vedi la pagina di riferimento dell'azione.	2 maggio 2024
Support per i rollback a livello di fase	È possibile ripristinare manualmente o automaticamente una fase a una precedente esecuzione riuscita della pipeline per la fase. Vedi Configurazione del rollback della fase e concetti.	26 aprile 2024
Aggiornamenti alla disponibilità delle aree geografiche per StackSets le azioni Step Functions	Le azioni StackSets and Step Functions sono ora disponibili in tutte le regioni in cui CodePipeline sono disponibili. Vedi riferimento AWS CloudFormation StackSets all'azione e riferimento all'azione AWS Step Functions .	27 marzo 2024

Aggiornamenti alla politica gestita	La politica AWS gestita `AWSCodePipeline_FullAccess` è stata aggiornata. Vedi le politiche AWS gestite per AWS CodePipeline.	15 marzo 2024
Support per il timeout configurabile per le azioni di approvazione manuali	Informazioni sulle quote aggiunte per il nuovo campo di timeout configurabile per le azioni di approvazione manuali. Per ulteriori informazioni, consulta Quote .	15 febbraio 2024
Support per il filtraggio dei trigger per rami e percorsi di file	Support aggiunto per la configurazione dei trigger che consente di filtrare lo stato delle pull request, i rami e i percorsi dei file per le pipeline di tipo V2. Per ulteriori informazioni, consulta Filtraggio dei trigger sulle richieste push o pull del codice, Trigger e Filtraggio sui rami delle funzionalità per avviare la pipeline e Quotas.	8 febbraio 2024

Support per nuove modalità di esecuzione della pipeline	Support aggiunto per le modalità di esecuzione della pipeline PARALLEL e QUEUED. Per ulteriori informazioni, vedere Impostazione della modalità di esecuzione della pipeline, Modalità di elaborazione delle esecuzioni in modalità QUEUED, Modalità di elaborazione delle esecuzioni in modalitàPARALLEL e Quotas.	8 febbraio 2024
Aggiornamenti alle pagine della console per la visualizzazione dei dettagli delle azioni, la revisione delle azioni di approvazione manuali e la pagina delle pipeline di elenco	Aggiornamenti della console documentati per il nuovo pulsante e la finestra di dialogo Visualizza dettagli, la nuova finestra di dialogo di approvazione manuale e le nuove colonne per le esecuzioni recenti nella pagina delle pipeline dell'elenco. Per ulteriori informazioni, consulta Visualizzazione delle pipeline (console), Visualizzazione dei dettagli delle azioni in una pipeline e Gestione delle azioni di approvazione nelle pipeline.	10 gennaio 2024

Support per la GitLab gestione automatica	Support aggiunto per la configurazione delle connessioni per le AWS risorse con cui interagire con la gestione GitLab automatica. Per ulteriori informazioni, consulta Connessioni per GitLab la gestione automatica.	28 dicembre 2023
Aggiornamenti alle azioni `CloudFormationStackSet` e `CloudFormationStackInstances`	Il `ConcurrencyMode` parametro è stato aggiunto per l'`CloudFormationStackInstances` azione `CloudFormationStackSet` and. Vedi la pagina di riferimento dell'azione.	19 dicembre 2023
Aggiornamenti ai parametri di AWS Device Farm azione in CodePipeline	I parametri per l' AWS Device Farm azione in CodePipeline sono stati aggiornati. Per ulteriori informazioni, consulta il riferimento AWS Device Farm all'azione.	18 dicembre 2023
Support aggiunto per messaggi di errore dettagliati relativi all' AWS CloudFormation azione in CodePipeline	AWS CloudFormation i messaggi di errore relativi all'azione ora possono mostrare dettagli sulle risorse che hanno avuto esito negativo. Per ulteriori informazioni, consulta il riferimento AWS CloudFormation all'azione.	15 dicembre 2023

Aggiornamenti per l'avvio di una pipeline con modifiche alla revisione del codice sorgente CodePipeline	È ora possibile avviare una pipeline con una revisione dell'origine specificata. Per ulteriori informazioni, consulta Avviare una pipeline con una modifica della revisione di origine.	17 novembre 2023
Nuove regioni supportate	CodePipeline è ora disponibile nelle regioni Asia Pacifico (Hyderabad), Asia Pacifico (Giacarta), Asia Pacifico (Melbourne), Asia Pacifico (Osaka), Medio Oriente (Emirati Arabi Uniti), Europa (Spagna) e Israele (Tel Aviv). L'argomento di riferimento del bucket segnaposto Events e l'argomento sugli endpoint sono stati aggiornati.Servizio AWS	13 novembre 2023
Aggiornamenti per i campi degli eventi in Amazon EventBridge	Ora puoi visualizzare i campi degli eventi aggiornati in Amazon EventBridge. Per ulteriori informazioni, consulta Monitoraggio CodePipeline degli eventi.	9 novembre 2023

Aggiornamenti per nuove pipeline di tipo V2, trigger sui tag Git e variabili di pipeline in CodePipeline	Ora puoi scegliere un tipo di pipeline in. CodePipeline Per una pipeline di tipo V2, ora puoi utilizzare una configurazione di trigger per avviare la pipeline sui tag Git. Con le pipeline di tipo V2, puoi anche usare variabili a livello di pipeline per passare i parametri di input per l'esecuzione di una pipeline. Per maggiori informazioni, consulta Variabili, Tutorial: Usa le variabili a livello di pipeline e Tutorial: usa i tag Git per avviare la pipeline. Per ulteriori informazioni sui tipi di pipeline, consulta Tipi di pipeline.	24 ottobre 2023
CodePipeline consente di riprovare tutte le azioni in una fase fallita	In caso di esito negativo di una fase CodePipeline, è possibile riprovare la fase senza eseguire nuovamente la pipeline. Puoi farlo riprovando le azioni fallite in una fase o riprovando tutte le azioni nella fase a partire dalla prima azione nella fase. Per ulteriori informazioni, consulta Riprovare una fase non riuscita o Azioni non riuscite in una fase.	17 ottobre 2023

