aws

Guida per l'utente

Amazon Managed Workflows for Apache Airflow

Amazon Managed Workflows for Apache Airflow: Guida per l'utente

Table of Contents

Che cos'è Amazon MWAA? 1
Funzionalità 1
Architettura 2
Integrazione 3
Versioni supportate 4
Fasi successive 4
Avvio rapido 5
In questo tutorial 5
Prerequisiti 6
Fase uno: salvare il AWS CloudFormation modello localmente 6
Fase due: creare lo stack utilizzando il AWS CLI 17
Fase tre: caricare un DAG su Amazon S3 ed eseguirlo nell'interfaccia utente di Apache Airflow 17
Fase quattro: Visualizza i log in Logs CloudWatch 18
Fasi successive 18
Nozioni di base 19
Prerequisiti 19
Informazioni sulla guida 19
Prima di iniziare 20
Regioni disponibili 20
Creazione di un bucket 21
Prima di iniziare 21
Creare are are are are are are are 22
Fasi successive 23
Crea la rete VPC 24
Prerequisiti 24
Prima di iniziare 25
Opzioni per creare la rete Amazon VPC 25
Fasi successive 39
Creazione di un ambiente 39
Prima di iniziare 40
Versioni Apache Airflow 40
Creazione di un ambiente 41
Fasi successive 23

Gestione dell'accesso ... 46
Accesso a un ambiente Amazon MWAA ... 46
Come funziona ... 47
Accesso completo alla console ... 48
Accesso completo all'API ... 55
Accesso alla console in sola lettura ... 59
Accesso all'interfaccia utente di Apache Airflow ... 60
Accesso alla CLI di Apache Airflow ... 61
Creazione di una policy JSON ... 61
Esempio di caso d'uso ... 62
Fasi successive ... 64
Ruolo collegato al servizio ... 64
Autorizzazioni di ruolo collegate ai servizi per Amazon MWAA ... 65
Creazione di un ruolo collegato ai servizi per Amazon MWAA ... 68
Modifica di un ruolo collegato al servizio per Amazon MWAA ... 68
Eliminazione di un ruolo collegato al servizio per Amazon MWAA ... 68
Regioni supportate per i ruoli collegati ai servizi Amazon MWAA ... 69
Aggiornamenti alle policy ... 69
Ruolo di esecuzione ... 70
Panoramica del ruolo di esecuzione ... 70
Crea un nuovo ruolo ... 73
Visualizza e aggiorna una politica sul ruolo di esecuzione ... 73
Concedi l'accesso al bucket Amazon S3 con blocco di accesso pubblico a livello di account ... 75
Usa le connessioni Apache Airflow ... 75
Policy di esempio ... 75
Fasi successive ... 81
Prevenzione del confused deputy tra servizi ... 82
Modalità di accesso Apache Airflow ... 83
Modalità di accesso Apache Airflow ... 84
Panoramica delle modalità di accesso ... 86
Configurazione per le modalità di accesso privato e pubblico ... 86
Accesso all'endpoint VPC per il server Web Apache Airflow (accesso alla rete privata) ... 88
Accesso ad Apache Airflow ... 89
Prerequisiti ... 89
Accesso ... 89

AWS CLI ... 90
Apri l'interfaccia utente Airflow ... 90
Accesso ad Apache Airflow ... 90
Creare un token di accesso al server Web ... 90
Prerequisiti ... 91
Utilizzando il AWS CLI ... 92
Utilizzando uno script bash ... 92
Utilizzo di una richiesta API POST ... 93
Usare uno script Python ... 93
Fasi successive ... 94
Token CLI Apache Airflow ... 94
Prerequisiti ... 95
Utilizzo di AWS CLI ... 96
Usare uno script curl ... 96
Usare uno script bash ... 98
Usare uno script Python ... 100
Fasi successive ... 102
Utilizzo dell'API REST di Apache Airflow ... 103
Crea un token di sessione del server Web ... 104
Chiama l'API REST di Apache Airflow ... 105
Riferimento ai comandi CLI di Apache Airflow ... 106
Prerequisiti ... 107
Cosa è cambiato nella v2 ... 108
Comandi CLI supportati ... 108
Codice di esempio ... 111
Gestione delle connessioni ... 114
Panoramica ... 114
Pacchetti Apache Airflow ... 114
Pacchetti provider per connessioni Apache Airflow v2.8.1 ... 115
Pacchetti provider per connessioni Apache Airflow v2.7.2 ... 116
Pacchetti provider per connessioni Apache Airflow v2.6.3 ... 117
Pacchetti provider per connessioni Apache Airflow v2.5.1 ... 118
Pacchetti provider per connessioni Apache Airflow v2.4.3 ... 119
Pacchetti provider per connessioni Apache Airflow v2.2.2 ... 119
Pacchetti provider per connessioni Apache Airflow v2.0.2 ... 120
Specificare pacchetti di provider più recenti ... 120

Tipi di connessione 121
Esempio di stringa URI di connessione 122
Modello di connessione 122
Esempio di utilizzo di un modello di connessione HTTP per una connessione Jdbc 124
Configurazione di Secrets Manager 126
Fase uno: fornire ad Amazon MWAA l'autorizzazione ad accedere alle chiavi segrete di Secrets Manager 127
Fase due: creare il backend Secrets Manager come opzione di configurazione di Apache Airflow 128
Fase tre: generare una stringa URI di connessione Apache Airflow AWS 129
Fase quattro: Aggiungere le variabili in Secrets Manager 132
Fase cinque: aggiungere la connessione in Secrets Manager 133
Codice di esempio 135
Risorse 135
Fasi successive 135
Gestione degli ambienti 136
Configurazione della classe di ambiente 136
Funzionalità ambientali 136
Apache Airflow Scheduler 138
Configurazione della scalabilità automatica dei lavoratori 138
Come funziona la scalabilità dei lavoratori 139
Utilizzo della console Amazon MWAA 140
Esempio di utilizzo ad alte prestazioni 140
Risoluzione dei problemi relativi alle attività bloccate nello stato di esecuzione 142
Fasi successive 142
Configurazione della scalabilità automatica del server Web 142
Come funziona la scalabilità dei server web 143
Utilizzo della console Amazon MWAA 143
Utilizzo delle opzioni di configurazione 144
Prerequisiti 145
Come funziona 145
Utilizzo delle opzioni di configurazione per caricare i plugin in Apache Airflow v2 145
Panoramica delle opzioni di configurazione 146
Informazioni di riferimento sulla configurazione 147
Esempi e codice di esempio 153
Fasi successive 155

Aggiornamento della versione 155
Aggiorna le risorse del tuo flusso di lavoro 155
Specificare la nuova versione 156
Utilizzo di uno script di avvio 157
Configurare uno script di avvio 158
Installa i runtime Linux 162
Impostazione delle variabili di ambiente 163
Lavorare con i DAG 167
Panoramica dei bucket Amazon S3 167
Aggiunta o aggiornamento di DAG 168
Prerequisiti 168
Come funziona 169
Cosa è cambiato nella versione 2 170
Test dei DAG utilizzando l'utilità CLI di Amazon MWAA 170
Caricamento del codice DAG in Amazon S3 170
Specificare il percorso di una cartella DAG 172
Visualizzazione delle modifiche nell'interfaccia utente di Apache Airflow 172
Fasi successive 172
Installazione di plugin personalizzati 173
Prerequisiti 173
Come funziona 174
Cosa è cambiato nella v2 174
Panoramica dei plugin personalizzati 175
Esempi di plugin personalizzati 176
Creazione di un file plugins.zip 185
Caricamento `plugins.zip` su Amazon S3 186
Installazione di plugin personalizzati nel tuo ambiente 187
Esempi di casi d'uso per plugins.zip 188
Fasi successive 189
Installazione delle dipendenze in Python 189
Prerequisiti 190
Come funziona 190
Panoramica delle dipendenze in Python 191
Creazione di un file requirements.txt 191
Caricamento `requirements.txt` su Amazon S3 194
Installazione delle dipendenze Python nel proprio ambiente 196

Visualizzazione dei log per il `requirements.txt` ... 197
Fasi successive ... 198
Eliminazione di file su Amazon S3 ... 198
Prerequisiti ... 198
Panoramica del controllo delle versioni ... 199
Come funziona ... 199
Eliminazione di un DAG su Amazon S3 ... 199
Rimozione di plugins.zip o requirements.txt «current» ... 200
Eliminare plugins.zip o requirements.txt «non correnti» ... 200
Eliminazione di file con cicli di vita ... 201
Esempio di politica del ciclo di vita ... 201
Fasi successive ... 202
Rete ... 203
Informazioni sul networking ... 203
Termini ... 204
Cosa è supportato ... 204
Panoramica dell'infrastruttura VPC ... 204
Esempi di casi d'uso per una modalità di accesso Amazon VPC e Apache Airflow ... 208
Sicurezza nel tuo VPC ... 210
Termini ... 210
Panoramica sulla sicurezza ... 211
Liste di controllo accessi di rete (ACL) ... 211
Gruppi di sicurezza VPC ... 212
Policy degli endpoint VPC (solo routing privato) ... 214
Gestione dell'accesso agli endpoint VPC ... 215
Prezzi ... 216
Panoramica degli endpoint VPC ... 216
Autorizzazione all'uso di altri servizi AWS ... 217
Visualizzazione degli endpoint VPC ... 217
Accesso all'endpoint VPC per il server Web Apache Airflow (accesso alla rete privata) ... 219
Endpoint del servizio VPC in Amazon VPC privati ... 221
Prezzi ... 221
Rete privata e routing privato ... 222
(Obbligatori) Endpoint VPC ... 223
Collegamento degli endpoint VPC richiesti ... 223
(Facoltativo) Abilita gli indirizzi IP privati per l'endpoint dell'interfaccia VPC Amazon S3 ... 227

Gestire i propri endpoint Amazon VPC 228
Creazione di un ambiente in un Amazon VPC condiviso 229
Tutorial 239
Tutorial: AWS Client VPN 239
Rete privata privata privata 240
Casi d'uso 241
Prima di iniziare 241
Obiettivi 241
(Facoltativo) Fase uno: identificazione del VPC, delle regole CIDR e delle protezioni VPC .. 242
Fase 2: Creazione dei certificati server e client e client e caricare i certificati server e client . 243
Fase tre: salvare ilAWS CloudFormation modello localmente 244
Fase quattro: creare loAWS CloudFormation stack Client VPN 246
Fase cinque: associa le sottoreti alla tua Client VPN 246
Fase sei: aggiungi una regola di accesso di autorizzazione alla tua Client VPN 247
Fase 7: Scaricare il file di configurazione dell'endpoint Client VPN VPN Client VPN VPN Client VPN VPN 247
Fase otto: Connect alAWS Client VPN 249
Fasi successive 250
Tutorial: Linux Bastion Host 250
Rete privata 251
Casi d'uso 252
Prima di iniziare 252
Obiettivi 252
Fase uno: creare l'istanza bastion 252
Fase due: creare il tunnel ssh 254
Fase tre: configura il bastion security group come regola in entrata 255
Fase quattro: Copia l'URL di Apache Airflow 256
Fase cinque: configurare le impostazioni del proxy 256
Fase sei: apri l'interfaccia utente di Apache Airflow 259
Fasi successive 259
Tutorial: limitazione degli utenti a un sottoinsieme di DAG 259
Prerequisiti 260
Fase uno: fornisci l'accesso al server Web Amazon MWAA al tuo principale IAM con il ruolo `Public` Apache Airflow predefinito. 260
Fase due: creare un nuovo ruolo personalizzato di Apache Airflow 261
Fase tre: assegna il ruolo che hai creato al tuo utente Amazon MWAA 262

Passaggi successivi 263
Risorse correlate 263
Tutorial: automatizza la gestione degli endpoint del tuo ambiente 263
Prerequisiti 264
Crea Amazon VPC 265
Creazione della funzione Lambda 265
Crea la regola EventBridge 266
Creazione dell'ambiente 267
Esempi di codice 268
Importa variabili DAG 269
Versione 269
Prerequisiti 269
Autorizzazioni 269
Dipendenze 269
Esempio di codice 270
Fasi successive 271
Utilizzo di `SSHOperator` 271
Versione 272
Prerequisiti 272
Autorizzazioni 272
Requisiti 273
Copia la tua chiave segreta su Amazon S3 273
Crea una nuova connessione Apache Airflow 273
Esempio di codice 274
Connessione Apache Airflow Snowflake in Secrets Manager 276
Versione 276
Prerequisiti 276
Autorizzazioni 277
Requisiti 277
Esempio di codice 277
Fasi successive 278
Utilizzo di un DAG per scrivere metriche personalizzate 278
Versione 279
Prerequisiti 279
Autorizzazioni 279
Dipendenze 279

Esempio di codice ... 279
Pulizia del database Aurora PostgreSQL ... 282
Versione ... 283
Prerequisiti ... 283
Dipendenze ... 283
Esempio di codice ... 283
Esportazione di metadati ambientali in Amazon S3 ... 285
Versione ... 286
Prerequisiti ... 286
Autorizzazioni ... 286
Requisiti ... 287
Esempio di codice ... 287
Utilizzo di una variabile Apache Airflow in Secrets Manager ... 289
Versione ... 290
Prerequisiti ... 290
Autorizzazioni ... 290
Requisiti ... 290
Esempio di codice ... 291
Fasi successive ... 292
Utilizzo di una connessione Apache Airflow in Secrets Manager ... 292
Versione ... 292
Prerequisiti ... 292
Autorizzazioni ... 293
Requisiti ... 290
Esempio di codice ... 293
Fasi successive ... 296
Plugin personalizzato con Oracle ... 296
Versione ... 297
Prerequisiti ... 297
Autorizzazioni ... 297
Requisiti ... 297
Esempio di codice ... 298
Crea il plugin personalizzato ... 299
Opzioni di configurazione del flusso d'aria ... 302
Fasi successive ... 302
Plugin personalizzato con variabili di ambiente ... 302

Versione 303
Prerequisiti 303
Autorizzazioni 303
Requisiti 303
Plug-in personalizzato 303
Plugins.zip 304
Opzioni di configurazione del flusso d'aria 304
Fasi successive 305
Modifica del fuso orario di un DAG 305
Versione 305
Prerequisiti 305
Autorizzazioni 306
Crea un plugin per modificare il fuso orario nei log di Airflow 306
Creazione di una destinazione `plugins.zip` 306
Esempio di codice 307
Fasi successive 308
Rinfrescante eAWS CodeArtifacttoken in fase di esecuzione 308
Versione 309
Prerequisiti 309
Autorizzazioni 309
Esempio di codice 310
Fasi successive 311
Plugin personalizzato con Apache Hive e Hadoop 311
Versione 312
Prerequisiti 312
Autorizzazioni 312
Requisiti 290
Scarica le dipendenze 313
Plugin personalizzato 314
Plugins.zip 315
Esempio di codice 315
Opzioni di configurazione del flusso d'aria 315
Fasi successive 316
Plugin personalizzato per la patchPythonVirtualenvOperator 316
Versione 316
Prerequisiti 317

Autorizzazioni ... 317
Requisiti ... 317
Codice di esempio di plugin personalizzato ... 317
Plugins.zip ... 319
Esempio di codice ... 319
Opzioni di configurazione del flusso d'aria ... 322
Fasi successive ... 322
Richiamare i DAG con Lambda ... 322
Versione ... 322
Prerequisiti ... 322
Autorizzazioni ... 323
Dipendenze ... 323
esempio di codice ... 324
Richiamare i DAG in diversi ambienti ... 325
Versione ... 325
Prerequisiti ... 325
Autorizzazioni ... 326
Dipendenze ... 326
Esempio di codice ... 326
Server Amazon RDS ... 328
Versione ... 328
Prerequisiti ... 329
Dipendenze ... 283
Connessione Apache Airflow v2 ... 329
Esempio di codice ... 330
Fasi successive ... 332
Integrazione con Amazon EMR ... 333
Versione ... 333
Esempio di codice ... 333
Amazon EKS (eksctl) ... 336
Versione ... 336
Prerequisiti ... 336
Crea una chiave pubblica per Amazon EC2 ... 337
Crea il cluster ... 337
Creare un`mwaa`namespace ... 338
Crea un ruolo per`mwaa`namespace ... 338

Crea e associa un ruolo IAM per il cluster Amazon EKS 340
Creare il file requirements.txt 343
Crea una mappatura dell'identità per Amazon EKS 343
Creazione del kubeconfig 343
Crea un DAG 344
Aggiungi il DAG ekube_config.yamlal bucket Amazon S3 346
Abilita e attiva l'esempio 347
Utilizzo di ECSOperator 347
Versione 347
Prerequisiti 348
Autorizzazioni 348
Crea un cluster Amazon ECS 349
Esempio di codice 354
Usare dbt con Amazon MWAA 357
Versione 358
Prerequisiti 358
Dipendenze 358
Caricare un progetto dbt su Amazon S3 359
Utilizzate un DAG per verificare l'installazione della dipendenza da dbt 360
Usa un DAG per eseguire un progetto dbt 361
AWS blog e tutorial 361
Best practice 362
Ottimizzazione delle prestazioni per Apache Airflow 362
Aggiungere un'opzione di configurazione Apache Airflow 363
Pianificatore Apache Airflow 363
Cartelle DAG 368
File DAG 370
Attività 375
Gestione delle dipendenze in Python 380
Test dei DAG utilizzando l'utilità CLI Amazon MWAA 381
Installazione delle dipendenze Python utilizzando il formato file dei requisiti PyPi .org 381
Abilitazione dei log sulla console Amazon MWAA 388
Visualizzazione dei log nella console Logs CloudWatch 388
Visualizzazione degli errori nell'interfaccia utente di Apache Airflow 389
Scenari di esempio requirements.txt 390
Monitoraggio e metriche 391

Panoramica ... 391
CloudWatch Panoramica di Amazon ... 392
AWS CloudTrail panoramica ... 392
Visualizzazione dei log di audit ... 392
Creare un percorso in CloudTrail ... 393
Visualizzazione degli eventi con la cronologia CloudTrail degli eventi ... 393
Esempio di percorso per `CreateEnvironment` ... 393
Fasi successive ... 395
Visualizzazione dei registri Airflow ... 395
Prezzi ... 396
Prima di iniziare ... 396
Tipi di log ... 396
Abilitazione dei log di Apache Airflow ... 396
Visualizzazione dei log di Apache Airflow ... 397
Esempi di log dello scheduler ... 397
Fasi successive ... 398
Dashboard e allarmi di monitoraggio ... 398
Metriche ... 399
Panoramica degli stati di allarme ... 399
Esempi di dashboard e allarmi personalizzati ... 400
Eliminazione di metriche e dashboard ... 405
Fasi successive ... 405
Metriche dell'ambiente Apache Airflow v2 ... 405
Termini ... 406
Dimensioni ... 407
Accesso alle metriche nella console CloudWatch ... 408
Le metriche di Apache Airflow sono disponibili in CloudWatch ... 408
Scelta delle metriche da segnalare ... 424
Fasi successive ... 424
Metriche relative a contenitori, code e database ... 424
Termini ... 425
Dimensioni ... 426
Accesso ai parametri ... 426
Elenco delle metriche ... 427
Sicurezza ... 431
Protezione dei dati ... 432

Crittografia ... 433
Utilizzo di chiavi gestite dal cliente ... 435
AWS Identity and Access Management ... 439
Destinatari ... 439
Autenticazione con identità ... 440
Gestione dell'accesso tramite policy ... 443
Consentire agli utenti di visualizzare le loro autorizzazioni ... 446
Risoluzione dei problemi relativi all'identità e all'accesso ad Amazon Managed Workflows per Apache Airflow ... 447
Come funziona Amazon MWAA con IAM ... 448
Convalida della conformità ... 454
Resilienza ... 455
Sicurezza dell'infrastruttura ... 455
Analisi della configurazione e delle vulnerabilità ... 456
Best practice ... 456
Le migliori pratiche di sicurezza in Apache Airflow ... 457
Versioni ... 459
Informazioni sulle versioni di Amazon MWAA ... 459
Versione più recente ... 459
Versioni di Apache Airflow ... 459
Componenti Apache Airflow ... 461
Pianificatori ... 461
Worker ... 461
Aggiornamento della versione di Apache Airflow ... 461
Versioni obsolete di Apache Airflow ... 462
Supporto e domande frequenti sulla versione di Apache Airflow ... 462
Domande frequenti ... 463
Endpoint e quote ... 465
Endpoint di servizio ... 465
Quote del servizio ... 465
Quote crescenti ... 466
Domande frequenti ... 467
Versioni supportate ... 468
Supporto per Apache Airflow ... 468
Versioni di Apache Airflow ... 468
Versione di Python ... 468

Versione Pip 469
Casi d'uso 470
Quando AWS Step Functions devo usare vs. Amazon MWAA? 470
Specifiche ambientali 470
Quanto spazio di archiviazione delle attività è disponibile per ogni ambiente? 470
Sistema operativo predefinito 470
Immagini personalizzate 470
Conformità agli standard HIPAA 471
Amazon MWAA supporta le istanze Spot? 471
Dominio personalizzato 471
Accesso SSH 471
Regola di autoreferenziazione 472
Parametri personalizzati 472
Memorizza i dati 472
Quota di lavoratori 472
VPC Amazon condivisi 473
Metriche 473
Metriche relative ai lavoratori 473
Parametri personalizzati 473
DAG, operatori, connessioni e altre domande 473
PythonVirtualenvOperator 473
Quanto tempo impiega Amazon MWAA a riconoscere un nuovo file DAG? 473
Perché il mio file DAG non viene prelevato da Apache Airflow? 474
Rimuovere plugins.zip o requirements.txt 474
Rimuovi plugins.zip o requirements.txt 474
Posso usare gli operatori AWS del Database Migration Service (DMS)? 475
Risoluzione dei problemi 476
Apache Airflow v2 479
Connessioni 479
Server Web 482
Processi 483
CLI 486
Operatori 487
Apache Airflow v1 489
Aggiornamento di requirements.txt 490
DAG rotto 490

Operatori 492
Connessioni 493
Server Web 495
Processi 497
CLI 500
Creazione/aggiornamento di Amazon MWAA 500
Aggiornamento degli `requirements.txt` 501
Plug-in 502
Creare bucket. 502
Creazione dell'ambiente 503
Ambiente di aggiornamento 506
Ambiente di accesso 507
CloudWatch Registri e CloudTrail 507
Log 508
Cronologia dei documenti 514
..... dlxxxiv

Che cos'è Amazon Managed Workflows per Apache Airflow?

Amazon Managed Workflows for Apache Airflow è un servizio di orchestrazione gestito per Apache Airflow che puoi utilizzare per configurare e gestire pipeline di dati nel cloud su larga scala. Apache Airflow è uno strumento open source utilizzato per creare, pianificare e monitorare in modo programmatico sequenze di processi e attività denominate flussi di lavoro. Con Amazon MWAA, puoi usare Apache Airflow e Python per creare flussi di lavoro senza dover gestire l'infrastruttura sottostante per scalabilità, disponibilità e sicurezza. Amazon MWAA ridimensiona automaticamente la capacità di esecuzione del flusso di lavoro per soddisfare le tue esigenze, Amazon MWAA si integra con i servizi di AWS sicurezza per aiutarti a fornire un accesso rapido e sicuro ai tuoi dati.

Contenuti

- Funzionalità
- Architettura
- Integrazione
- Versioni supportate
- Fasi successive

Funzionalità

- Configurazione automatica del flusso d'aria: configura rapidamente Apache Airflow scegliendo una versione di Apache Airflow quando crei un ambiente Amazon MWAA. Amazon MWAA configura Apache Airflow per te utilizzando la stessa interfaccia utente Apache Airflow e lo stesso codice open source che puoi scaricare da Internet.
- Scalabilità automatica: ridimensiona automaticamente Apache Airflow Workers impostando il numero minimo e massimo di Worker in esecuzione nel tuo ambiente. Amazon MWAA monitora i Worker nel tuo ambiente e utilizza il componente di scalabilità automatica per aggiungere Workers in base alla domanda, fino a raggiungere il numero massimo di Worker da te definito.
- Autenticazione integrata: abilita l'autenticazione e l'autorizzazione basate sui ruoli per il tuo server Web Apache Airflow definendo le politiche di controllo degli accessi in (IAM). AWS Identity and Access Management Gli Apache Airflow Workers assumono queste politiche per un accesso sicuro ai servizi. AWS

- Sicurezza integrata: gli Apache Airflow Workers and Scheduler vengono eseguiti nell'Amazon VPC di Amazon MWAA. Inoltre, i dati vengono crittografati automaticamente utilizzando AWS Key Management Service, quindi l'ambiente è sicuro per impostazione predefinita.
- Modalità di accesso pubblico o privato: accedi al tuo server Web Apache Airflow utilizzando una modalità di accesso privata o pubblica. La modalità di accesso alla rete pubblica utilizza un endpoint VPC per il server Web Apache Airflow accessibile tramite Internet. La modalità di accesso alla rete privata utilizza un endpoint VPC per il server Web Apache Airflow accessibile nel VPC. In entrambi i casi, l'accesso per gli utenti di Apache Airflow è controllato dalla politica di controllo degli accessi definita in AWS Identity and Access Management (IAM) e SSO. AWS
- Aggiornamenti e patch semplificati: Amazon MWAA fornisce periodicamente nuove versioni di Apache Airflow. Il team di Amazon MWAA aggiornerà e correggerà le immagini per queste versioni.
- Monitoraggio del flusso di lavoro: visualizza i log di Apache Airflow e le metriche di Apache Airflow in CloudWatch Amazon per identificare i ritardi nelle attività di Apache Airflow o gli errori del flusso di lavoro senza la necessità di strumenti di terze parti aggiuntivi. Amazon MWAA invia automaticamente i parametri di ambiente e, se abilitato, i log ad Apache Airflow. CloudWatch
- AWS integrazione: Amazon MWAA supporta integrazioni open source con Amazon Athena, Amazon, Amazon CloudWatch DynamoDB AWS Batch, Amazon AWS DataSync EMR, Amazon EKS, Amazon Data Firehose,, AWS Fargate Amazon AWS Glue AWS Lambda Redshift, Amazon SQS, Amazon SNS, Amazon e Amazon S3, oltre a centinaia di soluzioni integrate operatori e sensori creati dalla comunità SageMaker.
- Flotte di lavoratori: Amazon MWAA offre supporto per l'utilizzo di container per scalare la flotta di lavoratori su richiesta e ridurre le interruzioni dello scheduler utilizzando Amazon ECS on. AWS Fargate Sono supportati gli operatori che richiamano attività sui contenitori Amazon ECS e gli operatori Kubernetes che creano ed eseguono pod su un cluster Kubernetes.

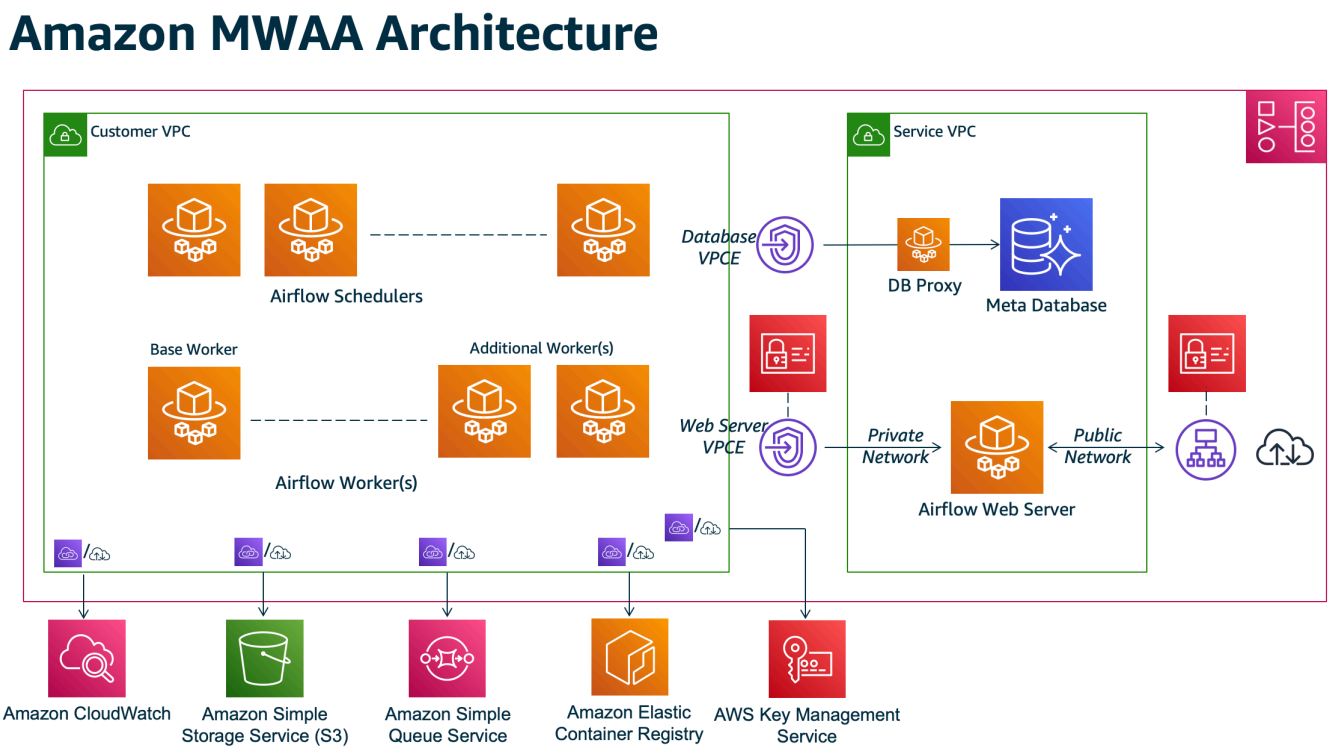
Architettura

Tutti i componenti contenuti nella confezione esterna (nell'immagine seguente) vengono visualizzati come un unico ambiente Amazon MWAA nel tuo account. Apache Airflow Scheduler e Workers sono AWS Fargate (Fargate) contenitori che si connettono alle sottoreti private di Amazon VPC per il tuo ambiente. Ogni ambiente ha il proprio metadatabase Apache Airflow gestito da AWS che è accessibile ai container Scheduler e Workers Fargate tramite un endpoint VPC protetto privatamente.

Amazon CloudWatch, Amazon S3, Amazon SQS, Amazon ECR AWS KMS e Amazon sono separati da Amazon MWAA e devono essere accessibili dai container Apache Airflow Scheduler e Workers in the Fargate.

È possibile accedere al server Web Apache Airflow tramite Internet selezionando la modalità di accesso Apache Airflow alla rete pubblica o all'interno del VPC selezionando la modalità di accesso Apache Airflow alla rete privata. In entrambi i casi, l'accesso per gli utenti di Apache Airflow è controllato dalla politica di controllo degli accessi definita in (IAM). AWS Identity and Access Management

> **Note**
>
> Più Apache Airflow Scheduler sono disponibili solo con Apache Airflow v2 e versioni successive. Scopri di più sul ciclo di vita delle attività di Apache Airflow su Concepts nella guida di riferimento di Apache Airflow.



Integrazione

La community open source di Apache Airflow, attiva e in crescita, fornisce operatori (plug-in che semplificano le connessioni ai servizi) che consentono l'integrazione di Apache Airflow con i servizi. AWS Ciò include servizi come Amazon S3, Amazon Redshift, Amazon AWS Batch EMR SageMaker e Amazon, oltre a servizi su altre piattaforme cloud.

L'uso di Apache Airflow con Amazon MWAA supporta completamente l'integrazione con AWS servizi e strumenti di terze parti diffusi come Apache Hadoop, Presto, Hive e Spark per eseguire attività di elaborazione dei dati. Amazon MWAA si impegna a mantenere la compatibilità con l'API Amazon MWAA e Amazon MWAA intende fornire integrazioni affidabili ai AWS servizi e renderli disponibili alla community, oltre a partecipare allo sviluppo di funzionalità della community.

Per il codice di esempio, consulta Esempi di codice per Amazon Managed Workflows for Apache Airflow.

Versioni supportate

Amazon MWAA supporta più versioni di Apache Airflow. Per ulteriori informazioni sulle versioni di Apache Airflow supportate e sui componenti di Apache Airflow inclusi in ciascuna versione, consulta. Versioni di Apache Airflow su Amazon Managed Workflows per Apache Airflow

Fasi successive

- Inizia con un unico AWS CloudFormation modello che crea un bucket Amazon S3 per i tuoi DAG Airflow e i file di supporto, un Amazon VPC con routing pubblico e un ambiente Amazon MWAA. Tutorial di avvio rapido per Amazon Managed Workflows for Apache Airflow
- Inizia in modo incrementale creando un bucket Amazon S3 per i tuoi DAG Airflow e i file di supporto, scegliendo tra una delle tre opzioni di rete Amazon VPC e creando un ambiente Amazon MWAA in. Inizia a usare Amazon Managed Workflows per Apache Airflow

Tutorial di avvio rapido per Amazon Managed Workflows for Apache Airflow

Questo tutorial di avvio rapido utilizza un AWS CloudFormation modello che crea contemporaneamente l'infrastruttura Amazon VPC, un bucket Amazon S3 con `dags` una cartella e un ambiente Amazon Managed Workflows for Apache Airflow.

Argomenti

- In questo tutorial
- Prerequisiti
- Fase uno: salvare il AWS CloudFormation modello localmente
- Fase due: creare lo stack utilizzando il AWS CLI
- Fase tre: caricare un DAG su Amazon S3 ed eseguirlo nell'interfaccia utente di Apache Airflow
- Fase quattro: Visualizza i log in Logs CloudWatch
- Fasi successive

In questo tutorial

Questo tutorial illustra tre AWS Command Line Interface (AWS CLI) comandi per caricare un DAG su Amazon S3, eseguire il DAG in Apache Airflow e visualizzare i log in. CloudWatch Conclude illustrandoti i passaggi per creare una policy IAM per un team di sviluppo di Apache Airflow.

> **Note**
>
> Il AWS CloudFormation modello in questa pagina crea un ambiente Amazon Managed Workflows for Apache Airflow per l'ultima versione di Apache Airflow disponibile in. AWS CloudFormation L'ultima versione disponibile è Apache Airflow v2.8.1.

Il AWS CloudFormation modello in questa pagina crea quanto segue:

- Infrastruttura VPC. Il modello utilizzaRouting pubblico su Internet. Utilizza il Modalità di accesso alla rete pubblica per il server Web Apache Airflow in. `WebserverAccessMode: PUBLIC_ONLY`

- Bucket Amazon S3. Il modello crea un bucket Amazon S3 con una cartella. `dags` È configurato per bloccare tutti gli accessi pubblici, con Bucket Versioning abilitato, come definito in. Creare are are are are are are are are are are are are are are are are are are are are are are are are are are are
- Ambiente Amazon MWAA. Il modello crea un ambiente Amazon MWAA associato alla `dags` cartella nel bucket Amazon S3, un ruolo di esecuzione con autorizzazione ai AWS servizi utilizzati da Amazon MWAA e l'impostazione predefinita per la crittografia utilizzando AWS una chiave di proprietà, come definito in. Crea un ambiente Amazon MWAA
- CloudWatch Registri. Il modello abilita gli accessi di Apache Airflow CloudWatch al livello «INFO» e superiore per il gruppo di log dello scheduler Airflow, il gruppo di log del server web Airflow, il gruppo di log di lavoro Airflow, il gruppo di log di elaborazione Airflow DAG e il gruppo di log delle attività Airflow, come definito in. Visualizzazione dei log di Airflow in Amazon CloudWatch

In questo tutorial, completerai le seguenti attività:

- Carica ed esegui un DAG. Carica il tutorial DAG di Apache Airflow per l'ultima versione di Apache Airflow supportata da Amazon MWAA su Amazon S3, quindi eseguilo nell'interfaccia utente di Apache Airflow, come definito in. Aggiunta o aggiornamento di DAG
- Visualizza i log. Visualizzate il gruppo di log del server web Airflow in CloudWatch Logs, come definito in. Visualizzazione dei log di Airflow in Amazon CloudWatch
- Crea una politica di controllo degli accessi. Crea una policy di controllo degli accessi in IAM per il tuo team di sviluppo di Apache Airflow, come definito in. Accesso a un ambiente Amazon MWAA

Prerequisiti

Il AWS Command Line Interface (AWS CLI) è uno strumento open source che consente di interagire con i AWS servizi utilizzando i comandi della shell della riga di comando. Per completare la procedura descritta in questa pagina, è necessario quanto segue:

- AWS CLI — Installa la versione 2.
- AWS CLI — Configurazione rapida con `aws configure`.