Support per GitLab gruppi	Support aggiunto per la configurazione delle connessioni affinché AWS le risorse interagiscano con GitLab i gruppi. Per ulteriori informazioni, consulta GitLab connessioni.	15 settembre 2023
CodePipeline supporta le connessioni a GitLab .com	È possibile utilizzare le connessioni per configurare AWS le risorse per interagire con GitLab .com. Puoi anche scegliere l'opzione di clonazione completa per utilizzare i comandi Git e i metadati per le azioni a valle. Per ulteriori informazioni, consultate le GitLab connessioni e l'argomento di riferimento sulla struttura delle CodeStarSourceConnection azioni.	10 agosto 2023
Aggiornamento dell'`CloudFormationStackInstances` azione	Il `RegionConcurrencyType` parametro è stato aggiunto per l'`CloudFormationStackInstances` azione. Vedi la pagina di riferimento dell'azione per l'`CloudFormationStackInstances` azione.	8 agosto 2023

Aggiorna l'`CloudFormationStackSet` azione	Il `RegionConcurrencyType` parametro è stato aggiunto per l'`CloudFormationStackSet` azione. Vedi la pagina di riferimento dell'azione per l'`CloudFormationStackSet` azione.	24 luglio 2023
Aggiornamenti alla politica gestita	La politica AWS gestita `AWSCodePipeline_FullAccess` è stata aggiornata. Vedi le politiche AWS gestite per AWS CodePipeline.	21 giugno 2023
Aggiornamenti alle procedure di migrazione per le pipeline di sondaggi	Le procedure per migrare (aggiornare) le pipeline di polling per utilizzare il rilevamento delle modifiche basato sugli eventi sono state aggiornate con i passaggi per le pipeline che utilizzano un bucket Amazon S3 abilitato per le notifiche. EventBridge Per ulteriori informazioni, consulta Migrare le pipeline di polling per utilizzare il rilevamento delle modifiche basato sugli eventi.	12 giugno 2023

Aggiornamenti alle politiche gestite	Le politiche AWS `AWSCodePipeline_FullAccess` gestite `AWSCodePipeline_ReadOnlyAccess` sono state aggiornate con un'autorizzazione aggiuntiva. Per ulteriori informazioni, consulta AWS CodePipeline gli aggiornamenti delle politiche AWS gestite.	16 maggio 2023
Aggiornamenti alle politiche gestite	Le politiche AWS `AWSCodePipelineFullAccess` gestite `AWSCodePipelineReadOnlyAccess` sono obsolete. Utilizza le politiche e. `AWSCodePipeline_FullAccess` `AWSCodePipeline_ReadOnlyAccess` Visualizza AWS CodePipeline gli aggiornamenti alle politiche AWS gestite.	17 novembre 2022

Aggiornamenti alle procedure che utilizzano CloudTrail	Tutte le procedure della console, i comandi CLI di esempio e gli AWS CloudFormation snippet e i modelli di esempio per una pipeline con un'origine S3 sono stati aggiornati con l'opzione di scegliere Write e selezionare false per gli eventi di gestione in. CloudTrail Guarda gli esempi aggiornati in Avvio di una pipeline, Tutorial: Create a pipeline with AWS CloudFormation, Edit pipelinesto use push events e Update polling pipelines.	27 aprile 2022
Nuova integrazione supportata con Snyk	Puoi utilizzare l'azione di invoca di Snyk CodePipeline per automatizzare la scansione di sicurezza del tuo codice open source. Per ulteriori informazioni, consulta il riferimento alle azioni e alle integrazioni di Snyk.	10 giugno 2021
Nuova regione Europa supportata (Milano)	CodePipeline è ora disponibile in Europa (Milano). L'argomento Limiti e l'argomento Servizio AWS sugli endpoint sono stati aggiornati.	27 gennaio 2021

Il rilevamento delle modifiche può essere disattivato per le azioni di origine con connessioni	Puoi utilizzare la CLI o l'SDK per aggiornare un'azione di `CodeStarSourceConnection` origine per disattivare il rilevamento automatico delle modifiche per il repository di origine. L'argomento di riferimento sulla struttura delle CodeStarSourceConnection azioni è stato aggiornato con una descrizione del parametro. `DetectChanges`	8 gennaio 2021
CodePipeline ora supporta le azioni AWS CloudFormation StackSets di distribuzione	Un nuovo tutorial, Tutorial: Create a pipeline da utilizzare AWS CloudFormation StackSets AWS CloudFormation StackSets come provider di distribuzione, fornisce i passaggi da seguire per creare e aggiornare i set di stack e le istanze di stack con la pipeline. È stato aggiunto anche l'argomento di riferimento sulla struttura delle AWS CloudFormation StackSets azioni.	30 dicembre 2020
Nuova regione supportata Asia Pacifico (Hong Kong)	CodePipeline è ora disponibile in Asia Pacifico (Hong Kong). L'argomento Limiti e l'argomento Servizio AWS sugli endpoint sono stati aggiornati.	22 dicembre 2020