Fase uno: salvare il AWS CloudFormation modello localmente

- Copia il contenuto del seguente modello e salvalo localmente con nome`mwaa-public-network.yml`. Puoi anche scaricare il modello.

```
AWSTemplateFormatVersion: "2010-09-09"

Parameters:

  EnvironmentName:
    Description: An environment name that is prefixed to resource names
    Type: String
    Default: MWAAEnvironment

  VpcCIDR:
    Description: The IP range (CIDR notation) for this VPC
    Type: String
    Default: 10.192.0.0/16

  PublicSubnet1CIDR:
    Description: The IP range (CIDR notation) for the public subnet in the first
 Availability Zone
    Type: String
    Default: 10.192.10.0/24

  PublicSubnet2CIDR:
    Description: The IP range (CIDR notation) for the public subnet in the second
 Availability Zone
    Type: String
    Default: 10.192.11.0/24

  PrivateSubnet1CIDR:
    Description: The IP range (CIDR notation) for the private subnet in the first
 Availability Zone
    Type: String
    Default: 10.192.20.0/24
  PrivateSubnet2CIDR:
    Description: The IP range (CIDR notation) for the private subnet in the second
 Availability Zone
    Type: String
    Default: 10.192.21.0/24
  MaxWorkerNodes:
    Description: The maximum number of workers that can run in the environment
    Type: Number
    Default: 2
  DagProcessingLogs:
    Description: Log level for DagProcessing
    Type: String
```

```
    Default: INFO
  SchedulerLogsLevel:
    Description: Log level for SchedulerLogs
    Type: String
    Default: INFO
  TaskLogsLevel:
    Description: Log level for TaskLogs
    Type: String
    Default: INFO
  WorkerLogsLevel:
    Description: Log level for WorkerLogs
    Type: String
    Default: INFO
  WebserverLogsLevel:
    Description: Log level for WebserverLogs
    Type: String
    Default: INFO

Resources:

 ##################################################################################################
  # CREATE VPC

 ##################################################################################################

  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCIDR
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: MWAAEnvironment

  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: MWAAEnvironment

  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
```

```
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC

  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs '' ]
      CidrBlock: !Ref PublicSubnet1CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ1)

  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 1, !GetAZs  '' ]
      CidrBlock: !Ref PublicSubnet2CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ2)

  PrivateSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs  '' ]
      CidrBlock: !Ref PrivateSubnet1CIDR
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Subnet (AZ1)

  PrivateSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 1, !GetAZs  '' ]
      CidrBlock: !Ref PrivateSubnet2CIDR
      MapPublicIpOnLaunch: false
```

```
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Subnet (AZ2)

  NatGateway1EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc

  NatGateway2EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc

  NatGateway1:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway1EIP.AllocationId
      SubnetId: !Ref PublicSubnet1

  NatGateway2:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway2EIP.AllocationId
      SubnetId: !Ref PublicSubnet2

  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Routes

  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
```

```
  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1

  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2


  PrivateRouteTable1:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Routes (AZ1)

  DefaultPrivateRoute1:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway1

  PrivateSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      SubnetId: !Ref PrivateSubnet1

  PrivateRouteTable2:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Routes (AZ2)

  DefaultPrivateRoute2:
    Type: AWS::EC2::Route
```

```
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway2

  PrivateSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      SubnetId: !Ref PrivateSubnet2

  SecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: "mwaa-security-group"
      GroupDescription: "Security group with a self-referencing inbound rule."
      VpcId: !Ref VPC

  SecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !Ref SecurityGroup
      IpProtocol: "-1"
      SourceSecurityGroupId: !Ref SecurityGroup

  EnvironmentBucket:
    Type: AWS::S3::Bucket
    Properties:
      VersioningConfiguration:
        Status: Enabled
      PublicAccessBlockConfiguration:
        BlockPublicAcls: true
        BlockPublicPolicy: true
        IgnorePublicAcls: true
        RestrictPublicBuckets: true


 ##################################################################################################
  # CREATE MWAA

 ##################################################################################################

  MwaaEnvironment:
    Type: AWS::MWAA::Environment
```

```
    DependsOn: MwaaExecutionPolicy
    Properties:
      Name: !Sub "${AWS::StackName}-MwaaEnvironment"
      SourceBucketArn: !GetAtt EnvironmentBucket.Arn
      ExecutionRoleArn: !GetAtt MwaaExecutionRole.Arn
      DagS3Path: dags
      NetworkConfiguration:
        SecurityGroupIds:
          - !GetAtt SecurityGroup.GroupId
        SubnetIds:
          - !Ref PrivateSubnet1
          - !Ref PrivateSubnet2
      WebserverAccessMode: PUBLIC_ONLY
      MaxWorkers: !Ref MaxWorkerNodes
      LoggingConfiguration:
        DagProcessingLogs:
          LogLevel: !Ref DagProcessingLogs
          Enabled: true
        SchedulerLogs:
          LogLevel: !Ref SchedulerLogsLevel
          Enabled: true
        TaskLogs:
          LogLevel: !Ref TaskLogsLevel
          Enabled: true
        WorkerLogs:
          LogLevel: !Ref WorkerLogsLevel
          Enabled: true
        WebserverLogs:
          LogLevel: !Ref WebserverLogsLevel
          Enabled: true
  SecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: !Ref VPC
      GroupDescription: !Sub "Security Group for Amazon MWAA Environment
${AWS::StackName}-MwaaEnvironment"
      GroupName: !Sub "airflow-security-group-${AWS::StackName}-MwaaEnvironment"

  SecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !Ref SecurityGroup
      IpProtocol: "-1"
      SourceSecurityGroupId: !Ref SecurityGroup
```

```
  SecurityGroupEgress:
    Type: AWS::EC2::SecurityGroupEgress
    Properties:
      GroupId: !Ref SecurityGroup
      IpProtocol: "-1"
      CidrIp: "0.0.0.0/0"

  MwaaExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - airflow-env.amazonaws.com
                - airflow.amazonaws.com
            Action:
             - "sts:AssumeRole"
      Path: "/service-role/"

  MwaaExecutionPolicy:
    DependsOn: EnvironmentBucket
    Type: AWS::IAM::ManagedPolicy
    Properties:
      Roles:
        - !Ref MwaaExecutionRole
      PolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Action: airflow:PublishMetrics
            Resource:
              - !Sub "arn:aws:airflow:${AWS::Region}:${AWS::AccountId}:environment/
${EnvironmentName}"
          - Effect: Deny
            Action: s3:ListAllMyBuckets
            Resource:
              - !Sub "${EnvironmentBucket.Arn}"
              - !Sub "${EnvironmentBucket.Arn}/*"

          - Effect: Allow
```

```
            Action:
              - "s3:GetObject*"
              - "s3:GetBucket*"
              - "s3:List*"
            Resource:
              - !Sub "${EnvironmentBucket.Arn}"
              - !Sub "${EnvironmentBucket.Arn}/*"
          - Effect: Allow
            Action:
              - logs:DescribeLogGroups
            Resource: "*"

          - Effect: Allow
            Action:
              - logs:CreateLogStream
              - logs:CreateLogGroup
              - logs:PutLogEvents
              - logs:GetLogEvents
              - logs:GetLogRecord
              - logs:GetLogGroupFields
              - logs:GetQueryResults
              - logs:DescribeLogGroups
            Resource:
              - !Sub "arn:aws:logs:${AWS::Region}:${AWS::AccountId}:log-
group:airflow-${AWS::StackName}*"
          - Effect: Allow
            Action: cloudwatch:PutMetricData
            Resource: "*"
          - Effect: Allow
            Action:
              - sqs:ChangeMessageVisibility
              - sqs:DeleteMessage
              - sqs:GetQueueAttributes
              - sqs:GetQueueUrl
              - sqs:ReceiveMessage
              - sqs:SendMessage
            Resource:
              - !Sub "arn:aws:sqs:${AWS::Region}:*:airflow-celery-*"
          - Effect: Allow
            Action:
              - kms:Decrypt
              - kms:DescribeKey
              - "kms:GenerateDataKey*"
              - kms:Encrypt
```

```
            NotResource: !Sub "arn:aws:kms:*:${AWS::AccountId}:key/*"
            Condition:
              StringLike:
                "kms:ViaService":
                  - !Sub "sqs.${AWS::Region}.amazonaws.com"
Outputs:
  VPC:
    Description: A reference to the created VPC
    Value: !Ref VPC

  PublicSubnets:
    Description: A list of the public subnets
    Value: !Join [ ",", [ !Ref PublicSubnet1, !Ref PublicSubnet2 ]]

  PrivateSubnets:
    Description: A list of the private subnets
    Value: !Join [ ",", [ !Ref PrivateSubnet1, !Ref PrivateSubnet2 ]]

  PublicSubnet1:
    Description: A reference to the public subnet in the 1st Availability Zone
    Value: !Ref PublicSubnet1

  PublicSubnet2:
    Description: A reference to the public subnet in the 2nd Availability Zone
    Value: !Ref PublicSubnet2

  PrivateSubnet1:
    Description: A reference to the private subnet in the 1st Availability Zone
    Value: !Ref PrivateSubnet1

  PrivateSubnet2:
    Description: A reference to the private subnet in the 2nd Availability Zone
    Value: !Ref PrivateSubnet2

  SecurityGroupIngress:
    Description: Security group with self-referencing inbound rule
    Value: !Ref SecurityGroupIngress

  MwaaApacheAirflowUI:
    Description: MWAA Environment
    Value: !Sub  "https://${MwaaEnvironment.WebserverUrl}"
```

Fase due: creare lo stack utilizzando il AWS CLI

1. Nel prompt dei comandi, accedi alla directory in cui `mwaa-public-network.yml` è memorizzato. Per esempio:

```
cd mwaaproject
```

2. Usa il `aws cloudformation create-stack`comando per creare lo stack usando. AWS CLI

```
aws cloudformation create-stack --stack-name mwaa-environment-public-network --
template-body file://mwaa-public-network.yml --capabilities CAPABILITY_IAM
```

> **Note**
>
> Sono necessari più di 30 minuti per creare l'infrastruttura Amazon VPC, il bucket Amazon S3 e l'ambiente Amazon MWAA.

Fase tre: caricare un DAG su Amazon S3 ed eseguirlo nell'interfaccia utente di Apache Airflow

1. Copia il contenuto del `tutorial.py` file per l'ultima versione supportata di Apache Airflow e salvalo localmente come. `tutorial.py`
2. Nel prompt dei comandi, accedi alla directory in cui `tutorial.py` è memorizzato. Per esempio:

```
cd mwaaproject
```

3. Usa il seguente comando per elencare tutti i tuoi bucket Amazon S3.

```
aws s3 ls
```

4. Usa il seguente comando per elencare i file e le cartelle nel bucket Amazon S3 per il tuo ambiente.

```
aws s3 ls s3://YOUR_S3_BUCKET_NAME
```

5. Usa lo script seguente per caricare il `tutorial.py` file nella tua `dags` cartella. Sostituisci il valore di esempio in *YOUR_S3_BUCKET_NAME*.

```
aws s3 cp tutorial.py s3://YOUR_S3_BUCKET_NAME/dags/
```

6. Apri la pagina Ambienti sulla console Amazon MWAA.
7. Scegli un ambiente.
8. Scegli Open Airflow UI.
9. Nell'interfaccia utente di Apache Airflow, dall'elenco dei DAG disponibili, scegli il tutorial DAG.
10. Nella pagina dei dettagli del DAG, scegliete l'interruttore Pause/Unpause DAG accanto al nome del DAG per riattivare il DAG.
11. Scegliete Trigger DAG.

Fase quattro: Visualizza i log in Logs CloudWatch

È possibile visualizzare i log di Apache Airflow nella CloudWatch console per tutti i log di Apache Airflow abilitati dallo stack. AWS CloudFormation La sezione seguente mostra come visualizzare i log per il gruppo di log del server web Airflow.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli il gruppo di log del server web Airflow nel riquadro Monitoraggio.
4. Scegli il `webserver_console_ip` log in Log Streams.

Fasi successive

- Scopri di più su come caricare DAG, specificare le dipendenze Python in a `requirements.txt` e i plugin personalizzati in un in. `plugins.zip` Utilizzo dei DAG su Amazon MWAA
- Scopri di più sulle best practice che consigliamo per ottimizzare le prestazioni del tuo ambiente. Ottimizzazione delle prestazioni per Apache Airflow su Amazon MWAA
- Crea una dashboard di monitoraggio per il tuo ambiente inPannelli di controllo e allarmi su Amazon MWAA.
- Esegui alcuni esempi di codice DAG inEsempi di codice per Amazon Managed Workflows for Apache Airflow.

Inizia a usare Amazon Managed Workflows per Apache Airflow

Amazon Managed Workflow per Apache Airflow utilizza Amazon VPC, codice DAG e file di supporto nel bucket di storage Amazon S3 per creare un ambiente. Questa guida descrive i prerequisiti e leAWS risorse necessarie per iniziare a usare Amazon MWAA.

Argomenti

- Prerequisiti
- Informazioni sulla guida
- Prima di iniziare
- Regioni disponibili
- Creare are are are are are are are are are are are are are are are are are are are are are are are
- Crea la rete VPC
- Crea un ambiente Amazon MWAA
- Fasi successive

Prerequisiti

Per creare un ambiente Amazon MWAA, potresti voler adottare misure aggiuntive per assicurarti di avere l'autorizzazione alleAWS risorse che devi creare.

- AWSaccount: unAWS account autorizzato a utilizzare Amazon MWAA e iAWS servizi e le risorse utilizzati dal tuo ambiente.

Informazioni sulla guida

Questa sezione descrive l'AWSinfrastruttura e le risorse che creerai in questa guida.

- Amazon VPC: i componenti di rete Amazon VPC richiesti da un ambiente Amazon MWAA. È possibile configurare un cloud privato virtuale esistente che soddisfi questi requisiti (avanzati) come mostrato inInformazioni sulla rete su Amazon MWAA, oppure creare il VPC e i componenti di rete, come definito inthe section called “Crea la rete VPC”.

- Bucket Amazon S3: un bucket Amazon S3 per archiviare i DAG e i file associati, ad esempio`plugins.zip` e`requirements.txt`. Il tuo bucket Amazon S3 deve essere configurato per bloccare tutti gli accessi pubblici, con il Bucket Versioning abilitato, come definito inCreare are are are are are are are are are are are are are are are are are are are are are are.
- Ambiente Amazon MWAA: un ambiente Amazon MWAA configurato con la posizione del bucket Amazon S3, il percorso del codice DAG ed eventuali plugin personalizzati o dipendenze Python e il tuo Amazon VPC e il relativo gruppo di sicurezza, come definito inCrea un ambiente Amazon MWAA.

Prima di iniziare

Per creare un ambiente Amazon MWAA, potresti voler eseguire ulteriori passaggi per creare e configurare altreAWS risorse prima di creare il tuo ambiente.

Per creare un ambiente, occorre quanto segue:

- AWS KMSchiave: unaAWS KMS chiave per la crittografia dei dati nel tuo ambiente. Puoi scegliere l'opzione predefinita sulla console Amazon MWAA per creare una chiave diAWS proprietà quando crei un ambiente o specificare una chiave gestita dal Cliente esistente con autorizzazioni per altriAWS servizi utilizzati dal tuo ambiente configurata (avanzata). Per ulteriori informazioni, consulta Utilizzo di chiavi gestite dal cliente per la crittografia.
- Ruolo di esecuzione: un ruolo di esecuzione che consente ad Amazon MWAA di accedere alleAWS risorse del tuo ambiente. Puoi scegliere l'opzione predefinita sulla console Amazon MWAA per creare un ruolo di esecuzione quando crei un ambiente. Per ulteriori informazioni, consulta Ruolo di esecuzione di Amazon MWAA.
- Gruppo di sicurezza VPC: un gruppo di sicurezza VPC che consente ad Amazon MWAA di accedere ad altreAWS risorse della rete VPC. Puoi scegliere l'opzione predefinita sulla console Amazon MWAA per creare un gruppo di sicurezza quando crei un ambiente o fornire a un gruppo di sicurezza le regole in entrata e in uscita appropriate (avanzate). Per ulteriori informazioni, consulta Sicurezza nel tuo VPC su Amazon MWAA.

Regioni disponibili

Amazon MWAA è nelle seguentiAWS regioni.

- Europa (Stoccolma) - eu-north-1

- Europa (Francoforte) - eu-central-1
- Europa (Irlanda) - eu-west-1
- Europa (Londra) - eu-west-2
- Europa (Parigi) - eu-west-3
- Asia Pacifico (Mumbai) - ap-south-1
- Asia Pacifico (Singapore) - ap-southeast-1
- Asia Pacifico (Sydney): ap-southeast-2
- Asia Pacifico (Tokyo), ap-northeast-1
- Asia Pacifico (Seoul) - ap-northeast-2
- Stati Uniti orientali (Virginia settentrionale), us-east-1
- Stati Uniti orientali (Ohio): us-east-2
- Stati Uniti occidentali (Oregon): us-west-2
- Canada (Centrale) - ca-central-1
- Sud America (San Paolo) - sa-east-1

Creare are are are are are are are are are are are are are are are are are are are are are are are

Questa guida descrive i passaggi per creare un bucket Amazon S3 per archiviare i grafici aciclici diretti (DAG) di Apache Airflow, i plugin personalizzati in un file e le dipendenze Python in un file. `plugins.zip` `requirements.txt`

Indice

- Prima di iniziare
- Creare are are are are are are are
- Fasi successive

Prima di iniziare

- Il et Amazon S3 non può essere modificato dopo la creazione et et Amazon S3 Per ulteriori informazioni, consulta le regole per la denominazione dei bucket nella Guida per l'utente di Amazon Simple Storage Service.

- Un bucket Amazon S3 utilizzato per un ambiente Amazon MWAA deve essere configurato per bloccare tutti gli accessi pubblici, con Bucket Versioning abilitato.
- Un bucket Amazon S3 utilizzato per un ambiente Amazon MWAA deve trovarsi nella stessa AWS regione di un ambiente Amazon MWAA. Per visualizzare un elenco di AWS regioni per Amazon MWAA, consulta gli endpoint e le quote di Amazon MWAA nel. Riferimenti generali di AWS

Creare are are are are are are

Questa sezione Amazon S3 per il et Amazon S3 per il et Amazon S3 per il

Per creare un bucket

1. Accedi alla AWS Management Console e apri la console di Amazon S3 all'indirizzo https://console.aws.amazon.com/s3/.
2. Scegliere Create bucket (Crea bucket).
3. In Bucket name (Nome bucket), immettere un nome conforme a DNS per il bucket.

   Il nome del bucket deve:

   - Essere univoco in tutto Amazon S3.
   - Deve contenere da 3 a 63 caratteri
   - Non contiene caratteri maiuscoli.
   - Iniziare con una lettera minuscola o un numero.

   > **Important**
   >
   > Evitare di includere informazioni riservate, ad esempio numeri di account, nel nome del bucket. Il nome bucket è visibile nell'URL che punta agli oggetti nel bucket.

4. Scegli una AWS regione nella regione. Deve corrispondere alla stessa AWS regione del tuo ambiente Amazon MWAA.

   - Ti consigliamo di scegliere una regione nelle vicinanze per ridurre al minimo la latenza e i costi o rispondere a requisiti normativi

5. Scegliere Block all public access (Blocca tutti gli accessi pubblici).
6. Scegli Abilita in Bucket Versioning.

7. Facoltativo: tag. Aggiungi coppie di tag chiave-valore per identificare il tuo bucket Amazon S3 in Tags. Ad esempio,`Bucket:Staging`.
8. Facoltativo: crittografia lato server. Facoltativamente, puoi abilitare una delle seguenti opzioni di crittografia sul tuo bucket Amazon S3.
   a. Scegli Amazon S3 (SSE-S3) in Tipo di chiave Amazon S3 per il et et et et et et et et et
   b. Scegli AWS Key Management Servicela chiave (SSE-KMS) per utilizzare una AWS KMS chiave per la crittografia sul tuo bucket Amazon S3:
      i. AWSchiave gestita (aws/s3): se scegli questa opzione, puoi utilizzare una chiave di AWSproprietà gestita da Amazon MWAA o specificare una chiave gestita dal Cliente per la crittografia del tuo ambiente Amazon MWAA.
      ii. Scegli tra AWS KMS le tue chiavi o Inserisci AWS KMS la chiave ARN: se scegli di specificare una chiave gestita dal cliente in questo passaggio, devi specificare un ID AWS KMS chiave o un ARN. AWS KMSgli alias e le chiavi multiregionali non sono supportati da Amazon MWAA. La AWS KMS chiave specificata deve essere utilizzata anche per la crittografia nel tuo ambiente Amazon MWAA.
9. Opzionale: impostazioni avanzate. Se si desidera abilitare Amazon S3:
   a. Scegli Impostazioni avanzate, Abilita.

      > **Important**
      >
      > L'attivazione di Object Lock consentirà il blocco permanente degli oggetti in questo bucket. Per ulteriori informazioni, consulta Blocare di oggetti Amazon S3 nella Guida per l'utente di Amazon S3 nella Guida per l'utente di Amazon S3 nella Guida per l'utente di Amazon S3 nella Guida per l'utente di

   b. Scegli il riconoscimento.
10. Seleziona Create bucket (Crea bucket).

Fasi successive

- Scopri come creare la rete Amazon VPC richiesta per un ambiente inCrea la rete VPC.
- Scopri come gestire le autorizzazioni di accesso in Come impostare le autorizzazioni dei bucket ACL?

- Scopri come eliminare et et et et et et et et et et et et et et et et et et et et et et et et et et et et et et et S3? .