Visualizza i modelli di EventBridge eventi aggiornati in CodePipeline	I modelli e gli stati aggiornati per gli eventi a livello di pipeline, stage e action sono stati aggiunti agli eventi di monitoraggio CodePipeline .	21 dicembre 2020
Visualizza le esecuzioni delle pipeline in entrata in CodePipeline	È possibile utilizzare la console o la CLI per visualizzare le esecuzioni in entrata. Per ulteriori informazioni, consulta Visualizzare un'esecuzione in entrata (console) e Visualizzare lo stato dell'esecuzione in entrata (CLI).	16 Novembre 2020
L'azione CodeCommit source in CodePipeline supporta l'opzione di clonazione completa	Quando usi un'azione CodeCommit sorgente, puoi scegliere l'opzione full clone per usare i comandi Git e i metadati per le azioni CodeBuild downstream. Per maggiori informazioni, consulta il riferimento all'CodeCommit azione e il tutorial: Usa full clone con una sorgente di pipeline. CodeCommit	11 novembre 2020

CodePipeline supporta connessioni a GitHub e Enterprise Server GitHub	È possibile utilizzare le connessioni per configura re AWS le risorse con cui interagire GitHub, GitHub Enterprise Cloud ed GitHub Enterprise Server. Puoi anche scegliere l'opzione di clonazion e completa per utilizzare i comandi Git e i metadati per le azioni a valle. Per ulteriori informazioni, consultate GitHub connessioni, connessio ni GitHub Enterprise Server e Tutorial: Use full clone with a pipeline source. GitHub Se disponi di una pipeline esistente con un'azione di GitHub origine, consulta Aggiornare un'azione di origine della GitHub versione 1 a un'azione di origine della GitHub versione 2.	30 settembre 2020
L' CodeBuild azione supporta l'abilitazione delle compilazioni in batch AWS CodePipeline	Per CodeBuild le azioni nella pipeline, puoi abilitare le build in batch per eseguire più build in un'unica esecuzione. Per ulteriori informazioni, consulta CodeBuild Action Structure Reference e Create a pipeline (console).	30 luglio 2020

AWS CodePipeline ora supporta le AWS AppConfig azioni di distribuzione	Un nuovo tutorial, Tutorial: Create a pipeline da utilizzare AWS AppConfig come provider di distribuzione, fornisce i passaggi da utilizzare AWS AppConfig per distribuire i file di configurazione con la pipeline. È stato aggiunto anche l'argomento di riferimento sulla struttura delle AWS AppConfig azioni.	25 giugno 2020
AWS CodePipeline ora supporta Amazon VPC negli Stati Uniti AWS GovCloud occidentali	Ora puoi connetterti direttamente AWS CodePipeline tramite un endpoint Amazon VPC privato in AWS GovCloud (Stati Uniti occidentali). Per ulteriori informazioni, consulta Utilizzo CodePipeline con Amazon Virtual Private Cloud.	2 giugno 2020
AWS CodePipeline ora supporta le azioni di AWS Step Functions richiamo	Ora puoi creare una pipeline CodePipeline che utilizzi AWS Step Functions come provider di azioni di invoke. Un nuovo tutorial, Tutorial: Use an AWS Step Functions invoke action in a pipeline, fornisce i passaggi per avviare l'esecuzione di una macchina a stati dalla pipeline. È stato aggiunto anche l'argomento Riferimento per la struttura dell'operazione AWS Step Functions.	28 maggio 2020

Visualizza, elenca e aggiorna le connessioni	Puoi elencare, eliminare e aggiornare le connessioni nella console. Vedi Elenca connessioni in CodePipeline.	21 maggio 2020
Le connessioni supportano l'etichettatura delle risorse di connessione nella CLI	Le risorse di connessione ora supportano il tagging nella AWS CLI. Le connessioni ora si integrano con. AWS CodeGuru Consulta la documentazione di riferimento delle autorizzazioni IAM per le connessioni.	6 maggio 2020
CodePipeline è ora disponibile in AWS GovCloud (Stati Uniti occidentali)	Ora puoi usarlo CodePipeline in AWS GovCloud (Stati Uniti occidentali). Per ulteriori informazioni, consulta Quote .	8 aprile 2020
L'argomento sulle quote mostra quali quote di CodePipeline servizio sono configurabili	L'argomento sulle CodePipeline quote è stato riformattato. La documentazione mostra quali quote del servizio sono configurabili e quali quote non sono configurabili. Vedi Quote in. AWS CodePipeline	12 marzo 2020
Il timeout dell'azione di distribuzione di Amazon ECS è configurabile	Il timeout dell'azione di distribuzione di Amazon ECS è configurabile fino a un'ora (il timeout predefinito). Vedere Limiti in AWS CodePipeline.	5 febbraio 2020

I nuovi argomenti descrivono come interrompere l'esecuzione di una pipeline	È possibile interrompere l'esecuzione di una pipeline in. CodePipeline È possibile specificare che l'esecuzione si interrompe dopo il completamento delle azioni in corso oppure è possibile specificare di interrompere immediatamente l'esecuzione e abbandonare le azioni in corso. Vedi Come vengono interrotte le esecuzioni di una pipeline e Interrompi l'esecuzione di una pipeline in. CodePipeline	21 gennaio 2020
CodePipeline supporta le connessioni	È possibile utilizzare le connessioni per configurare AWS le risorse in modo che interagiscano con archivi di codice esterni. Ogni connessione è una risorsa che può essere utilizzata da servizi come la connessione CodePipeline a un repository di terze parti, come Bitbucket Cloud. Per ulteriori informazioni, consulta Lavorare con le connessioni in. CodePipeline	18 dicembre 2019

Argomenti aggiornati sulla sicurezza, l'autenticazione e il controllo degli accessi	Le informazioni sulla sicurezza , l'autenticazione e il controllo degli accessi per sono CodePipeline state organizza te in un nuovo capitolo sulla sicurezza. Per ulteriori informazioni, consulta Sicurezza.	17 dicembre 2019
Nuovi argomenti descrivon o come utilizzare le variabili nelle pipeline	È ora possibile configurare gli spazi dei nomi per le azioni e generare variabili ogni volta che l'esecuzione dell'azio ne è completata. È possibile impostare azioni downstream per fare riferimento a questi spazi dei nomi e variabili. Consulta Utilizzo delle variabili e Variabili.	14 novembre 2019

I nuovi argomenti descrivono come funzionano le esecuzioni della pipeline, perché le fasi vengono bloccate durante un'esecuzione e quando le esecuzioni della pipeline vengono sostituite	Sono stati aggiunti numerosi argomenti alla sezione iniziale per descrivere il funzionamento delle esecuzioni della pipeline, tra cui il motivo per cui le fasi sono bloccate durante un'esecuzione e ciò che accade quando le esecuzioni della pipeline vengono sostituite. Questi argomenti includono un elenco di concetti, un esempio di DevOps flusso di lavoro e consigli su come strutturare una pipeline. Sono stati aggiunti i seguenti argomenti: termini della pipeline, esempio di DevOps pipeline e Come funzionano le esecuzioni delle pipeline.	11 novembre 2019
CodePipeline supporta le regole di notifica	È ora possibile utilizzare le regole di notifica per notificare agli utenti modifiche importanti nelle pipeline. Per ulteriori informazioni, consulta Creazione di una regola di notifica.	5 novembre 2019

CodeBuild variabili di ambiente disponibili in CodePipeline	Puoi impostare le variabili di CodeBuild ambiente nell'azione di CodeBuild compilazione della tua pipeline. Puoi utilizzare la console o l'interfaccia a riga di comando per aggiungere il parametro `EnvironmentVariables` alla struttura della pipeline. L'argomento Creazione di una pipeline (console) è stato aggiornato. Anche gli esempi di configurazione delle azioni nel riferimento all'azione per CodeBuild sono stati aggiornati.	14 ottobre 2019
Nuova regione	CodePipeline è ora disponibile in Europa (Stoccolma). L'argomento Limiti e l'argomento Servizio AWS sugli endpoint sono stati aggiornati.	5 settembre 2019
Specificare gli ACL predefiniti e il controllo della cache per le azioni di distribuzione di Amazon S3	Ora puoi specificare opzioni predefinite ACL e di controllo della cache quando crei un'azione di distribuzione di Amazon S3 in. CodePipeline I seguenti argomenti sono stati aggiornati: Creazione di una pipeline (console), riferimento alla struttura della CodePipeline pipeline e tutorial: creazione di una pipeline che utilizza Amazon S3 come provider di distribuzione.	27 giugno 2019

Ora puoi aggiungere tag alle risorse in AWS CodePipeline	Ora puoi utilizzare i tag per tracciare e gestire AWS CodePipeline risorse come pipeline, azioni personalizzate e webhook. Sono stati aggiunti i seguenti nuovi argomenti: Etichettatura delle risorse, Utilizzo dei tag per controllare l'accesso alle CodePipeline risorse, Etichettare una pipeline CodePipeline, Etichettare un'azione personalizzata e Aggiungere tag a CodePipeline un webhook. CodePipeline I seguenti argomenti sono stati aggiornati per mostrare come utilizzare la CLI per etichettare le risorse: Creazione di una pipeline (CLI), Creazione di un'azione personalizzata (CLI)e Creazione di un webhook per una fonte. GitHub	15 maggio 2019

È ora possibile visualizzare la cronologia di esecuzione delle azioni in AWS CodePipeline	È ora possibile visualizzare i dettagli sulle esecuzioni precedenti di tutte le operazioni in una pipeline. Questi dettagli includono gli orari di inizio e fine, la durata, l'ID di esecuzione dell'operazione, lo stato, i dettagli sulla posizione degli artefatti di input e output e i dettagli delle risorse esterne. L'argomento Visualizzazione dei dettagli e della cronologia della pipeline è stato aggiornato per riflettere questo supporto.	20 marzo 2019
AWS CodePipeline ora supporta la pubblicazione di applicazioni su AWS Serverless Application Repository	È ora possibile creare una pipeline in CodePipeline cui pubblicare l'applicazione serverless su. AWS Serverless Application Repository Un nuovo tutorial, Tutorial: Publish applications to the AWS Serverless Application Repository, fornisce i passaggi per creare e configurare una pipeline per fornire continuamente la tua applicazione serverless a. AWS Serverless Application Repository	8 marzo 2019

AWS CodePipeline ora supporta azioni interregionali nella console	Ora puoi gestire le azioni interregionali nella AWS CodePipeline console. L'azione Aggiungi un'azione interregionale è stata aggiornata con i passaggi per aggiungere, modificar e o eliminare un'azione che si trova in una AWS regione diversa dalla pipeline. Gli argomenti di riferimento Create a pipeline, Edit a pipeline e CodePipeline Pipeline structure sono stati aggiornati.	14 febbraio 2019
AWS CodePipeline ora supporta le distribuzioni di Amazon S3	Ora puoi creare una pipeline CodePipeline che utilizza Amazon S3 come provider di azioni di distribuzione. Un nuovo tutorial, Tutorial: crea una pipeline che utilizza Amazon S3 come provider di distribuzione, fornisce i passaggi per distribuire file di esempio nel tuo bucket Amazon S3 con. CodePipel ine Anche l'argomento di riferimento sulla struttura della CodePipeline pipeline è stato aggiornato.	16 gennaio 2019