Crea la rete VPC

Amazon Managed Workflows per Apache Airflow richiede un Amazon VPC e componenti di rete specifici per supportare un ambiente. Questa guida descrive le diverse opzioni per creare la rete Amazon VPC per un ambiente Amazon Managed Workflows for Apache Airflow.

> **Note**
>
> Apache Airflow funziona al meglio in un ambiente di rete a bassa latenza. Se utilizzi un Amazon VPC esistente che indirizza il traffico verso un'altra regione o verso un ambiente locale, ti consigliamo di aggiungereAWS PrivateLinkendpoint per Amazon SQS, CloudWatch, Amazon S3,AWS KMSe Amazon ECR. Per ulteriori informazioni sulla configurazioneAWS PrivateLinkper Amazon MWAA, vediCreazione di una rete Amazon VPC senza accesso a Internet.

Indice

- Prerequisiti
- Prima di iniziare
- Opzioni per creare la rete Amazon VPC
  - Opzione uno: creazione della rete VPC sulla console Amazon MWAA
  - Opzione due: creazione di una rete Amazon VPCconaccesso a Internet
  - Opzione tre: creazione di una rete Amazon VPCsenzaaccesso a Internet
- Fasi successive

Prerequisiti

AWS Command Line Interface (AWS CLI) è uno strumento open source che consente di interagire con i servizi AWS utilizzando i comandi nella shell a riga di comando. Per completare la procedura descritta in questa pagina, è necessario quanto segue:

- AWS CLI— Installa la versione 2.

- AWS CLI— Configurazione rapida con`aws configure`.

Prima di iniziare

- LaRete VPCle impostazioni specificate per l'ambiente non possono essere modificate dopo la creazione dell'ambiente.
- Puoi utilizzare il routing privato o pubblico per Amazon VPC e Apache AirflowServer Web. Per visualizzare un elenco di opzioni, vederethe section called “Esempi di casi d'uso per una modalità di accesso Amazon VPC e Apache Airflow”.

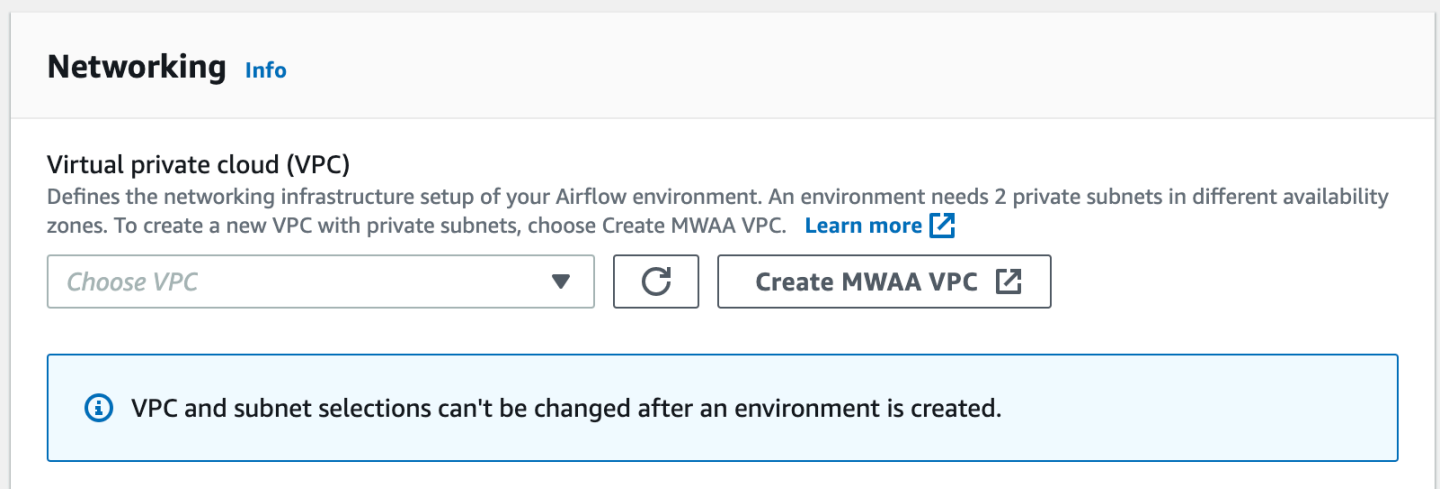
Opzioni per creare la rete Amazon VPC

La sezione seguente descrive le opzioni disponibili per creare la rete Amazon VPC per un ambiente.

Opzione uno: creazione della rete VPC sulla console Amazon MWAA

La sezione seguente mostra come creare una rete Amazon VPC sulla console Amazon MWAA. Questa opzione utilizzaRouting pubblico su Internet. Può essere usato per un Apache AirflowServer Webcon ilrete privataoRete pubblicamodalità di accesso.

L'immagine seguente mostra dove è possibile trovare ilCrea MWAA VPCpulsante sulla console Amazon MWAA.



Opzione due: creazione di una rete Amazon VPCconaccesso a Internet

Quanto segueAWS CloudFormationil modello crea una rete Amazon VPCcon accesso a Internetnella tua impostazione predefinitaAWSRegione. Questa opzione utilizzaRouting pubblico su Internet.

Questo modello può essere utilizzato per un Apache AirflowServer Webcon ilrete privataoRete pubblicamodalità di accesso.

1. Copia il contenuto del seguente modello e salvalo localmente come`cfn-vpc-public-private.yaml`. Puoi anchescarica il modello.

```
Description:  This template deploys a VPC, with a pair of public and private
 subnets spread
  across two Availability Zones. It deploys an internet gateway, with a default
  route on the public subnets. It deploys a pair of NAT gateways (one in each AZ),
  and default routes for them in the private subnets.

Parameters:
  EnvironmentName:
    Description: An environment name that is prefixed to resource names
    Type: String
    Default: mwaa-

  VpcCIDR:
    Description: Please enter the IP range (CIDR notation) for this VPC
    Type: String
    Default: 10.192.0.0/16

  PublicSubnet1CIDR:
    Description: Please enter the IP range (CIDR notation) for the public subnet in
 the first Availability Zone
    Type: String
    Default: 10.192.10.0/24

  PublicSubnet2CIDR:
    Description: Please enter the IP range (CIDR notation) for the public subnet in
 the second Availability Zone
    Type: String
    Default: 10.192.11.0/24

  PrivateSubnet1CIDR:
    Description: Please enter the IP range (CIDR notation) for the private subnet
 in the first Availability Zone
    Type: String
    Default: 10.192.20.0/24

  PrivateSubnet2CIDR:
```

```
    Description: Please enter the IP range (CIDR notation) for the private subnet
 in the second Availability Zone
    Type: String
    Default: 10.192.21.0/24

Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCIDR
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName

  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC

  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs '' ]
      CidrBlock: !Ref PublicSubnet1CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ1)

  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
```

```
      AvailabilityZone: !Select [ 1, !GetAZs  '' ]
      CidrBlock: !Ref PublicSubnet2CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Subnet (AZ2)

  PrivateSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs  '' ]
      CidrBlock: !Ref PrivateSubnet1CIDR
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Subnet (AZ1)

  PrivateSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 1, !GetAZs  '' ]
      CidrBlock: !Ref PrivateSubnet2CIDR
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Subnet (AZ2)

  NatGateway1EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc

  NatGateway2EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc

  NatGateway1:
    Type: AWS::EC2::NatGateway
    Properties:
```

```
      AllocationId: !GetAtt NatGateway1EIP.AllocationId
      SubnetId: !Ref PublicSubnet1

  NatGateway2:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway2EIP.AllocationId
      SubnetId: !Ref PublicSubnet2

  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Public Routes

  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway

  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1

  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2


  PrivateRouteTable1:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
```

```
          Value: !Sub ${EnvironmentName} Private Routes (AZ1)

  DefaultPrivateRoute1:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway1

  PrivateSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      SubnetId: !Ref PrivateSubnet1

  PrivateRouteTable2:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub ${EnvironmentName} Private Routes (AZ2)

  DefaultPrivateRoute2:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway2

  PrivateSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      SubnetId: !Ref PrivateSubnet2

  SecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: "mwaa-security-group"
      GroupDescription: "Security group with a self-referencing inbound rule."
      VpcId: !Ref VPC

  SecurityGroupIngress:
```

```
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !Ref SecurityGroup
      IpProtocol: "-1"
      SourceSecurityGroupId: !Ref SecurityGroup

Outputs:
  VPC:
    Description: A reference to the created VPC
    Value: !Ref VPC

  PublicSubnets:
    Description: A list of the public subnets
    Value: !Join [ ",", [ !Ref PublicSubnet1, !Ref PublicSubnet2 ]]

  PrivateSubnets:
    Description: A list of the private subnets
    Value: !Join [ ",", [ !Ref PrivateSubnet1, !Ref PrivateSubnet2 ]]

  PublicSubnet1:
    Description: A reference to the public subnet in the 1st Availability Zone
    Value: !Ref PublicSubnet1

  PublicSubnet2:
    Description: A reference to the public subnet in the 2nd Availability Zone
    Value: !Ref PublicSubnet2

  PrivateSubnet1:
    Description: A reference to the private subnet in the 1st Availability Zone
    Value: !Ref PrivateSubnet1

  PrivateSubnet2:
    Description: A reference to the private subnet in the 2nd Availability Zone
    Value: !Ref PrivateSubnet2

  SecurityGroupIngress:
    Description: Security group with self-referencing inbound rule
    Value: !Ref SecurityGroupIngress
```

2. Nel prompt dei comandi, vai alla directory in cui`cfn-vpc-public-private.yaml`è memorizzato. Ad esempio:

```
cd mwaaproject
```

3. Usa il`aws cloudformation create-stack`comando per creare lo stack usando ilAWS CLI.

```
aws cloudformation create-stack --stack-name mwaa-environment --template-body
 file://cfn-vpc-public-private.yaml
```

> **Note**
>
> Sono necessari circa 30 minuti per creare l'infrastruttura Amazon VPC.

Opzione tre: creazione di una rete Amazon VPCsenzaaccesso a Internet

Quanto segueAWS CloudFormationil modello crea una rete Amazon VPCsenza accesso a Internetnella tua impostazione predefinitaAWSregione.

> **Important**
>
> Quando utilizzi un Amazon VPC senza accesso a Internet, devi concedere l'autorizzazione ad Amazon ECR per accedere ad Amazon S3 utilizzando un endpoint gateway. Puoi creare un endpoint gateway effettuando le seguenti operazioni:
>
> 1. Copia quanto segue`JSON`Policy IAM e salvala localmente come`s3-gw-endpoint-policy.json`. La policy concede l'autorizzazione minima richiesta per Amazon ECR per accedere alle risorse Amazon S3.
>
>    ```
>    {
>      "Statement": [
>        {
>          "Sid": "Access-to-specific-bucket-only",
>          "Principal": "*",
>          "Action": [
>            "s3:GetObject"
>          ],
>          "Effect": "Allow",
>          "Resource": ["arn:aws:s3:::prod-region-starport-layer-bucket/*"]
>        }
>      ]
>    }
>    ```

2. Crea l'endpoint utilizzando quanto segueAWS CLIcomando. Sostituisci i valori per`--vpc-ide`con`--route-table-ids`con le informazioni per il tuo Amazon VPC. Sostituisci`--service-name`con il nome in base alla tua regione.

```
$ aws ec2 create-vpc-endpoint --vpc-id vpc-1a2b3c4d \
--service-name com.amazonaws.us-west-2.s3 \
--route-table-ids rtb-11aa22bb \
--vpc-endpoint-type Gateway \
--policy-document file://s3-gw-endpoint-policy.json
```

Per ulteriori informazioni sulla creazione di endpoint gateway Amazon S3 per Amazon ECR, consultaCrea l'endpoint gateway Amazon S3nelGuida per l'utente di Amazon Elastic Container Registry.

Questa opzione utilizzaRouting privato senza accesso a Internet. Questo modello può essere utilizzato per un Apache AirflowServer Webcon ilrete privatasolo modalità di accesso. Crea il necessarioEndpoint VPC perAWSservizi utilizzati da un ambiente.

1. Copiare il contenuto del seguente modello e salvarlo localmente come`cfn-vpc-private.yaml`. Puoi anchescarica il modello.

```
AWSTemplateFormatVersion: "2010-09-09"

Parameters:
   VpcCIDR:
     Description: The IP range (CIDR notation) for this VPC
     Type: String
     Default: 10.192.0.0/16

   PrivateSubnet1CIDR:
     Description: The IP range (CIDR notation) for the private subnet in the first
 Availability Zone
     Type: String
     Default: 10.192.10.0/24

   PrivateSubnet2CIDR:
     Description: The IP range (CIDR notation) for the private subnet in the second
 Availability Zone
     Type: String
```

```
    Default: 10.192.11.0/24

Resources:
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCIDR
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
       - Key: Name
         Value: !Ref AWS::StackName

  RouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
       - Key: Name
         Value: !Sub "${AWS::StackName}-route-table"

  PrivateSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 0, !GetAZs  '' ]
      CidrBlock: !Ref PrivateSubnet1CIDR
      MapPublicIpOnLaunch: false
      Tags:
       - Key: Name
         Value: !Sub "${AWS::StackName} Private Subnet (AZ1)"

  PrivateSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [ 1, !GetAZs  '' ]
      CidrBlock: !Ref PrivateSubnet2CIDR
      MapPublicIpOnLaunch: false
      Tags:
       - Key: Name
         Value: !Sub "${AWS::StackName} Private Subnet (AZ2)"

  PrivateSubnet1RouteTableAssociation:
```

```
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref RouteTable
      SubnetId: !Ref PrivateSubnet1

  PrivateSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref RouteTable
      SubnetId: !Ref PrivateSubnet2

  S3VpcEndoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub "com.amazonaws.${AWS::Region}.s3"
      VpcEndpointType: Gateway
      VpcId: !Ref VPC
      RouteTableIds:
       - !Ref RouteTable

  SecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      VpcId: !Ref VPC
      GroupDescription: Security Group for Amazon MWAA Environments to access VPC
endpoints
      GroupName: !Sub "${AWS::StackName}-mwaa-vpc-endpoints"

  SecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !Ref SecurityGroup
      IpProtocol: "-1"
      SourceSecurityGroupId: !Ref SecurityGroup

  SqsVpcEndoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub "com.amazonaws.${AWS::Region}.sqs"
      VpcEndpointType: Interface
      VpcId: !Ref VPC
      PrivateDnsEnabled: true
      SubnetIds:
       - !Ref PrivateSubnet1
```

```
        - !Ref PrivateSubnet2
      SecurityGroupIds:
        - !Ref SecurityGroup

  CloudWatchLogsVpcEndoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub "com.amazonaws.${AWS::Region}.logs"
      VpcEndpointType: Interface
      VpcId: !Ref VPC
      PrivateDnsEnabled: true
      SubnetIds:
        - !Ref PrivateSubnet1
        - !Ref PrivateSubnet2
      SecurityGroupIds:
        - !Ref SecurityGroup

  CloudWatchMonitoringVpcEndoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub "com.amazonaws.${AWS::Region}.monitoring"
      VpcEndpointType: Interface
      VpcId: !Ref VPC
      PrivateDnsEnabled: true
      SubnetIds:
        - !Ref PrivateSubnet1
        - !Ref PrivateSubnet2
      SecurityGroupIds:
        - !Ref SecurityGroup

  KmsVpcEndoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub "com.amazonaws.${AWS::Region}.kms"
      VpcEndpointType: Interface
      VpcId: !Ref VPC
      PrivateDnsEnabled: true
      SubnetIds:
        - !Ref PrivateSubnet1
        - !Ref PrivateSubnet2
      SecurityGroupIds:
        - !Ref SecurityGroup

  EcrApiVpcEndoint:
```

```
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub "com.amazonaws.${AWS::Region}.ecr.api"
      VpcEndpointType: Interface
      VpcId: !Ref VPC
      PrivateDnsEnabled: true
      SubnetIds:
       - !Ref PrivateSubnet1
       - !Ref PrivateSubnet2
      SecurityGroupIds:
       - !Ref SecurityGroup

  EcrDkrVpcEndoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub "com.amazonaws.${AWS::Region}.ecr.dkr"
      VpcEndpointType: Interface
      VpcId: !Ref VPC
      PrivateDnsEnabled: true
      SubnetIds:
       - !Ref PrivateSubnet1
       - !Ref PrivateSubnet2
      SecurityGroupIds:
       - !Ref SecurityGroup

  AirflowApiVpcEndoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub "com.amazonaws.${AWS::Region}.airflow.api"
      VpcEndpointType: Interface
      VpcId: !Ref VPC
      PrivateDnsEnabled: true
      SubnetIds:
       - !Ref PrivateSubnet1
       - !Ref PrivateSubnet2
      SecurityGroupIds:
       - !Ref SecurityGroup

  AirflowEnvVpcEndoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      ServiceName: !Sub "com.amazonaws.${AWS::Region}.airflow.env"
      VpcEndpointType: Interface
      VpcId: !Ref VPC
```

```
        PrivateDnsEnabled: true
        SubnetIds:
         - !Ref PrivateSubnet1
         - !Ref PrivateSubnet2
        SecurityGroupIds:
         - !Ref SecurityGroup

Outputs:
   VPC:
     Description: A reference to the created VPC
     Value: !Ref VPC

   MwaaSecurityGroupId:
     Description: Associates the Security Group to the environment to allow access
 to the VPC endpoints
     Value: !Ref SecurityGroup

   PrivateSubnets:
     Description: A list of the private subnets
     Value: !Join [ ",", [ !Ref PrivateSubnet1, !Ref PrivateSubnet2 ]]

   PrivateSubnet1:
     Description: A reference to the private subnet in the 1st Availability Zone
     Value: !Ref PrivateSubnet1

   PrivateSubnet2:
     Description: A reference to the private subnet in the 2nd Availability Zone
     Value: !Ref PrivateSubnet2
```

2. Nel prompt dei comandi, vai alla directory in cui`cfn-vpc-private.yml`è memorizzato. Ad esempio:

```
cd mwaaproject
```

3. Usa il`aws cloudformation create-stack`comando per creare lo stack usando ilAWS CLI.

```
aws cloudformation create-stack --stack-name mwaa-private-environment --template-
body file://cfn-vpc-private.yml
```

> **Note**
>
> Sono necessari circa 30 minuti per creare l'infrastruttura Amazon VPC.

4. Dovrai creare un meccanismo per accedere a questi endpoint VPC dal tuo computer. Per ulteriori informazioni, consulta Gestione dell'accesso agli endpoint Amazon VPC specifici del servizio su Amazon MWAA.

> **Note**
>
> Puoi limitare ulteriormente l'accesso in uscita nel CIDR del tuo gruppo di sicurezza Amazon MWAA. Ad esempio, puoi limitarti a se stesso aggiungendo una regola in uscita autoreferenziale, laelenco di prefissiper Amazon S3 e il CIDR del tuo Amazon VPC.

Fasi successive

- Scopri come creare un ambiente Amazon MWAA inCrea un ambiente Amazon MWAA.
- Scopri come creare un tunnel VPN dal tuo computer al tuo Amazon VPC con routing privato inTutorial: Configurazione dell'accesso alla rete privata utilizzando unAWS Client VPN.

Crea un ambiente Amazon MWAA

Amazon Managed Workflows for Apache Airflow configura Apache Airflow su un ambiente nella versione prescelta utilizzando lo stesso Apache Airflow open source e la stessa interfaccia utente disponibili in Apache. Questa guida descrive i passaggi per creare un ambiente Amazon MWAA.