AWS CodePipeline ora supporta le implementazioni di Alexa Skills Kit	Ora puoi usare l' CodePipeline Alexa Skills Kit per la distribuzione continua di competenze Alexa. Un nuovo tutorial, Tutorial: crea una pipeline che distribuisce una skill Amazon Alexa, contiene i passaggi per creare credenziali che consentono di connettersi AWS CodePipeline al tuo account sviluppatore Alexa Skills Kit e quindi creare una pipeline che distribuisca una skill di esempio. L'argomento di riferimento sulla struttura della CodePipeline pipeline è stato aggiornato.	19 dicembre 2018
AWS CodePipeline ora supporta gli endpoint Amazon VPC con tecnologia AWS PrivateLink	Ora puoi connetterti direttamente AWS CodePipeline tramite un endpoint privato nel tuo VPC, mantenendo tutto il traffico all'interno del tuo VPC e della rete. AWS Per ulteriori informazioni, consulta Utilizzo CodePipeline con Amazon Virtual Private Cloud.	6 dicembre 2018

AWS CodePipeline ora supporta le azioni di origine di Amazon ECR e le azioni di distribuzione ECS CodeDeploy	Ora puoi usare CodePipeline e CodeDeploy con Amazon ECR e Amazon ECS per la distribuzione continua di applicazioni basate su contenitori. Un nuovo tutorial, Create a pipeline with an Amazon ECR source and CodeDeploy ECS-to-deployment, contiene i passaggi per utilizzare la console per creare una pipeline che distribuisca applicazioni container archiviate in un repository di immagini su un cluster Amazon ECS con routing del traffico. CodeDeploy Gli argomenti di riferimento sulla creazione di una pipeline e sulla struttura della pipeline sono stati aggiornati. CodePipeline	27 novembre 2018
AWS CodePipeline ora supporta azioni interregionali in una pipeline	Un nuovo argomento, Aggiungi un'azione interregionale, contiene i passaggi per utilizzare AWS CLI o AWS CloudFormation aggiungere un'azione che si trova in un'area diversa dalla pipeline. Gli argomenti di riferimento Create a pipeline, Edit a pipeline e CodePipeline Pipeline structure sono stati aggiornati.	12 novembre 2018

AWS CodePipeline ora si integra con Service Catalog	Ora puoi aggiungere Service Catalog come azione di distribuzione alla tua pipeline. Ciò consente di configurare una pipeline per pubblicare gli aggiornamenti dei prodotti su Service Catalog quando si apporta una modifica al repository di origine. L'argomento Integrazioni è stato aggiornato per riflettere questo supporto per Service Catalog. Due tutorial di Service Catalog sono stati aggiunti alla sezione AWS CodePipeline tutorial.	16 ottobre 2018
AWS CodePipeline ora si integra con AWS Device Farm	Ora puoi aggiungerla AWS Device Farm come azione di test alla tua pipeline. In questo modo puoi impostare una pipeline per testare le applicazioni per dispositivi mobili. L'argomento Integrazioni è stato aggiornato per riflettere questo supporto per. AWS Device Farm Sono stati aggiunti due tutorial AWS Device Farm alla sezione Tutorial di AWS CodePipeline.	19 luglio 2018

AWS CodePipeline Le notifiche di aggiornamento della Guida per l'utente sono ora disponibili tramite RSS	La versione HTML della Guida per l' CodePipeline utente ora supporta un feed RSS di aggiornamenti documenta ti nella pagina Cronologia degli aggiornamenti della documentazione. Il feed RSS include gli aggiornamenti effettuati dopo il 30 giugno 2018. Gli aggiornamenti annunciati in precedenza sono tuttora disponibili nella pagina della cronologia di aggiornam ento della documentazione. Utilizza il pulsante RSS nel pannello del menu in alto per registrarti al feed.	30 giugno 2018

Aggiornamenti precedenti

La tabella seguente descrive le modifiche importanti apportate in ogni versione della Guida per l' CodePipeline utente del 30 giugno 2018 e precedenti.

Modifica	Descrizione	Data della modifica
Utilizza i webhook per rilevare le modifiche all'origi ne nelle pipeline GitHub	Quando crei o modifichi una pipeline nella console, CodePipeline ora crea un webhook che rileva le modifiche al tuo repository di GitHub origine e quindi avvia la pipeline. Per informazioni sulla migrazione della pipeline, consulta Configurare le pipeline per utilizzare i webhook per il GitHub rilevamento delle modifiche. Per ulteriori informazi oni, consulta Avviare l'esecuzione di una pipeline in. CodePipeline	1 maggio 2018

Modifica	Descrizione	Data della modifica
Argomenti aggiornati	Quando crei o modifichi una pipeline nella console, CodePipeline ora crea una regola Amazon CloudWatch Events e un AWS CloudTrail trail che rileva le modifiche al tuo bucket di origine Amazon S3 e quindi avvia la pipeline. Per informazioni sulla migrazione della pipeline, consulta Azioni all'origine e metodi di rilevamento delle modifiche. Tutorial: creazione di una semplice pipeline (bucket S3)È stato aggiornato per mostrare come vengono create la regola e il percorso di Amazon CloudWatch Events quando si seleziona una fonte Amazon S3. Creare una pipeline in CodePipelinee Modificare una tubazione in CodePipeline sono stati inoltre aggiornati. Per ulteriori informazioni, consulta Avvia una pipeline in CodePipeline.	22 marzo 2018
Argomento aggiornato	CodePipeline è ora disponibile in Europa (Parigi). L'argomento Quote in AWS CodePipeline è stato aggiornato.	21 febbraio 2018
Argomenti aggiornati	Ora puoi utilizzare CodePipeline Amazon ECS per la distribuzione continua di applicazioni basate su contenitori. Quando crei una pipeline, puoi selezionare Amazon ECS come fornitore di distribuzione. Una modifica al codice nel tuo repository di controllo del codice sorgente attiva la pipeline per creare una nuova immagine Docker, inviarla al registro dei contenitori e quindi distribuire l'immagine aggiornata su un servizio Amazon ECS. Gli argomenti Integrazioni di prodotti e servizi con CodePipelineCreare una pipeline in CodePipeline, e CodePipeline riferimento alla struttura della tubazione sono stati aggiornati per riflettere questo supporto per Amazon ECS.	12 dicembre 2017