Indice

- Prima di iniziare
- Versioni Apache Airflow
- Creazione di un ambiente
  - Fase uno: specificare i dettagli
  - Fase due: configurare le impostazioni avanzate
  - Fase tre: revisione e creazione

Prima di iniziare

- La rete VPC specificata per l'ambiente non può essere modificata dopo la creazione dell'ambiente.
- È necessario un bucket Amazon S3 configurato per bloccare tutti gli accessi pubblici, con Bucket Versioning abilitato.
- È necessario un AWS account con autorizzazioni per utilizzare Amazon MWAA e autorizzazione in AWS Identity and Access Management (IAM) per creare ruoli IAM. Se scegli la modalità di accesso alla rete privata per il server Web Apache Airflow, che limita l'accesso ad Apache Airflow all'interno del tuo Amazon VPC, avrai bisogno dell'autorizzazione in IAM per creare endpoint Amazon VPC.

Versioni Apache Airflow

Le seguenti versioni di Apache Airflow sono supportate su Amazon Managed Workflows for Apache Airflow.

> **Note**
>
> - A partire da Apache Airflow v2.2.2, Amazon MWAA supporta l'installazione di requisiti Python, pacchetti provider e plug-in personalizzati direttamente sul server Web Apache Airflow.
> - A partire da Apache Airflow v2.7.2, il file dei requisiti deve includere una dichiarazione. `--constraint` Se non fornisci un vincolo, Amazon MWAA te ne specificherà uno per garantire che i pacchetti elencati nei tuoi requisiti siano compatibili con la versione di Apache Airflow che stai utilizzando.
>
>   Per ulteriori informazioni sull'impostazione dei vincoli nel file dei requisiti, consulta Installazione delle dipendenze in Python.

Versione Apache Airflow	Guida Apache Airflow	Vincoli di Apache Airflow	Versione di Python
v2.8.1	Guida di riferimento di Apache Airflow v2.8.1	File dei vincoli di Apache Airflow v2.8.1	Python 3.11

Versione Apache Airflow	Guida Apache Airflow	Vincoli di Apache Airflow	Versione di Python
v2.7.2	Guida di riferimento di Apache Airflow v2.7.2	File dei vincoli di Apache Airflow v2.7.2	Python 3.11
v2.6.3	Guida di riferimento di Apache Airflow v2.6.3	File dei vincoli di Apache Airflow v2.6.3	Python 3.10
v2.5.1	Guida di riferimento di Apache Airflow v2.5.1	File dei vincoli di Apache Airflow v2.5.1	Python 3.10
v2.4.3	Guida di riferimento di Apache Airflow v2.4.3	File dei vincoli di Apache Airflow v2.4.3	Python 3.10
v2.2.2	Guida di riferimento di Apache Airflow v2.2.2	File dei vincoli di Apache Airflow v2.2.2	Python 3.7
v2.0.2	Guida di riferimento di Apache Airflow v2.0.2	File dei vincoli di Apache Airflow v2.0.2	Python 3.7

Per ulteriori informazioni sulla migrazione delle distribuzioni di Apache Airflow autogestite o sulla migrazione di un ambiente Amazon MWAA esistente, incluse le istruzioni per il backup del database di metadati, consulta la Amazon MWAA Migration Guide.

Creazione di un ambiente

La sezione seguente descrive i passaggi per creare un ambiente Amazon MWAA.

Fase uno: specificare i dettagli

Per specificare i dettagli per l'ambiente

1. Apri la console Amazon MWAA.
2. Usa il selettore AWS della regione per selezionare la tua regione.
3. Seleziona Create environment (Crea ambiente).
4. Nella pagina Specificare i dettagli, in Dettagli sull'ambiente:
    a. Digita un nome univoco per il tuo ambiente in Nome.

b. Scegli la versione Apache Airflow nella versione Airflow.

> **Note**
>
> Se non viene specificato alcun valore, per impostazione predefinita viene utilizzata la versione più recente di Airflow. L'ultima versione disponibile è Apache Airflow v2.8.1.

5. Sotto il codice DAG in Amazon S3, specifica quanto segue:

   a. Bucket S3. Scegli Browse S3 e seleziona il tuo bucket Amazon S3 oppure inserisci l'URI Amazon S3.

   b. Cartella DAgS. Scegli Browse S3 e seleziona la `dags` cartella nel tuo bucket Amazon S3 oppure inserisci l'URI Amazon S3.

   c. File dei plugin: opzionale. Scegli Browse S3 e seleziona il `plugins.zip` file nel tuo bucket Amazon S3 oppure inserisci l'URI Amazon S3.

   d. File dei requisiti: facoltativo. Scegli Browse S3 e seleziona il `requirements.txt` file nel tuo bucket Amazon S3 oppure inserisci l'URI Amazon S3.

   e. File di script di avvio: facoltativo, scegli Sfoglia S3 e seleziona il file di script sul tuo bucket Amazon S3 oppure inserisci l'URI di Amazon S3.

6. Seleziona Successivo.

Fase due: configurare le impostazioni avanzate

Configurazione delle impostazioni avanzate

1. Nella pagina Configura impostazioni avanzate, in Rete:

   - Scegli il tuo Amazon VPC.

     Questo passaggio popola due sottoreti private nel tuo Amazon VPC.

2. In Accesso al server Web, seleziona la modalità di accesso Apache Airflow preferita:

   a. Rete privata. Ciò limita l'accesso all'interfaccia utente di Apache Airflow agli utenti all'interno del tuo Amazon VPC a cui è stato concesso l'accesso alla policy IAM per il tuo ambiente. È necessaria l'autorizzazione per creare endpoint Amazon VPC per questa fase.

> **Note**
>
> Scegli l'opzione Rete privata se si accede all'interfaccia utente di Apache Airflow solo all'interno di una rete aziendale e non è necessario accedere a repository pubblici per l'installazione dei requisiti del server Web. Se scegli questa opzione di modalità di accesso, devi creare un meccanismo per accedere al tuo server Web Apache Airflow nel tuo Amazon VPC. Per ulteriori informazioni, consulta Accesso all'endpoint VPC per il server Web Apache Airflow (accesso alla rete privata).

b. Rete pubblica. Ciò consente l'accesso all'interfaccia utente di Apache Airflow tramite Internet agli utenti a cui è concesso l'accesso alla policy IAM per l'ambiente in uso.

3. In Gruppi di sicurezza, scegli il gruppo di sicurezza utilizzato per proteggere il tuo Amazon VPC:

   a. Per impostazione predefinita, Amazon MWAA crea un gruppo di sicurezza nel tuo Amazon VPC con regole specifiche in entrata e in uscita in Crea nuovo gruppo di sicurezza.

   b. Facoltativo. Deseleziona la casella di controllo in Crea nuovo gruppo di sicurezza per selezionare fino a 5 gruppi di sicurezza.

> **Note**
>
> Un gruppo di sicurezza Amazon VPC esistente deve essere configurato con regole specifiche in entrata e in uscita per consentire il traffico di rete. Per ulteriori informazioni, consulta Sicurezza nel tuo VPC su Amazon MWAA.

4. In Classe Environment, scegli una classe di ambiente.

   Ti consigliamo di scegliere la dimensione più piccola necessaria per supportare il tuo carico di lavoro. Puoi cambiare la classe di ambiente in qualsiasi momento.

5. Per Numero massimo di lavoratori, specifica il numero massimo di lavoratori Apache Airflow da eseguire nell'ambiente.

   Per ulteriori informazioni, consulta Esempio di utilizzo ad alte prestazioni.

6. Specificare Numero massimo di server Web e Numero minimo di server Web per configurare il modo in cui Amazon MWAA ridimensiona i server Web Apache Airflow nel tuo ambiente.

   Per ulteriori informazioni sulla scalabilità automatica dei server Web, consulta. the section called “Configurazione della scalabilità automatica del server Web”

7. In Crittografia, scegli un'opzione di crittografia dei dati:

   a. Per impostazione predefinita, Amazon MWAA utilizza una chiave AWS proprietaria per crittografare i dati.

   b. Facoltativo. Scegli Personalizza le impostazioni di crittografia (avanzate) per scegliere una chiave diversa. AWS KMS Se si sceglie di specificare una chiave gestita dal cliente in questo passaggio, è necessario specificare un ID AWS KMS chiave o un ARN. AWS KMS gli alias e le chiavi multiregionali non sono supportati da Amazon MWAA. Se hai specificato una chiave Amazon S3 per la crittografia lato server sul tuo bucket Amazon S3, devi specificare la stessa chiave per il tuo ambiente Amazon MWAA.

   > **Note**
   >
   > È necessario disporre delle autorizzazioni per la chiave per selezionarla sulla console Amazon MWAA. È inoltre necessario concedere le autorizzazioni ad Amazon MWAA per utilizzare la chiave allegando la politica descritta in. Allega una politica chiave

8. Consigliato. In Monitoraggio, scegli una o più categorie di log per la configurazione di registrazione Airflow per inviare i log di Apache Airflow a Logs: CloudWatch

   a. Registri delle attività di Airflow. Scegli il tipo di log delle attività di Apache Airflow da inviare a CloudWatch Logs in Log level.

   b. Registri del server web Airflow. Scegli il tipo di log del server web Apache Airflow da inviare a CloudWatch Logs in Log level.

   c. Registri dello scheduler Airflow. Scegli il tipo di log dello scheduler Apache Airflow da inviare a Logs in Log level. CloudWatch

   d. Registri degli operatori di Airflow. Scegli il tipo di log di lavoro di Apache Airflow da inviare a CloudWatch Logs in Log level.

   e. Registri di elaborazione Airflow DAG. Scegli il tipo di log di elaborazione di Apache Airflow DAG da inviare a Logs in Log level. CloudWatch

9. Facoltativo. Per le opzioni di configurazione Airflow, scegli Aggiungi opzione di configurazione personalizzata.

   Puoi scegliere dall'elenco a discesa suggerito delle opzioni di configurazione di Apache Airflow per la tua versione di Apache Airflow o specificare opzioni di configurazione personalizzate. Ad esempio,:. `core.default_task_retries` `3`

10. Facoltativo. In Tag, scegli Aggiungi nuovo tag per associare i tag al tuo ambiente. Ad esempio,`Environment:Staging`.
11. In Autorizzazioni, scegli un ruolo di esecuzione:

    a. Per impostazione predefinita, Amazon MWAA crea un ruolo di esecuzione in Crea un nuovo ruolo. È necessario disporre dell'autorizzazione per creare ruoli IAM per utilizzare questa opzione.
    b. Facoltativo. Scegli Inserisci ruolo ARN per inserire l'Amazon Resource Name (ARN) di un ruolo di esecuzione esistente.
12. Seleziona Successivo.

Fase tre: revisione e creazione

Per esaminare un riepilogo dell'ambiente

- Esamina il riepilogo dell'ambiente, scegli Crea ambiente.

> **Note**
>
> Occorrono dai venti ai trenta minuti per creare un ambiente.

Fasi successive

- Come creare un bucket Amazon S3Creare are are are are are are are are are are are are are are are are are are are are are are are.

Gestione dell'accesso a un ambiente Amazon MWAA

Amazon Managed Workflows for Apache Airflow deve essere autorizzato a utilizzare altri AWS servizi e risorse utilizzati da un ambiente. È inoltre necessario ottenere l'autorizzazione per accedere a un ambiente Amazon MWAA e all'interfaccia utente di Apache Airflow in AWS Identity and Access Management (IAM). Questa sezione descrive il ruolo di esecuzione utilizzato per concedere l'accesso alle AWS risorse per il tuo ambiente e come aggiungere le autorizzazioni, nonché le autorizzazioni dell' AWS account necessarie per accedere all'ambiente Amazon MWAA e all'interfaccia utente di Apache Airflow.

Argomenti

- Accesso a un ambiente Amazon MWAA
- Ruolo collegato ai servizi per Amazon MWAA
- Ruolo di esecuzione di Amazon MWAA
- Prevenzione del confused deputy tra servizi
- Modalità di accesso Apache Airflow

Accesso a un ambiente Amazon MWAA

Per utilizzare Amazon Managed Workflows for Apache Airflow, devi utilizzare un account e entità IAM con le autorizzazioni necessarie. Questa pagina descrive le politiche di accesso che puoi collegare al tuo team di sviluppo di Apache Airflow e agli utenti di Apache Airflow per il tuo ambiente Amazon Managed Workflows for Apache Airflow.

Ti consigliamo di utilizzare credenziali temporanee e configurare identità federate con gruppi e ruoli per accedere alle tue risorse Amazon MWAA. Come best practice, evita di associare le policy direttamente agli utenti IAM e definisci invece gruppi o ruoli per fornire un accesso temporaneo alle risorse. AWS

Un ruolo IAM è un'identità IAM che puoi creare nel tuo account e che dispone di autorizzazioni specifiche. Un ruolo IAM è simile a quello di un utente IAM in quanto è un' AWS identità con policy di autorizzazioni che determinano ciò in cui l'identità può e non può fare. AWS Tuttavia, invece di essere associato in modo univoco a una persona, un ruolo è destinato a essere assunto da chiunque. Inoltre, un ruolo non ha credenziali a lungo termine standard associate (password o chiavi di accesso). Tuttavia, quando assumi un ruolo, vengono fornite le credenziali di sicurezza provvisorie per la sessione del ruolo.

Per assegnare le autorizzazioni a un'identità federata, si crea un ruolo e si definiscono 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 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 informazioni sui set di autorizzazioni, consulta Set di autorizzazioni nella Guida per l'utente di AWS IAM Identity Center .

Puoi utilizzare un ruolo IAM nel tuo account per concedere altre Account AWS autorizzazioni per accedere alle risorse del tuo account. Per un esempio, consulta Tutorial: Delegate l'accesso attraverso l' Account AWS utilizzo dei ruoli IAM nella IAM User Guide.

Sections

- Come funziona
- Politica di accesso completo alla console: FullConsole AmazonMWAA Access
- Politica completa di accesso all'API e alla console: FullApi AmazonMWAA Access
- Politica di accesso alla console di sola lettura: AmazonMWAA Access ReadOnly
- Politica di accesso all'interfaccia utente di Apache Airflow: AmazonMWAA Access WebServer
- Politica CLI di Apache Airflow: AmazonMWAA Access AirflowCli
- Creazione di una policy JSON
- Esempio di utilizzo per allegare politiche a un gruppo di sviluppatori
- Fasi successive

Come funziona

Le risorse e i servizi utilizzati in un ambiente Amazon MWAA non sono accessibili a tutte le entità AWS Identity and Access Management (IAM). È necessario creare una policy che conceda agli utenti di Apache Airflow l'autorizzazione ad accedere a queste risorse. Ad esempio, è necessario concedere l'accesso al team di sviluppo di Apache Airflow.

Amazon MWAA utilizza queste policy per verificare se un utente dispone delle autorizzazioni necessarie per eseguire un'azione sulla AWS console o tramite le API utilizzate da un ambiente.

Puoi utilizzare le policy JSON descritte in questo argomento per creare una policy per gli utenti di Apache Airflow in IAM e quindi collegarla a un utente, gruppo o ruolo in IAM.

- AmazonMWAA FullConsole Access: utilizza questa politica per concedere l'autorizzazione a configurare un ambiente sulla console Amazon MWAA.
- AmazonMWAA FullApi Access: utilizza questa policy per concedere l'accesso a tutte le API Amazon MWAA utilizzate per gestire un ambiente.
- AmazonMWAA ReadOnly Access: utilizza questa policy per concedere l'accesso alla visualizzazione delle risorse utilizzate da un ambiente sulla console Amazon MWAA.
- AmazonMWAA WebServer Access: utilizza questa politica per concedere l'accesso al server web Apache Airflow.
- AmazonMWAA AirflowCli Access: utilizza questa politica per concedere l'accesso all'esecuzione dei comandi CLI di Apache Airflow.

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.

Politica di accesso completo alla console: FullConsole AmazonMWAA Access

Un utente potrebbe aver bisogno di accedere alla politica `AmazonMWAAFullConsoleAccess` delle autorizzazioni se deve configurare un ambiente sulla console Amazon MWAA.

> **Note**
>
> La tua policy di accesso completo alla console deve includere le autorizzazioni per l'esecuzione. `iam:PassRole` Ciò consente all'utente di passare ruoli collegati al servizio e ruoli di esecuzione ad Amazon MWAA. Amazon MWAA assume ogni ruolo per chiamare altri AWS servizi per tuo conto. L'esempio seguente utilizza la chiave `iam:PassedToService` condition per specificare il servizio Amazon MWAA principal (`airflow.amazonaws.com`) come servizio a cui passare un ruolo.
> Per ulteriori informazioni in merito*`iam:PassRole`*, consulta Concessione a un utente delle autorizzazioni per il trasferimento di un ruolo a un AWS servizio nella IAM User Guide.

Utilizza la seguente policy se desideri creare e gestire i tuoi ambienti Amazon MWAA utilizzando una crittografia Chiave di proprietà di AWSa riposo.

Utilizzando un Chiave di proprietà di AWS

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"airflow:*",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:PassRole"
         ],
         "Resource":"*",
         "Condition":{
            "StringLike":{
               "iam:PassedToService":"airflow.amazonaws.com"
            }
         }
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:ListRoles"
```

```
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:CreatePolicy"
         ],
         "Resource":"arn:aws:iam::YOUR_ACCOUNT_ID:policy/service-role/MWAA-Execution-
Policy*"
      },
            {
         "Effect":"Allow",
         "Action":[
            "iam:AttachRolePolicy",
            "iam:CreateRole"
         ],
         "Resource":"arn:aws:iam::YOUR_ACCOUNT_ID:role/service-role/AmazonMWAA*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:CreateServiceLinkedRole"
         ],
         "Resource":"arn:aws:iam::*:role/aws-service-role/airflow.amazonaws.com/
AWSServiceRoleForAmazonMWAA"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetBucketLocation",
            "s3:ListAllMyBuckets",
            "s3:ListBucket",
            "s3:ListBucketVersions"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:CreateBucket",
            "s3:PutObject",
            "s3:GetEncryptionConfiguration"
         ],
```

```
            "Resource":"arn:aws:s3:::*"
        },
        {
            "Effect":"Allow",
            "Action":[
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:DescribeRouteTables"
            ],
            "Resource":"*"
        },
        {
            "Effect":"Allow",
            "Action":[
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateSecurityGroup"
            ],
            "Resource":"arn:aws:ec2:*:*:security-group/airflow-security-group-*"
        },
        {
            "Effect":"Allow",
            "Action":[
                "kms:ListAliases"
            ],
            "Resource":"*"
        },
        {
            "Effect":"Allow",
            "Action":"ec2:CreateVpcEndpoint",
            "Resource":[
                "arn:aws:ec2:*:*:vpc-endpoint/*",
                "arn:aws:ec2:*:*:vpc/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        },
        {
            "Effect":"Allow",
            "Action":[
                "ec2:CreateNetworkInterface"
            ],
            "Resource":[
                "arn:aws:ec2:*:*:subnet/*",
```

```
            "arn:aws:ec2:*:*:network-interface/*"
         ]
      }
   ]
}
```

Utilizza la seguente policy se desideri creare e gestire i tuoi ambienti Amazon MWAA utilizzando una chiave gestita dal cliente per la crittografia a riposo. Per utilizzare una chiave gestita dal cliente, il responsabile IAM deve disporre dell'autorizzazione ad accedere alle AWS KMS risorse utilizzando la chiave memorizzata nel tuo account.

Utilizzo di una chiave gestita dal cliente

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"airflow:*",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:PassRole"
         ],
         "Resource":"*",
         "Condition":{
            "StringLike":{
               "iam:PassedToService":"airflow.amazonaws.com"
            }
         }
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:ListRoles"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
```

```
            "iam:CreatePolicy"
         ],
         "Resource":"arn:aws:iam::YOUR_ACCOUNT_ID:policy/service-role/MWAA-Execution-
Policy*"
      },
            {
         "Effect":"Allow",
         "Action":[
            "iam:AttachRolePolicy",
            "iam:CreateRole"
         ],
         "Resource":"arn:aws:iam::YOUR_ACCOUNT_ID:role/service-role/AmazonMWAA*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:CreateServiceLinkedRole"
         ],
         "Resource":"arn:aws:iam::*:role/aws-service-role/airflow.amazonaws.com/
AWSServiceRoleForAmazonMWAA"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetBucketLocation",
            "s3:ListAllMyBuckets",
            "s3:ListBucket",
            "s3:ListBucketVersions"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:CreateBucket",
            "s3:PutObject",
            "s3:GetEncryptionConfiguration"
         ],
         "Resource":"arn:aws:s3:::*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeSecurityGroups",
```

```
            "ec2:DescribeSubnets",
            "ec2:DescribeVpcs",
            "ec2:DescribeRouteTables"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:AuthorizeSecurityGroupIngress",
            "ec2:CreateSecurityGroup"
         ],
         "Resource":"arn:aws:ec2:*:*:security-group/airflow-security-group-*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "kms:ListAliases"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "kms:DescribeKey",
            "kms:ListGrants",
            "kms:CreateGrant",
            "kms:RevokeGrant",
            "kms:Decrypt",
            "kms:Encrypt",
            "kms:GenerateDataKey*",
            "kms:ReEncrypt*"
         ],
         "Resource":"arn:aws:kms:*:YOUR_ACCOUNT_ID:key/YOUR_KMS_ID"
      },
      {
         "Effect":"Allow",
         "Action":"ec2:CreateVpcEndpoint",
         "Resource":[
            "arn:aws:ec2:*:*:vpc-endpoint/*",
            "arn:aws:ec2:*:*:vpc/*",
            "arn:aws:ec2:*:*:subnet/*",
            "arn:aws:ec2:*:*:security-group/*"
         ]
```

```
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:CreateNetworkInterface"
         ],
         "Resource":[
            "arn:aws:ec2:*:*:subnet/*",
            "arn:aws:ec2:*:*:network-interface/*"
         ]
      }
   ]
}
```

Politica completa di accesso all'API e alla console: FullApi AmazonMWAA Access

Un utente potrebbe aver bisogno di accedere alla politica `AmazonMWAAFullApiAccess` delle autorizzazioni se ha bisogno di accedere a tutte le API Amazon MWAA utilizzate per gestire un ambiente. Non concede le autorizzazioni per accedere all'interfaccia utente di Apache Airflow.

> **Note**
>
> Una politica di accesso API completa deve includere le autorizzazioni da eseguire. `iam:PassRole` Ciò consente all'utente di passare ruoli collegati al servizio e ruoli di esecuzione ad Amazon MWAA. Amazon MWAA assume ogni ruolo per chiamare altri AWS servizi per tuo conto. L'esempio seguente utilizza la chiave `iam:PassedToService` condition per specificare il servizio Amazon MWAA principal (`airflow.amazonaws.com`) come servizio a cui passare un ruolo.
> Per ulteriori informazioni in merito*`iam:PassRole`*, consulta Concessione a un utente delle autorizzazioni per il trasferimento di un ruolo a un AWS servizio nella IAM User Guide.

Utilizza la seguente policy se desideri creare e gestire i tuoi ambienti Amazon MWAA utilizzando una crittografia Chiave di proprietà di AWS a riposo.

Utilizzando un Chiave di proprietà di AWS

```
{
   "Version":"2012-10-17",
```

```
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"airflow:*",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:PassRole"
         ],
         "Resource":"*",
         "Condition":{
            "StringLike":{
               "iam:PassedToService":"airflow.amazonaws.com"
            }
         }
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:CreateServiceLinkedRole"
         ],
         "Resource":"arn:aws:iam::*:role/aws-service-role/airflow.amazonaws.com/
AWSServiceRoleForAmazonMWAA"
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeSecurityGroups",
            "ec2:DescribeSubnets",
            "ec2:DescribeVpcs",
            "ec2:DescribeRouteTables"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetEncryptionConfiguration"
         ],
         "Resource":"arn:aws:s3:::*"
      },
      {
```

```
         "Effect":"Allow",
         "Action":"ec2:CreateVpcEndpoint",
         "Resource":[
            "arn:aws:ec2:*:*:vpc-endpoint/*",
            "arn:aws:ec2:*:*:vpc/*",
            "arn:aws:ec2:*:*:subnet/*",
            "arn:aws:ec2:*:*:security-group/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:CreateNetworkInterface"
         ],
         "Resource":[
            "arn:aws:ec2:*:*:subnet/*",
            "arn:aws:ec2:*:*:network-interface/*"
         ]
      }
   ]
}
```

Utilizza la seguente policy se desideri creare e gestire i tuoi ambienti Amazon MWAA utilizzando una chiave gestita dal cliente per la crittografia a riposo. Per utilizzare una chiave gestita dal cliente, il responsabile IAM deve disporre dell'autorizzazione ad accedere alle AWS KMS risorse utilizzando la chiave memorizzata nel tuo account.

Utilizzo di una chiave gestita dal cliente

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":"airflow:*",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:PassRole"
         ],
         "Resource":"*",
```

```
         "Condition":{
            "StringLike":{
               "iam:PassedToService":"airflow.amazonaws.com"
            }
         }
      },
      {
         "Effect":"Allow",
         "Action":[
            "iam:CreateServiceLinkedRole"
         ],
         "Resource":"arn:aws:iam::*:role/aws-service-role/airflow.amazonaws.com/
AWSServiceRoleForAmazonMWAA"
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:DescribeSecurityGroups",
            "ec2:DescribeSubnets",
            "ec2:DescribeVpcs",
            "ec2:DescribeRouteTables"
         ],
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "kms:DescribeKey",
            "kms:ListGrants",
            "kms:CreateGrant",
            "kms:RevokeGrant",
            "kms:Decrypt",
            "kms:Encrypt",
            "kms:GenerateDataKey*",
            "kms:ReEncrypt*"
         ],
         "Resource":"arn:aws:kms:*:YOUR_ACCOUNT_ID:key/YOUR_KMS_ID"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetEncryptionConfiguration"
         ],
         "Resource":"arn:aws:s3:::*"
```

```
      },
      {
         "Effect":"Allow",
         "Action":"ec2:CreateVpcEndpoint",
         "Resource":[
            "arn:aws:ec2:*:*:vpc-endpoint/*",
            "arn:aws:ec2:*:*:vpc/*",
            "arn:aws:ec2:*:*:subnet/*",
            "arn:aws:ec2:*:*:security-group/*"
         ]
      },
      {
         "Effect":"Allow",
         "Action":[
            "ec2:CreateNetworkInterface"
         ],
         "Resource":[
            "arn:aws:ec2:*:*:subnet/*",
            "arn:aws:ec2:*:*:network-interface/*"
         ]
      }
   ]
}
```

Politica di accesso alla console di sola lettura: AmazonMWAA Access ReadOnly

Un utente potrebbe aver bisogno di accedere alla politica `AmazonMWAAReadOnlyAccess` delle autorizzazioni se deve visualizzare le risorse utilizzate da un ambiente nella pagina dei dettagli dell'ambiente della console Amazon MWAA. Non consente a un utente di creare nuovi ambienti, modificare ambienti esistenti o visualizzare l'interfaccia utente di Apache Airflow.

```
{
        "Version": "2012-10-17",
        "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "airflow:ListEnvironments",
                "airflow:GetEnvironment",
                "airflow:ListTagsForResource"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

Politica di accesso all'interfaccia utente di Apache Airflow: AmazonMWAA Access WebServer

Un utente potrebbe aver bisogno di accedere alla politica delle `AmazonMWAAWebServerAccess` autorizzazioni se deve accedere all'interfaccia utente di Apache Airflow. Non consente all'utente di visualizzare gli ambienti sulla console Amazon MWAA o di utilizzare le API Amazon MWAA per eseguire alcuna azione. Specificate il `Admin0p`,`User`, `Viewer` o il `Public` ruolo in `{airflow-role}` per personalizzare il livello di accesso per l'utente del token web. Per ulteriori informazioni, consulta Ruoli predefiniti nella guida di riferimento di Apache Airflow.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "airflow:CreateWebLoginToken",
            "Resource": [
                "arn:aws:airflow:{your-region}:YOUR_ACCOUNT_ID:role/{your-environment-name}/{airflow-role}"
            ]
        }
    ]
}
```

> **Note**
>
> Amazon MWAA fornisce l'integrazione IAM con i cinque ruoli RBAC (Role-based access control) predefiniti di Apache Airflow. Per ulteriori informazioni sull'utilizzo dei ruoli personalizzati di Apache Airflow, consulta. the section called “Tutorial: limitazione degli utenti a un sottoinsieme di DAG”

Politica CLI di Apache Airflow: AmazonMWAA Access AirflowCli

Un utente potrebbe aver bisogno di accedere alla politica `AmazonMWAAAirflowCliAccess` delle autorizzazioni se deve eseguire comandi CLI di Apache Airflow (come). `trigger_dag` Non consente all'utente di visualizzare gli ambienti sulla console Amazon MWAA o di utilizzare le API Amazon MWAA per eseguire alcuna azione.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "airflow:CreateCliToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Creazione di una policy JSON

Puoi creare la policy JSON e allegarla al tuo utente, ruolo o gruppo sulla console IAM. I passaggi seguenti descrivono come creare una policy JSON in IAM.

Per creare la policy JSON

1. Apri la pagina Policies sulla console IAM.
2. Scegli Crea policy.
3. Scegliere la scheda JSON.
4. Aggiungi la tua policy JSON.
5. Scegli Verifica policy.
6. Inserisci un valore nel campo di testo per Nome e Descrizione (opzionale).

   Ad esempio, è possibile assegnare un nome alla politica`AmazonMWAAReadOnlyAccess`.
7. Scegli Crea policy.

Esempio di utilizzo per allegare politiche a un gruppo di sviluppatori

Supponiamo che tu stia utilizzando un gruppo in IAM denominato `AirflowDevelopmentGroup` per applicare le autorizzazioni a tutti gli sviluppatori del tuo team di sviluppo di Apache Airflow. Questi utenti devono accedere alle politiche di `AmazonMWAAFullConsoleAccess` accesso e `AmazonMWAAAirflowCliAccess` `AmazonMWAAWebServerAccess` autorizzazione. Questa sezione descrive come creare un gruppo in IAM, creare e allegare queste policy e associare il gruppo a un utente IAM. I passaggi presuppongono che tu stia utilizzando una chiave AWS proprietaria.

Per creare la policy FullConsoleAccess AmazonMWAA

1. Scarica la politica di accesso di AmazonMWAA. FullConsoleAccess
2. Apri la pagina delle politiche sulla console IAM.
3. Scegli Crea policy.
4. Scegliere la scheda JSON.
5. Incolla la policy JSON per`AmazonMWAAFullConsoleAccess`.
6. Sostituisci i seguenti valori:
   a. *`{your-account-id} — L'ID`* del tuo AWS account (ad esempio) `0123456789`
   b. *`{your-kms-id}`* — L'identificatore univoco di una chiave gestita dal cliente, applicabile solo se utilizzi una chiave gestita dal cliente per la crittografia a riposo.
7. Scegli la politica di revisione.
8. `AmazonMWAAFullConsoleAccess`Digita il nome.
9. Scegli Crea policy.

Per creare la politica WebServerAccess AmazonMWAA

1. Scarica la politica di accesso di AmazonMWAA. WebServerAccess
2. Apri la pagina delle politiche sulla console IAM.
3. Scegli Crea policy.
4. Scegliere la scheda JSON.
5. Incolla la policy JSON per`AmazonMWAAWebServerAccess`.
6. Sostituisci i seguenti valori:
   a. *`{your-region}`*: la regione del tuo ambiente Amazon MWAA (ad esempio) `us-east-1`

   b. *{your-account-id} — l'ID* del tuo account (ad esempio) AWS `0123456789`
   c. *{your-environment-name} — il nome* dell'ambiente Amazon MWAA (ad esempio) `MyAirflowEnvironment`
   d. *{airflow-role} — il ruolo predefinito di Apache Airflow Admin*
7. Scegli Verifica policy.
8. `AmazonMWAAWebServerAccess`Digita il nome.
9. Scegli Crea policy.

Per creare la politica AirflowCliAccess AmazonMWAA

1. Scarica la politica di accesso di AmazonMWAA. AirflowCliAccess
2. Apri la pagina delle politiche sulla console IAM.
3. Scegli Crea policy.
4. Scegliere la scheda JSON.
5. Incolla la policy JSON per`AmazonMWAAAirflowCliAccess`.
6. Scegli la politica di revisione.
7. `AmazonMWAAAirflowCliAccess`Digita il nome.
8. Scegli Crea policy.

Per creare il gruppo

1. Apri la pagina Gruppi sulla console IAM.
2. Digita un nome di`AirflowDevelopmentGroup`.
3. Selezionare Next Step (Fase successiva).
4. Digitare `AmazonMWAA` per filtrare i risultati in Filtro.
5. Seleziona le tre politiche che hai creato.
6. Selezionare Next Step (Fase successiva).
7. Selezionare Create Group (Crea gruppo).

Da associare a un utente

1. Apri la pagina Utenti sulla console IAM.

2. Scegli un utente.
3. Scegliere Groups (Gruppi).
4. Scegli Aggiungi utente ai gruppi.
5. Seleziona il AirflowDevelopmentgruppo.
6. Scegliere Add to Groups (Aggiungi a gruppi).

Fasi successive

- Scopri come generare un token per accedere all'interfaccia utente di Apache Airflow. Accesso ad Apache Airflow
- Scopri di più sulla creazione di policy IAM in Creazione di policy IAM.

Ruolo collegato ai servizi per Amazon MWAA

Amazon Managed Workflows for Apache Airflow utilizza ruoli collegati ai servizi AWS Identity and Access Management (IAM). Un ruolo collegato ai servizi è un tipo unico di ruolo IAM collegato direttamente ad Amazon MWAA. I ruoli collegati ai servizi sono predefiniti da Amazon MWAA e includono tutte le autorizzazioni richieste dal servizio per chiamare altri servizi per tuo conto. AWS

Un ruolo collegato al servizio semplifica la configurazione di Amazon MWAA perché non è necessario aggiungere manualmente le autorizzazioni necessarie. Amazon MWAA definisce le autorizzazioni dei suoi ruoli collegati ai servizi e, se non diversamente definito, solo Amazon MWAA può assumerne i ruoli. Le autorizzazioni definite includono la policy di attendibilità e la policy delle autorizzazioni che non può essere allegata a nessun'altra entità IAM.

È possibile eliminare un ruolo collegato ai servizi solo dopo aver eliminato le risorse correlate. In questo modo proteggi le tue risorse Amazon MWAA perché non puoi rimuovere inavvertitamente l'autorizzazione ad accedere alle risorse.

Per informazioni sugli altri servizi che supportano i ruoli collegati ai servizi, consulta Servizi AWS che funzionano con IAM e cerca i servizi che riportano Yes (Sì) nella colonna Service-linked roles (Ruoli collegati ai servizi). Scegli Sì in corrispondenza di un link per visualizzare la documentazione relativa al ruolo collegato al servizio per tale servizio.

Autorizzazioni di ruolo collegate ai servizi per Amazon MWAA

Amazon MWAA utilizza il ruolo collegato al servizio denominato`AWSServiceRoleForAmazonMWAA`: il ruolo collegato al servizio creato nel tuo account concede ad Amazon MWAA l'accesso ai seguenti servizi: AWS

- Amazon CloudWatch Logs (CloudWatch Logs): per creare gruppi di log per i log di Apache Airflow.
- Amazon CloudWatch (CloudWatch): per pubblicare nel tuo account metriche relative al tuo ambiente e ai relativi componenti sottostanti.
- Amazon Elastic Compute Cloud (Amazon EC2) — Per creare le seguenti risorse:
  - Un endpoint Amazon VPC nel tuo VPC per un cluster di database AWS Amazon Aurora PostgreSQL gestito da utilizzare con Apache Airflow Scheduler and Worker.
  - Un endpoint Amazon VPC aggiuntivo per consentire l'accesso di rete al server Web se scegli l'opzione di rete privata per il tuo server Web Apache Airflow.
  - Interfacce di rete elastiche (ENI) nel tuo Amazon VPC per consentire l'accesso di rete alle AWS risorse ospitate nel tuo Amazon VPC.

La seguente politica di fiducia consente al responsabile del servizio di assumere il ruolo collegato al servizio. Il servizio principale per Amazon MWAA è quello `airflow.amazonaws.com` dimostrato dalla policy.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "airflow.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

La politica di autorizzazione dei ruoli denominata `AmazonMWAAServiceRolePolicy` consente ad Amazon MWAA di completare le seguenti azioni sulle risorse specificate:

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:DescribeLogGroups"
            ],
            "Resource": "arn:aws:logs:*:*:log-group:airflow-*:*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachNetworkInterface",
                "ec2:CreateNetworkInterface",
                "ec2:CreateNetworkInterfacePermission",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteNetworkInterfacePermission",
                "ec2:DescribeDhcpOptions",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcEndpoints",
                "ec2:DescribeVpcs",
                "ec2:DetachNetworkInterface"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateVpcEndpoint",
            "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
            "Condition": {
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "AmazonMWAAManaged"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ModifyVpcEndpoint",
                "ec2:DeleteVpcEndpoints"
```

```
            ],
            "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/AmazonMWAAManaged": false
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateVpcEndpoint",
                "ec2:ModifyVpcEndpoint"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:vpc/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:subnet/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateVpcEndpoint"
                },
                "ForAnyValue:StringEquals": {
                    "aws:TagKeys": "AmazonMWAAManaged"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "cloudwatch:PutMetricData",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": [
                        "AWS/MWAA"
                    ]
                }
            }
```

```
        }
    ]
}
```

Per consentire a un'entità IAM (come un utente, un gruppo o un ruolo) di creare, modificare o eliminare un ruolo collegato ai servizi devi configurare le relative autorizzazioni. Per ulteriori informazioni, consulta Autorizzazioni del ruolo collegato ai servizi nella Guida per l'utente di IAM.

Creazione di un ruolo collegato ai servizi per Amazon MWAA

Non hai bisogno di creare manualmente un ruolo collegato ai servizi. Quando crei un nuovo ambiente Amazon MWAA utilizzando l' AWS Management Console, la o l' AWS API AWS CLI, Amazon MWAA crea il ruolo collegato al servizio per te.

Se elimini questo ruolo collegato ai servizi, puoi ricrearlo seguendo lo stesso processo utilizzato per ricreare il ruolo nell'account. Quando crei un altro ambiente, Amazon MWAA crea nuovamente il ruolo collegato al servizio per te.

Modifica di un ruolo collegato al servizio per Amazon MWAA

Amazon MWAA non consente di modificare il ruolo collegato al AWSServiceRoleForAmazonMWAA servizio. Dopo aver creato un ruolo collegato al servizio, non potrai modificarne il nome perché varie entità potrebbero farvi riferimento. È possibile tuttavia modificarne la descrizione utilizzando IAM. Per ulteriori informazioni, consulta Modifica di un ruolo collegato ai servizi nella Guida per l'utente di IAM.