Modifica	Descrizione	Data della modifica
Argomenti aggiornati	Quando crei o modifichi una pipeline nella console, CodePipeline ora crea una regola Amazon CloudWatch Events che rileva le modifiche al tuo CodeCommit repository e quindi avvia automaticamente la pipeline. Per informazioni sulla migrazione della pipeline esistente, consulta Azioni all'origine e metodi di rilevamento delle modifiche. Tutorial: crea una pipeline semplice (CodeCommitrepository)È stato aggiornato per mostrare come vengono creati la regola e il ruolo di Amazon CloudWatch Events quando selezioni un CodeCommit repository e un branch. Creare una pipeline in CodePipelinee Modificare una tubazione in CodePipeline sono stati inoltre aggiornati. Per ulteriori informazioni, consulta Avvia una pipeline in CodePipeline.	11 ottobre 2017
Argomenti nuovi e aggiornati	CodePipeline ora fornisce supporto integrato per le notifiche di modifica dello stato della pipeline tramite Amazon CloudWatch Events e Amazon Simple Notification Service (Amazon SNS). È stato aggiunto un nuovo tutorial Tutorial: imposta una regola CloudWatch Events per ricevere notifiche e-mail per le modifiche allo stato della pipeline. Per ulteriori informazioni, consulta Monitoraggio CodePipeline degli eventi.	8 settembre 2017

Modifica	Descrizione	Data della modifica
Argomenti nuovi e aggiornati	Ora puoi aggiungere CodePipeline come destinazione per le azioni di Amazon CloudWatch Events. Le regole di Amazon CloudWatch Events possono essere configurate per rilevare le modifiche all'origine in modo che la pipeline si avvii non appena si verificano tali modifiche, oppure possono essere impostate per eseguire esecuzioni di pipeline pianificate. Sono state aggiunte informazioni per l'opzione di configurazione dell'azione PollForSourceChanges sorgente. Per ulteriori informazioni, consulta Avvia una pipeline in CodePipeline.	5 settembre 2017
Nuove regioni	CodePipeline è ora disponibile in Asia Pacifico (Seoul) e Asia Pacifico (Mumbai). L'argomento Quote in AWS CodePipeline e l'argomento Regioni ed endpoint sono stati aggiornati.	27 luglio 2017
Nuove regioni	CodePipeline è ora disponibile negli Stati Uniti occidentali (California settentrionale), Canada (Centrale) ed Europa (Londra). L'argomento Quote in AWS CodePipeline e l'argomento Regioni ed endpoint sono stati aggiornati.	29 giugno 2017
Argomenti aggiornati	Puoi ora visualizzare i dettagli relativi alle esecuzioni precedenti di una pipeline, non solo l'esecuzione più recente. Questi dettagli includono l'ora di inizio e di fine, la durata e l'ID di esecuzione. I dettagli sono disponibili per un massimo di 100 esecuzioni della pipeline durante il periodo di 12 mesi più recente. Gli argomenti Visualizza le pipeline e i dettagli in CodePipeline, CodePipeline riferimento alle autorizzazioni e Quote in AWS CodePipeline sono stati aggiornati per riflettere questo supporto.	22 giugno 2017
Argomento aggiornato	Nouvola è stato aggiunto all'elenco di operazioni disponibili in Integrazioni di operazioni di test.	18 maggio 2017

Modifica	Descrizione	Data della modifica
Argomenti aggiornati	Nella AWS CodePipeline procedura guidata, la pagina Step 4: Beta è stata rinominata Step 4: Deploy. Il nome predefinito della fase creata da questo passaggio è stato modificato da "Beta" a "Gestione temporanea". Numerosi argomenti e screenshot sono stati aggiornati per riflettere queste modifiche.	7 Aprile 2017
Argomenti aggiornati	Ora puoi aggiungerla AWS CodeBuild come azione di test a qualsiasi fase di una pipeline. Ciò consente di AWS CodeBuild eseguire più facilmente test unitari sul codice. Prima di questa versione, era possibile AWS CodeBuild eseguire gli unit test solo come parte di un'azione di compilazione. Un'operazione di compilazione richiede un artefatto di output di compilazione, che in genere gli unit test non generano. Gli argomenti Integrazioni di prodotti e servizi con CodePipelineModificare una tubazione in CodePipeline, e CodePipeline riferimento alla struttura della tubazione sono stati aggiornati per riflettere questo supporto per AWS CodeBuild.	8 marzo 2017

Modifica	Descrizione	Data della modifica
Argomenti nuovi e aggiornati	Il sommario è stato riorganizzato per includere sezioni per pipeline, operazioni e transizioni di fase. È stata aggiunta una nuova sezione per i CodePipeline tutorial. Per una migliore usabilità, Integrazioni di prodotti e servizi con CodePipeline è stato suddiviso in argomenti più brevi. Una nuova sezione, Authorization and Access Control, fornisce informazioni complete sull'utilizzo di AWS Identity and Access Management (IAM) e CodePipeline su come proteggere l'accesso alle risorse tramite l'uso di credenzia li. Queste credenziali forniscono le autorizzazioni necessari e per accedere alle AWS risorse, ad esempio inserire e recuperare artefatti dai bucket Amazon S3 e integrare gli stack nelle pipeline. AWS OpsWorks	8 febbraio 2017
Nuova regione	CodePipeline è ora disponibile in Asia Pacifico (Tokyo). L'argomento Quote in AWS CodePipeline e l'argomento Regioni ed endpoint sono stati aggiornati.	14 dicembre 2016
Nuova regione	CodePipeline è ora disponibile in Sud America (San Paolo). L'argomento Quote in AWS CodePipeline e l'argomento Regioni ed endpoint sono stati aggiornati.	7 dicembre 2016

Modifica	Descrizione	Data della modifica
Argomenti aggiornati	Ora puoi aggiungerlo AWS CodeBuild come azione di compilazione a qualsiasi fase di una pipeline. AWS CodeBuild è un servizio di compilazione completamente gestito nel cloud che compila il codice sorgente, esegue test unitari e produce artefatti pronti per l'implementazione. Puoi utilizzare un progetto di build esistente o crearne uno nella console. CodePipeline L'output del progetto di compilazione può quindi essere distribuito come parte di una pipeline. Gli argomenti Integrazioni di prodotti e servizi con CodePipelineCreare una pipeline in CodePipeline, Autenticazione e controllo degli accessi, sono CodePipeline riferimento alla struttura della tubazione stati aggiornati per riflettere questo supporto per AWS CodeBuild. Ora puoi utilizzare CodePipeline con AWS CloudFormation e il AWS Serverless Application Model per fornire continuamente le tue applicazioni serverless. L'argomento Integrazioni di prodotti e servizi con CodePipeline è stato aggiornato in modo da riflettere questo supporto. Integrazioni di prodotti e servizi con CodePipelineè stato riorganizzato in modo da raggruppare le offerte dei partner AWS e dei partner suddivise per tipo di azione.	1° dicembre 2016
Nuova regione	CodePipeline è ora disponibile in Europa (Francoforte). L'argomento Quote in AWS CodePipeline e l'argomento Regioni ed endpoint sono stati aggiornati.	16 novembre 2016

Modifica	Descrizione	Data della modifica
Argomenti aggiornati	AWS CloudFormation ora può essere selezionato come fornitore di distribuzione nelle pipeline, consentendoti di intervenire sugli AWS CloudFormation stack e sui set di modifiche come parte dell'esecuzione di una pipeline. Gli argomenti Integrazioni di prodotti e servizi con CodePipelineCreare una pipeline in CodePipeline, Autenticazione e controllo degli accessi, CodePipeline riferimento alla struttura della tubazione sono stati aggiornati per riflettere questo supporto per. AWS CloudFormation	3 Novembre 2016
Nuova regione	CodePipeline è ora disponibile nella regione Asia Pacifico (Sydney). L'argomento Quote in AWS CodePipeline e l'argomento Regioni ed endpoint sono stati aggiornati.	26 ottobre 2016
Nuova regione	CodePipeline è ora disponibile in Asia Pacifico (Singapore). L'argomento Quote in AWS CodePipeline e l'argomento Regioni ed endpoint sono stati aggiornati.	20 ottobre 2016
Nuova regione	CodePipeline è ora disponibile nella regione Stati Uniti orientali (Ohio). L'argomento Quote in AWS CodePipeline e l'argomento Regioni ed endpoint sono stati aggiornati.	17 ottobre 2016
Argomento aggiornato	Creare una pipeline in CodePipeline è stato aggiornato in modo da riflettere il supporto per la visualizzazione di identificatori di versione di operazioni personalizzate negli elenchi Source provider (Provider di origine) e Build provider (Provider di compilazione).	22 settembre 2016
Argomento aggiornato	La sezione Gestisci le azioni di approvazione in CodePipeline è stata aggiornata in modo da riflettere un miglioramento che consente ai revisori dell'operazione di approvazione di aprire il modulo Approve or reject the revision (Approva o rifiuta la revisione) direttamente da una notifica e-mail.	14 settembre 2016

Modifica	Descrizione	Data della modifica
Argomenti nuovi e aggiornati	Un nuovo argomento che descrive come visualizzare i dettagli sulle modifiche al codice attualmente in corso nella pipeline di rilascio del software. L'accesso rapido alle informazioni può essere utile durante la revisione delle operazioni di approvazione manuale o la risoluzione di problemi nella pipeline. Una nuova sezione, Monitoraggio di pipeline, fornisce un percorso centralizzato per tutti gli argomenti correlati al monitoraggio dello stato e all'avanzamento delle pipeline.	08 settembre 2016
Argomenti nuovi e aggiornati	Una nuova sezione, Gestisci le azioni di approvazione in CodePipeline, fornisce informazioni sulla configura zione e l'utilizzo di operazioni di approvazione manuale nelle pipeline. Gli argomenti di questa sezione forniscon o informazioni concettuali sul processo di approvazione, istruzioni per configurare le autorizzazioni IAM richieste, creare azioni di approvazione e approvare o rifiutare azioni di approvazione ed esempi di dati JSON generati quando viene raggiunta un'azione di approvazione in una pipeline.	06 luglio 2016
Nuova regione	CodePipeline è ora disponibile nella regione Europa (Irlanda). Gli argomenti Quote in AWS CodePipeline e Regioni ed endpoint sono stati aggiornati.	23 giugno 2016
Nuovo argomento	Un nuovo argomento, Riprova un'azione fallita in una fase, è stato aggiunto per descrivere la modalità di ripetizione di un'operazione non riuscita o di un gruppo di operazioni non riuscite parallele nella fase.	22 giugno 2016