Eliminazione di un ruolo collegato al servizio per Amazon MWAA

Se non è più necessario utilizzare una funzionalità o un servizio che richiede un ruolo collegato al servizio, ti consigliamo di eliminare il ruolo. In questo modo non sarà più presente un'entità non utilizzata che non viene monitorata e gestita attivamente.

Quando elimini un ambiente Amazon MWAA, Amazon MWAA elimina tutte le risorse associate che utilizza come parte del servizio. Tuttavia, devi attendere che Amazon MWAA completi l'eliminazione del tuo ambiente, prima di tentare di eliminare il ruolo collegato al servizio. Se elimini il ruolo collegato al servizio prima che Amazon MWAA elimini l'ambiente, Amazon MWAA potrebbe non essere in grado di eliminare tutte le risorse associate all'ambiente.

Per eliminare manualmente il ruolo collegato ai servizi mediante IAM

Utilizza la console IAM, l'o l'API per eliminare il AWS CLI ruolo collegato al AWS servizio. AWSServiceRoleForAmazonMWAA Per ulteriori informazioni, consulta Eliminazione del ruolo collegato al servizio nella Guida per l'utente di IAM.

Regioni supportate per i ruoli collegati ai servizi Amazon MWAA

Amazon MWAA supporta l'utilizzo di ruoli collegati al servizio in tutte le regioni in cui il servizio è disponibile. Per ulteriori informazioni, consulta Amazon Managed Workflows for Apache Airflow endpoint e quote.

Aggiornamenti alle policy

Modifica	Descrizione	Data
Amazon MWAA aggiorna la sua politica di autorizzazione dei ruoli collegati ai servizi	`AmazonMWAAServiceRolePolicy` — Amazon MWAA aggiorna la politica di autorizzazione per il suo ruolo collegato al servizio per concedere ad Amazon MWAA l'autorizzazione a pubblicare metriche aggiuntive relative alle risorse sottostanti del servizio per gli account dei clienti. Queste nuove metriche sono pubblicate sotto la `AWS/MWAA`	18 novembre 2022
Amazon MWAA ha iniziato a tracciare le modifiche	Amazon MWAA ha iniziato a tracciare le modifiche alla sua politica di autorizzazione dei AWS ruoli collegati ai servizi gestiti.	18 novembre 2022

Ruolo di esecuzione di Amazon MWAA

Un ruolo di esecuzione è un ruolo AWS Identity and Access Management (IAM) con una politica di autorizzazioni che concede ad Amazon Managed Workflows for Apache Airflow l'autorizzazione a richiamare le risorse di altri servizi per tuo conto. AWS Ciò può includere risorse come il bucket Amazon S3, la chiave AWS proprietaria e i log. CloudWatch Gli ambienti Amazon MWAA richiedono un ruolo di esecuzione per ambiente. Questa pagina descrive come utilizzare e configurare il ruolo di esecuzione per il tuo ambiente per consentire ad Amazon MWAA di accedere ad altre AWS risorse utilizzate dal tuo ambiente.

Indice

- Panoramica del ruolo di esecuzione
  - Autorizzazioni allegate per impostazione predefinita
  - Come aggiungere l'autorizzazione all'uso di altri servizi AWS
  - Come associare un nuovo ruolo di esecuzione
- Crea un nuovo ruolo
- Visualizza e aggiorna una politica sul ruolo di esecuzione
  - Allega una policy JSON per utilizzare altri servizi AWS
- Concedi l'accesso al bucket Amazon S3 con blocco di accesso pubblico a livello di account
- Usa le connessioni Apache Airflow
- Esempi di politiche JSON per un ruolo di esecuzione
  - Esempio di politica per una chiave gestita dal cliente
  - Esempio di politica per una chiave di proprietà AWS
- Fasi successive

Panoramica del ruolo di esecuzione

L'autorizzazione per Amazon MWAA a utilizzare altri AWS servizi utilizzati dal tuo ambiente viene ottenuta dal ruolo di esecuzione. Un ruolo di esecuzione di Amazon MWAA richiede l'autorizzazione per i seguenti AWS servizi utilizzati da un ambiente:

- Amazon CloudWatch (CloudWatch): per inviare metriche e log di Apache Airflow.
- Amazon Simple Storage Service (Amazon S3) Simple Storage Service (Amazon S3): per analizzare il codice DAG dell'ambiente e i file di supporto (ad esempio a). `requirements.txt`

- Amazon Simple Queue Service (Amazon SQS): per mettere in coda le attività Apache Airflow del tuo ambiente in una coda Amazon SQS di proprietà di Amazon MWAA.
- AWS Key Management Service (AWS KMS) — per la crittografia dei dati del tuo ambiente (utilizzando una chiave proprietaria o una AWS chiave gestita dal cliente).

> **Note**
>
> Se hai scelto Amazon MWAA di utilizzare una chiave KMS AWS gestita per crittografare i tuoi dati, devi definire le autorizzazioni in una policy allegata al tuo ruolo di esecuzione Amazon MWAA che conceda l'accesso a chiavi KMS arbitrarie archiviate all'esterno del tuo account tramite Amazon SQS. Affinché il ruolo di esecuzione del tuo ambiente possa accedere a chiavi KMS arbitrarie, sono necessarie le due condizioni seguenti:
>
> - Una chiave KMS in un account di terze parti deve consentire questo accesso su più account tramite la relativa politica delle risorse.
> - Il codice DAG deve accedere a una coda Amazon SQS che inizia `airflow-celery-` con l'account di terze parti e utilizza la stessa chiave KMS per la crittografia.
>
> Per mitigare i rischi associati all'accesso alle risorse su più account, ti consigliamo di rivedere il codice inserito nei tuoi DAG per assicurarti che i flussi di lavoro non accedano a code arbitrarie di Amazon SQS al di fuori del tuo account. Inoltre, puoi utilizzare una chiave KMS gestita dal cliente memorizzata nel tuo account per gestire la crittografia su Amazon MWAA. Ciò limita il ruolo di esecuzione del tuo ambiente all'accesso solo alla chiave KMS del tuo account.
>
> Tieni presente che dopo aver scelto un'opzione di crittografia, non puoi modificare la selezione per un ambiente esistente.

Un ruolo di esecuzione richiede inoltre l'autorizzazione per le seguenti azioni IAM:

- `airflow:PublishMetrics`— per consentire ad Amazon MWAA di monitorare lo stato di un ambiente.

Autorizzazioni allegate per impostazione predefinita

Puoi utilizzare le opzioni predefinite sulla console Amazon MWAA per creare un ruolo di esecuzione e una chiave di AWS proprietà, quindi segui i passaggi in questa pagina per aggiungere politiche di autorizzazione al tuo ruolo di esecuzione.

- Quando scegli l'opzione Crea nuovo ruolo sulla console, Amazon MWAA assegna le autorizzazioni minime necessarie a un ambiente al tuo ruolo di esecuzione.
- In alcuni casi, Amazon MWAA assegna le autorizzazioni massime. Ad esempio, consigliamo di scegliere l'opzione sulla console Amazon MWAA per creare un ruolo di esecuzione quando crei un ambiente. Amazon MWAA aggiunge automaticamente le politiche di autorizzazione per tutti i gruppi di CloudWatch Logs utilizzando il modello regex nel ruolo di esecuzione come. `"arn:aws:logs:your-region:your-account-id:log-group:airflow-your-environment-name-*"`

Come aggiungere l'autorizzazione all'uso di altri servizi AWS

Amazon MWAA non può aggiungere o modificare policy di autorizzazione a un ruolo di esecuzione esistente dopo la creazione di un ambiente. È necessario aggiornare il ruolo di esecuzione con politiche di autorizzazione aggiuntive necessarie al proprio ambiente. Ad esempio, se il tuo DAG richiede l'accesso a AWS Glue, Amazon MWAA non può rilevare automaticamente che tali autorizzazioni sono richieste dal tuo ambiente o aggiungere le autorizzazioni al tuo ruolo di esecuzione.

Puoi aggiungere autorizzazioni a un ruolo di esecuzione in due modi:

- Modificando la politica JSON per il ruolo di esecuzione in linea. Puoi utilizzare i documenti di esempio relativi alle policy JSON in questa pagina per aggiungere o sostituire la policy JSON del tuo ruolo di esecuzione sulla console IAM.
- Creando una policy JSON per un AWS servizio e associandola al tuo ruolo di esecuzione. Puoi utilizzare i passaggi di questa pagina per associare un nuovo documento di policy JSON per un AWS servizio al tuo ruolo di esecuzione sulla console IAM.

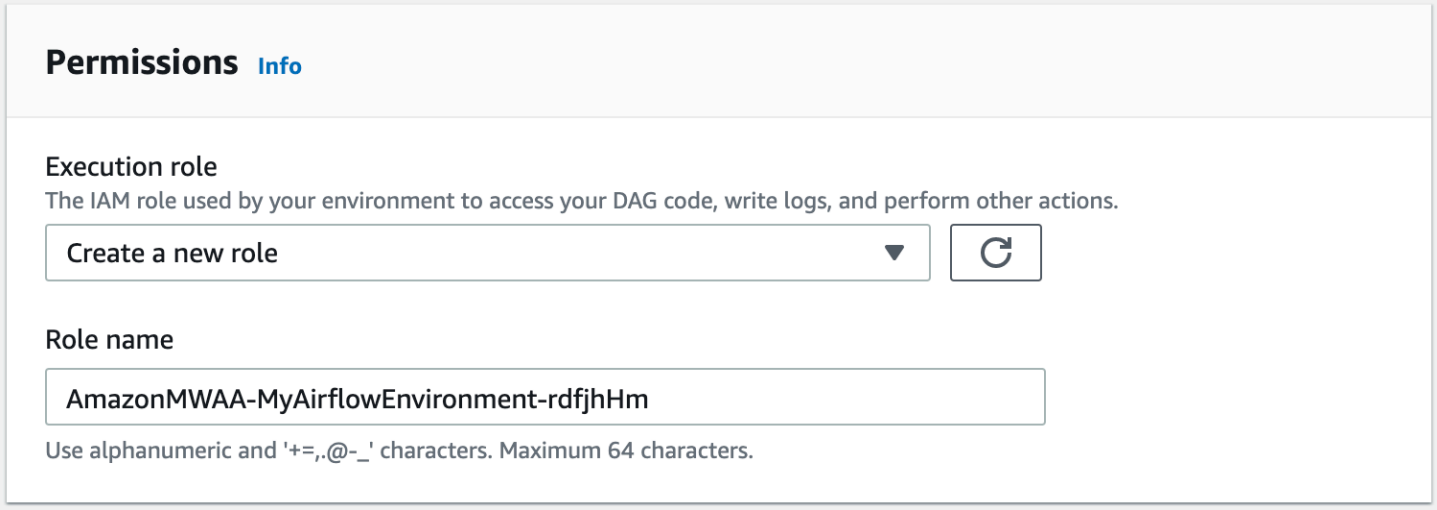
Supponendo che il ruolo di esecuzione sia già associato al tuo ambiente, Amazon MWAA può iniziare a utilizzare immediatamente le politiche di autorizzazione aggiunte. Ciò significa anche che se rimuovi le autorizzazioni necessarie da un ruolo di esecuzione, i tuoi DAG potrebbero fallire.

Come associare un nuovo ruolo di esecuzione

Puoi modificare il ruolo di esecuzione per il tuo ambiente in qualsiasi momento. Se un nuovo ruolo di esecuzione non è già associato al proprio ambiente, utilizzare i passaggi riportati in questa pagina per creare una nuova politica relativa al ruolo di esecuzione e associare il ruolo al proprio ambiente.

Crea un nuovo ruolo

Per impostazione predefinita, Amazon MWAA crea una chiave AWS proprietaria per la crittografia dei dati e un ruolo di esecuzione per tuo conto. Puoi scegliere le opzioni predefinite sulla console Amazon MWAA quando crei un ambiente. L'immagine seguente mostra l'opzione predefinita per creare un ruolo di esecuzione per un ambiente.



Visualizza e aggiorna una politica sul ruolo di esecuzione

Puoi visualizzare il ruolo di esecuzione per il tuo ambiente sulla console Amazon MWAA e aggiornare la policy JSON per il ruolo sulla console IAM.

Per aggiornare una politica del ruolo di esecuzione

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli il ruolo di esecuzione nel riquadro Autorizzazioni per aprire la pagina delle autorizzazioni in IAM.
4. Scegli il nome del ruolo di esecuzione per aprire la politica delle autorizzazioni.
5. Selezionare Edit policy (Modifica policy).
6. Scegli la scheda JSON.
7. Aggiorna la tua policy JSON.
8. Scegli Verifica policy.
9. Seleziona Salvataggio delle modifiche.

Allega una policy JSON per utilizzare altri servizi AWS

Puoi creare una policy JSON per un AWS servizio e collegarla al tuo ruolo di esecuzione. Ad esempio, puoi allegare la seguente politica JSON per concedere l'accesso in sola lettura a tutte le risorse in. AWS Secrets Manager

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "secretsmanager:GetResourcePolicy",
            "secretsmanager:GetSecretValue",
            "secretsmanager:DescribeSecret",
            "secretsmanager:ListSecretVersionIds"
         ],
         "Resource":[
            "*"
         ]
      }
   ]
}
```

Per allegare una policy al tuo ruolo di esecuzione

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli il tuo ruolo di esecuzione nel riquadro Autorizzazioni.
4. Scegli Collega policy.
5. Scegli Crea policy.
6. Scegli JSON.
7. Incolla la policy JSON.
8. Scegli Avanti: Tag, Avanti: Revisione.
9. Inserisci un nome descrittivo (ad esempio`SecretsManagerReadPolicy`) e una descrizione per la politica.
10. Scegli Crea policy.

Concedi l'accesso al bucket Amazon S3 con blocco di accesso pubblico a livello di account

Potresti voler bloccare l'accesso a tutti i bucket del tuo account utilizzando l'operazione `PutPublicAccessBlock`Amazon S3. Quando blocchi l'accesso a tutti i bucket del tuo account, il ruolo di esecuzione dell'ambiente deve includere l'`s3:GetAccountPublicAccessBlock`azione in una politica di autorizzazione.

L'esempio seguente mostra la politica che devi associare al tuo ruolo di esecuzione quando blocchi l'accesso a tutti i bucket Amazon S3 nel tuo account.

```
{
  "Version": "2012-10-17",
  "Statement": [
     {
       "Effect": "Allow",
       "Action": "s3:GetAccountPublicAccessBlock",
       "Resource": "*"
     }
  ]
}
```

Per ulteriori informazioni sulla limitazione dell'accesso ai tuoi bucket Amazon S3, consulta Bloccare l'accesso pubblico allo storage Amazon S3 nella Guida per l'utente di Amazon Simple Storage Service.

Usa le connessioni Apache Airflow

È inoltre possibile creare una connessione Apache Airflow e specificare il ruolo di esecuzione e il relativo ARN nell'oggetto di connessione Apache Airflow. Per ulteriori informazioni, consulta Gestione delle connessioni ad Apache Airflow.

Esempi di politiche JSON per un ruolo di esecuzione

I criteri di autorizzazione di esempio in questa sezione mostrano due criteri che è possibile utilizzare per sostituire i criteri di autorizzazione utilizzati per il ruolo di esecuzione esistente o per creare un nuovo ruolo di esecuzione da utilizzare per l'ambiente. Queste policy contengono segnaposti Resource ARN per i gruppi di log Apache Airflow, un bucket Amazon S3 e un ambiente Amazon MWAA.

Ti consigliamo di copiare la policy di esempio, sostituire gli ARN o i placeholder di esempio, quindi utilizzare la policy JSON per creare o aggiornare un ruolo di esecuzione. Ad esempio, sostituendo con. `{your-region}` `us-east-1`

Esempio di politica per una chiave gestita dal cliente

L'esempio seguente mostra una politica del ruolo di esecuzione che è possibile utilizzare per una chiave gestita dal cliente.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:GetBucket*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::{your-s3-bucket-name}",
                "arn:aws:s3:::{your-s3-bucket-name}/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents",
                "logs:GetLogEvents",
                "logs:GetLogRecord",
                "logs:GetLogGroupFields",
                "logs:GetQueryResults"
            ],
            "Resource": [
                "arn:aws:logs:{your-region}:{your-account-id}:log-group:airflow-{your-
environment-name}-*"
```

```
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "cloudwatch:PutMetricData",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sqs:ChangeMessageVisibility",
                "sqs:DeleteMessage",
                "sqs:GetQueueAttributes",
                "sqs:GetQueueUrl",
                "sqs:ReceiveMessage",
                "sqs:SendMessage"
            ],
            "Resource": "arn:aws:sqs:{your-region}:*:airflow-celery-*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey*",
                "kms:Encrypt"
```

```
            ],
            "Resource": "arn:aws:kms:{your-region}:{your-account-id}:key/{your-kms-cmk-
id}",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": [
                        "sqs.{your-region}.amazonaws.com",
                        "s3.{your-region}.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Successivamente, devi consentire ad Amazon MWAA di assumere questo ruolo per eseguire azioni per tuo conto. Ciò può essere fatto aggiungendo "`airflow.amazonaws.com`" i responsabili del "`airflow-env.amazonaws.com`" servizio all'elenco delle entità attendibili per questo ruolo di esecuzione utilizzando la console IAM o inserendo questi principali di servizio nel documento sulla politica di assume role per questo ruolo di esecuzione tramite il comando IAM create-role utilizzando il. AWS CLI Di seguito è riportato un esempio di documento sulla politica relativa all'assunzione del ruolo:

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Principal": {
            "Service": ["airflow.amazonaws.com","airflow-env.amazonaws.com"]
        },
        "Action": "sts:AssumeRole"
      }
   ]
}
```

Quindi allega la seguente politica JSON alla tua chiave gestita dal cliente. Questo criterio utilizza il prefisso `kms:EncryptionContext`condition key per consentire l'accesso al gruppo di log di Apache Airflow in Logs. CloudWatch

```
{
```

```
    "Sid": "Allow logs access",
    "Effect": "Allow",
    "Principal": {
        "Service": "logs.{your-region}.amazonaws.com"
    },
    "Action": [
        "kms:Encrypt*",
        "kms:Decrypt*",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:Describe*"
    ],
    "Resource": "*",
    "Condition": {
        "ArnLike": {
            "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:{your-region}:{your-
account-id}:*"
        }
    }
}
```

Esempio di politica per una chiave di proprietà AWS

L'esempio seguente mostra una politica del ruolo di esecuzione che è possibile utilizzare per una chiave AWS di proprietà.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "airflow:PublishMetrics",
            "Resource": "arn:aws:airflow:{your-region}:{your-account-id}:environment/
{your-environment-name}"
        },
        {
            "Effect": "Deny",
            "Action": "s3:ListAllMyBuckets",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "s3:GetObject*",
                "s3:GetBucket*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::{your-s3-bucket-name}",
                "arn:aws:s3:::{your-s3-bucket-name}/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
                "logs:CreateLogGroup",
                "logs:PutLogEvents",
                "logs:GetLogEvents",
                "logs:GetLogRecord",
                "logs:GetLogGroupFields",
                "logs:GetQueryResults"
            ],
            "Resource": [
                "arn:aws:logs:{your-region}:{your-account-id}:log-group:airflow-{your-
environment-name}-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetAccountPublicAccessBlock"
            ],
            "Resource": [
                "*"
            ]
        },
        {
```

```
            "Effect": "Allow",
            "Action": "cloudwatch:PutMetricData",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sqs:ChangeMessageVisibility",
                "sqs:DeleteMessage",
                "sqs:GetQueueAttributes",
                "sqs:GetQueueUrl",
                "sqs:ReceiveMessage",
                "sqs:SendMessage"
            ],
            "Resource": "arn:aws:sqs:{your-region}:*:airflow-celery-*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey*",
                "kms:Encrypt"
            ],
            "NotResource": "arn:aws:kms:*:{your-account-id}:key/*",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": [
                        "sqs.{your-region}.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

Fasi successive

- Scopri le autorizzazioni necessarie a te e ai tuoi utenti di Apache Airflow per accedere al tuo ambiente. Accesso a un ambiente Amazon MWAA
- Ulteriori informazioni su Utilizzo di chiavi gestite dal cliente per la crittografia.
- Esplora altri esempi di policy gestite dai clienti.