Modifica	Descrizione	Data della modifica
Argomenti aggiornati	Alcuni argomentiCreare una pipeline in CodePipeline, tra cui Autenticazione e controllo degli accessiCodePipeline riferimento alla struttura della tubazione, sono stati aggiornati per riflettere il supporto alla configurazione di una pipeline per la distribuzione del codice insieme ai ricettari e alle applicazioni Chef personalizzati creati in. Integrazioni di prodotti e servizi con CodePipeline AWS OpsWorks CodePipeline il supporto per AWS OpsWorks è attualmente disponibile solo nella regione Stati Uniti orientali (Virginia settentrionale) (us-east-1).	2 giugno 2016
Argomenti nuovi e aggiornati	È stato aggiunto un nuovo argomento Tutorial: crea una pipeline semplice (CodeCommitrepository). Questo argomento fornisce una procedura dettagliata di esempio che mostra come utilizzare un CodeCommit repository e un branch come posizione di origine per un'azione di origine in una pipeline. Diversi altri argomenti sono stati aggiornati per riflettere questa integrazione con CodeCommit, tra cui Autenticazione e controllo degli accessi,, Integrazioni di prodotti e servizi con CodePipeline e. Tutorial: creazione di una pipeline a quattro fasi Risoluzione dei problemi CodePipeline	18 Aprile 2016
Nuovo argomento	È stato aggiunto un nuovo argomento Invoca una AWS Lambda funzione in una pipeline in CodePipeline. Questo argomento contiene AWS Lambda funzioni e passaggi di esempio per aggiungere funzioni Lambda alle pipeline.	27 gennaio 2016
Argomento aggiornato	Una nuova sezione è stata aggiunta a Autenticazione e controllo accessi, Policy basate sulle risorse.	22 gennaio 2016

Modifica	Descrizione	Data della modifica
Nuovo argomento	È stato aggiunto un nuovo argomento Integrazioni di prodotti e servizi con CodePipeline. Le informazioni sulle integrazioni con partner e altro sono Servizi AWS state spostate in questo argomento. Sono anche stati aggiunti collegamenti a blog e video.	17 dicembre 2015
Argomento aggiornato	Dettagli di integrazione con Solano CI sono stati aggiunti in Integrazioni di prodotti e servizi con CodePipeline.	17 Novembre 2015
Argomento aggiornato	Il CodePipeline Plugin for Jenkins è ora disponibile tramite Jenkins Plugin Manager come parte della libreria di plugin per Jenkins. Le fasi di installazione del plugin sono state aggiornate in Tutorial: creazione di una pipeline a quattro fasi.	9 Novembre 2015
Nuova regione	CodePipeline è ora disponibile nella regione Stati Uniti occidentali (Oregon). L'argomento Quote in AWS CodePipeline è stato aggiornato. Sono stati aggiunti collegamenti a Regioni ed endpoint.	22 ottobre 2015
Nuovo argomento	Sono stati aggiunti due nuovi argomenti, Configura la crittografia lato server per gli artefatti archiviati in Amazon S3 per CodePipeline e Crea una pipeline CodePipeline che utilizzi le risorse di un altro account AWS. Una nuova sezione è stata aggiunta a Autenticazione e controllo accessi, Esempio 8: utilizzo delle risorse AWS associate a un altro account in una pipeline.	25 agosto 2015
Argomento aggiornato	L'argomento Creare e aggiungere un'azione personalizzata in CodePipeline è stato aggiornato in modo da riflettere le modifiche nella struttura, inclusi `inputArtifactDetails` e `outputArtifactDetails` .	17 agosto 2015

Modifica	Descrizione	Data della modifica
Argomento aggiornato	L'Risoluzione dei problemi CodePipelineargomento è stato aggiornato con passaggi rivisti per la risoluzione dei problemi con il ruolo di servizio e Elastic Beanstalk.	11 agosto 2015
Argomento aggiornato	L'argomento Autenticazione e controllo degli accessi è stato aggiornato con le ultime modifiche al ruolo di servizio per. CodePipeline	6 agosto 2015
Nuovo argomento	È stato aggiunto un argomento Risoluzione dei problemi CodePipeline. Sono stati aggiunti passaggi aggiornati per i ruoli IAM e Jenkins inTutorial: creazione di una pipeline a quattro fasi.	24 luglio 2015
Aggiornamento di argomento	Fasi aggiornate sono state aggiunte per il download di file di esempio in Tutorial: creazione di una semplice pipeline (bucket S3) e Tutorial: creazione di una pipeline a quattro fasi.	22 luglio 2015
Aggiornamento di argomento	Una soluzione alternativa temporanea per problemi di download con i file di esempio è stata aggiunta in Tutorial: creazione di una semplice pipeline (bucket S3).	17 luglio 2015
Aggiornamento di argomento	Un collegamento è stato aggiunto in Quote in AWS CodePipeline per puntare a informazioni relative ai limiti che possono essere modificati.	15 luglio 2015
Aggiornamento di argomento	La sezione relativa alle policy gestite in Autenticazione e controllo accessi è stata aggiornata.	10 luglio 2015
Versione pubblica iniziale	Questa è la versione pubblica iniziale della Guida per l' CodePipeline utente.	9 luglio 2015

AWS Glossario

Per la AWS terminologia più recente, consultate il AWS glossario nella sezione Reference.Glossario AWS

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.