Prevenzione del confused deputy tra servizi

Con "confused deputy" si intende un problema di sicurezza in cui un'entità che non dispone dell'autorizzazione per eseguire una certa operazione può costringere un'entità con più privilegi a eseguire tale operazione. Nel AWS, l'impersonificazione tra servizi può portare al confuso problema del vicesceriffo. La rappresentazione tra servizi può verificarsi quando un servizio (il servizio chiamante) effettua una chiamata a un altro servizio (il servizio chiamato). Il servizio chiamante può essere manipolato per utilizzare le proprie autorizzazioni e agire sulle risorse di un altro cliente, a cui normalmente non avrebbe accesso. Per evitare ciò, AWS fornisce strumenti per poterti a proteggere i tuoi dati per tutti i servizi con entità di servizio a cui è stato concesso l'accesso alle risorse del tuo account.

Ti consigliamo di utilizzare le chiavi di contesto `aws:SourceArn`e `aws:SourceAccount`global condition nel ruolo di esecuzione del tuo ambiente per limitare le autorizzazioni che Amazon MWAA fornisce a un altro servizio per accedere alla risorsa. Utilizza `aws:SourceArn` se desideri consentire l'associazione di una sola risorsa all'accesso tra servizi. Utilizza `aws:SourceAccount` se desideri consentire l'associazione di qualsiasi risorsa in tale account all'uso tra servizi.

Il modo più efficace per proteggersi dal problema "confused deputy" è quello di usare la chiave di contesto della condizione globale `aws:SourceArn`con l'ARN completo della risorsa. Se non conosci l'ARN completo della risorsa o scegli più risorse, utilizza la chiave di contesto della condizione globale `aws:SourceArn` con caratteri jolly (`*`) per le parti sconosciute dell'ARN. Ad esempio, `arn:aws:airflow:*:`*`123456789012`*`:environment/*`.

Il valore di `aws:SourceArn` deve essere l'ARN dell'ambiente Amazon MWAA, per il quale stai creando un ruolo di esecuzione.

L'esempio seguente mostra come è possibile utilizzare le chiavi di contesto `aws:SourceArn` e `aws:SourceAccount` global condition nella policy di fiducia dei ruoli di esecuzione del proprio ambiente per evitare il confuso problema del vicedirettore. È possibile utilizzare la seguente politica di fiducia quando si crea un nuovo ruolo di esecuzione.

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Principal": {
            "Service": ["airflow.amazonaws.com","airflow-env.amazonaws.com"]
```

```
        },
        "Action": "sts:AssumeRole",
        "Condition":{
            "ArnLike":{
               "aws:SourceArn":"arn:aws:airflow:your-
region:123456789012:environment/your-environment-name"
            },
            "StringEquals":{
               "aws:SourceAccount":"123456789012"
            }
         }
      }
   ]
}
```

Modalità di accesso Apache Airflow

La console Amazon Managed Workflows for Apache Airflow contiene opzioni integrate per configurare il routing privato o pubblico verso il server Web Apache Airflow nel tuo ambiente. Questa guida descrive le modalità di accesso disponibili per il server Web Apache Airflow sul tuo ambiente Amazon Managed Workflows for Apache Airflow e le risorse aggiuntive che dovrai configurare nel tuo Amazon VPC se scegli l'opzione di rete privata.

Indice

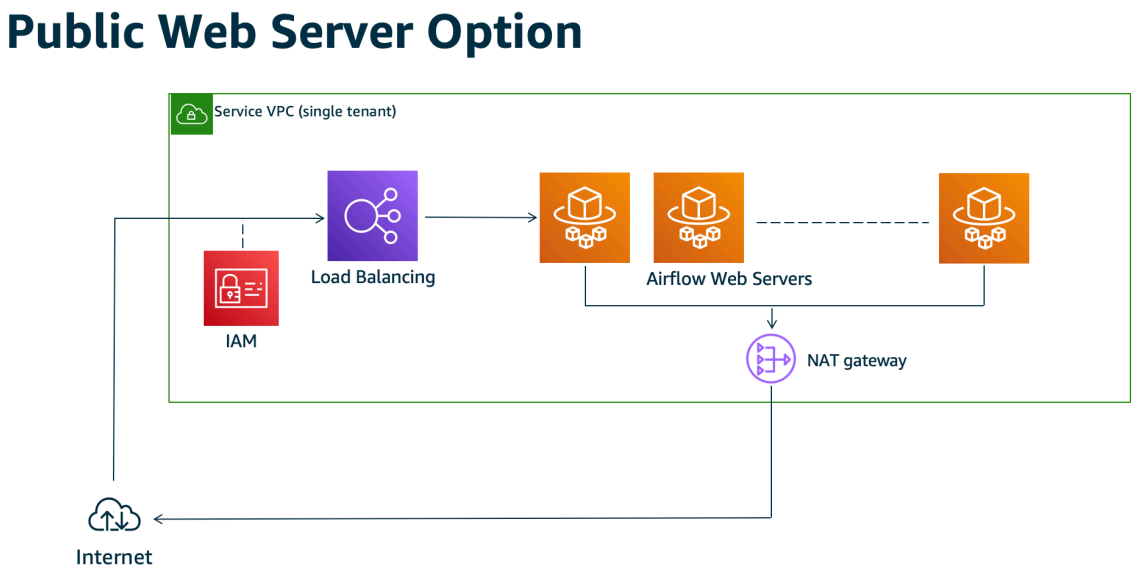
- Modalità di accesso Apache Airflow
  - Rete pubblica
  - Rete privata
- Panoramica delle modalità di accesso
  - Modalità di accesso alla rete pubblica
  - Modalità di accesso alla rete privata
- Configurazione per le modalità di accesso privato e pubblico
  - Configurazione per rete pubblica
  - Configurazione per rete privata
- Accesso all'endpoint VPC per il server Web Apache Airflow (accesso alla rete privata)

Modalità di accesso Apache Airflow

Puoi scegliere un routing privato o pubblico per il tuo server Web Apache Airflow. Per abilitare il routing privato, scegli Rete privata. Ciò limita l'accesso degli utenti a un server Web Apache Airflow all'interno di un Amazon VPC. Per abilitare il routing pubblico, scegli Rete pubblica. Ciò consente agli utenti di accedere al server Web Apache Airflow tramite Internet.

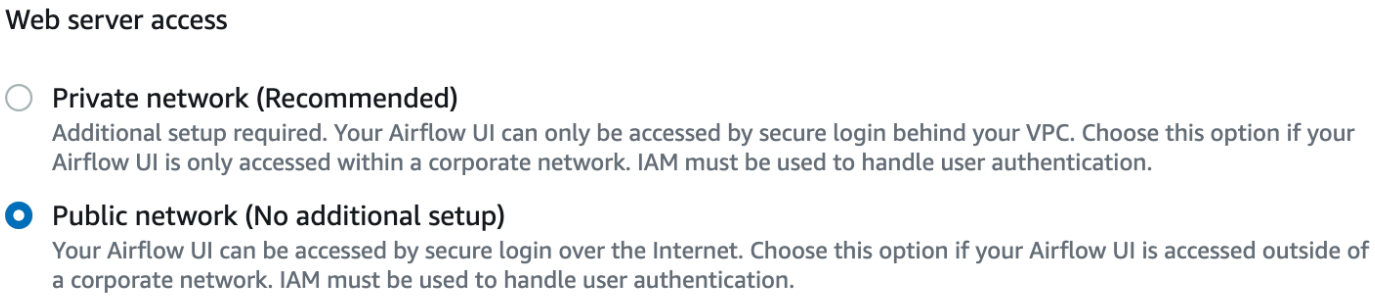
Rete pubblica

Il seguente diagramma architettonico mostra un ambiente Amazon MWAA con un server Web pubblico.



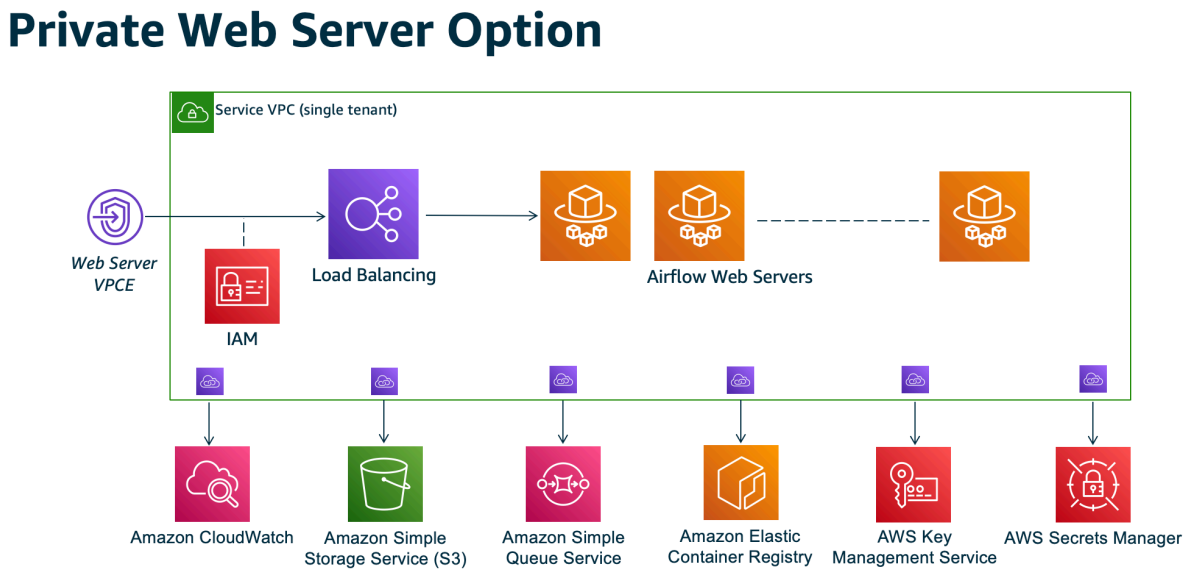
La modalità di accesso alla rete pubblica consente l'accesso all'interfaccia utente di Apache Airflow tramite Internet agli utenti a cui è stato concesso l'accesso alla policy IAM per il tuo ambiente.

L'immagine seguente mostra dove trovare l'opzione Rete pubblica sulla console Amazon MWAA.



Rete privata

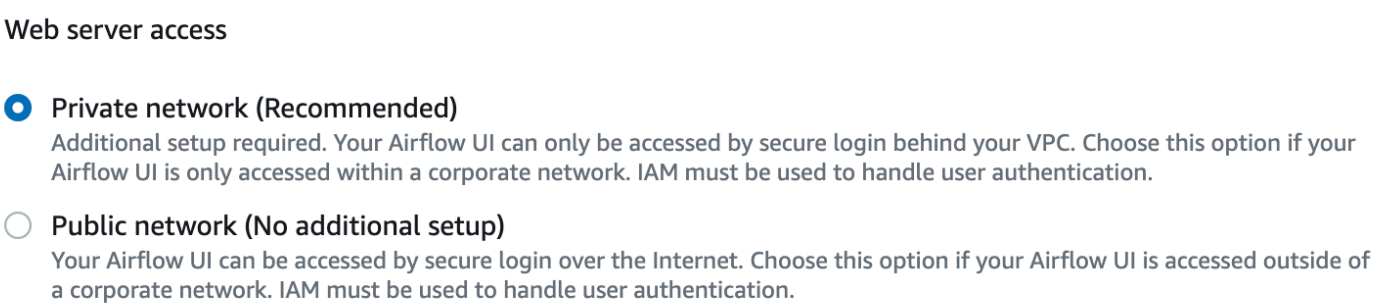
Il seguente diagramma architettonico mostra un ambiente Amazon MWAA con un server Web privato.



La modalità di accesso alla rete privata limita l'accesso all'interfaccia utente di Apache Airflow agli utenti del tuo Amazon VPC a cui è stato concesso l'accesso alla policy IAM per il tuo ambiente.

Quando crei un ambiente con accesso privato al server web, devi impacchettare tutte le tue dipendenze in un archivio Python wheel `.whl` (), quindi fare riferimento a `.whl` nel tuo. `requirements.txt` Per istruzioni su come impacchettare e installare le dipendenze usando wheel, vedi Gestire le dipendenze usando Python wheel.

L'immagine seguente mostra dove trovare l'opzione Rete privata sulla console Amazon MWAA.



Panoramica delle modalità di accesso

Questa sezione descrive gli endpoint VPC (AWS PrivateLink) creati nel tuo Amazon VPC quando scegli la modalità di accesso alla rete pubblica o alla rete privata.

Modalità di accesso alla rete pubblica

Se si sceglie la modalità di accesso alla rete pubblica per il server Web Apache Airflow, il traffico di rete viene instradato pubblicamente su Internet.

- Amazon MWAA crea un endpoint di interfaccia VPC per il tuo database di metadati Amazon Aurora PostgreSQL. L'endpoint viene creato nelle zone di disponibilità mappate alle sottoreti private ed è indipendente dagli altri account. AWS
- Amazon MWAA associa quindi un indirizzo IP dalle sottoreti private agli endpoint dell'interfaccia. Questo è progettato per supportare la best practice di associare un singolo IP da ogni zona di disponibilità di Amazon VPC.

Modalità di accesso alla rete privata

Se hai scelto la modalità di accesso alla rete privata per il tuo server Web Apache Airflow, il traffico di rete viene instradato privatamente all'interno del tuo Amazon VPC.

- Amazon MWAA crea un endpoint di interfaccia VPC per il server Web Apache Airflow e un endpoint di interfaccia per il database di metadati PostgreSQL di Amazon Aurora. Gli endpoint vengono creati nelle zone di disponibilità mappate alle sottoreti private e sono indipendenti dagli altri account. AWS
- Amazon MWAA associa quindi un indirizzo IP dalle sottoreti private agli endpoint dell'interfaccia. Questo è progettato per supportare la best practice di associare un singolo IP da ogni zona di disponibilità di Amazon VPC.

Per ulteriori informazioni, consulta the section called “Esempi di casi d'uso per una modalità di accesso Amazon VPC e Apache Airflow”.

Configurazione per le modalità di accesso privato e pubblico

La sezione seguente descrive la configurazione e le configurazioni aggiuntive necessarie in base alla modalità di accesso Apache Airflow scelta per il proprio ambiente.

Configurazione per rete pubblica

Se scegli l'opzione Rete pubblica per il server Web Apache Airflow, puoi iniziare a utilizzare l'interfaccia utente di Apache Airflow dopo aver creato l'ambiente.

È necessario eseguire le seguenti operazioni per configurare l'accesso per gli utenti e l'autorizzazione dell'ambiente a utilizzare altri servizi. AWS

1. Aggiungi autorizzazioni. Amazon MWAA necessita dell'autorizzazione per utilizzare altri AWS servizi. Quando crei un ambiente, Amazon MWAA crea un ruolo collegato al servizio che gli consente di utilizzare determinate azioni IAM per Amazon Elastic Container Registry (Amazon ECR), Logs e Amazon EC2 CloudWatch .

   Puoi aggiungere l'autorizzazione per utilizzare azioni aggiuntive per questi servizi o per utilizzare altri AWS servizi aggiungendo autorizzazioni al tuo ruolo di esecuzione. Per ulteriori informazioni, consulta Ruolo di esecuzione di Amazon MWAA.
2. Crea politiche per gli utenti. Potrebbe essere necessario creare più policy IAM per consentire agli utenti di configurare l'accesso al proprio ambiente e all'interfaccia utente di Apache Airflow. Per ulteriori informazioni, consulta Accesso a un ambiente Amazon MWAA.

Configurazione per rete privata

Se scegli l'opzione Rete privata per il tuo server Web Apache Airflow, dovrai configurare l'accesso per i tuoi utenti, l'autorizzazione per il tuo ambiente a utilizzare altri AWS servizi e creare un meccanismo per accedere alle risorse del tuo Amazon VPC dal tuo computer.

1. Aggiungi autorizzazioni. Amazon MWAA necessita dell'autorizzazione per utilizzare altri AWS servizi. Quando crei un ambiente, Amazon MWAA crea un ruolo collegato al servizio che gli consente di utilizzare determinate azioni IAM per Amazon Elastic Container Registry (Amazon ECR), Logs e Amazon EC2 CloudWatch .

   Puoi aggiungere l'autorizzazione per utilizzare azioni aggiuntive per questi servizi o per utilizzare altri AWS servizi aggiungendo autorizzazioni al tuo ruolo di esecuzione. Per ulteriori informazioni, consulta Ruolo di esecuzione di Amazon MWAA.
2. Crea politiche per gli utenti. Potrebbe essere necessario creare più policy IAM per consentire agli utenti di configurare l'accesso al proprio ambiente e all'interfaccia utente di Apache Airflow. Per ulteriori informazioni, consulta Accesso a un ambiente Amazon MWAA.

3. Abilita l'accesso alla rete. Dovrai creare un meccanismo nel tuo Amazon VPC per connetterti all'endpoint VPC (AWS PrivateLink) per il tuo server Web Apache Airflow. Ad esempio, creando un tunnel VPN dal tuo computer utilizzando un. AWS Client VPN

Accesso all'endpoint VPC per il server Web Apache Airflow (accesso alla rete privata)

Se hai scelto l'opzione Rete privata, dovrai creare un meccanismo nel tuo Amazon VPC per accedere all'endpoint VPC (AWS PrivateLink) per il tuo server Web Apache Airflow. Ti consigliamo di utilizzare lo stesso Amazon VPC, lo stesso gruppo di sicurezza VPC e le stesse sottoreti private dell'ambiente Amazon MWAA per queste risorse.

Per ulteriori informazioni, consulta Gestione dell'accesso per gli endpoint VPC.

Accesso ad Apache Airflow

Amazon MWAA ti consente di accedere al tuo ambiente Apache Airflow utilizzando diversi metodi: la console dell'interfaccia utente (UI) Apache Airflow, la CLI di Apache Airflow e l'API REST di Apache Airflow. Puoi utilizzare la console Amazon MWAA per visualizzare e richiamare un DAG nell'interfaccia utente di Apache Airflow oppure utilizzare le API Amazon MWAA per ottenere un token e richiamare un DAG. Questa sezione descrive le autorizzazioni necessarie per accedere all'interfaccia utente di Apache Airflow, come generare un token per effettuare chiamate all'API Amazon MWAA direttamente nella shell dei comandi e i comandi supportati nella CLI di Apache Airflow.

Argomenti

- Prerequisiti
- Apri l'interfaccia utente Airflow
- Accesso ad Apache Airflow
- Creare un token di accesso al server web Apache Airflow
- Creazione di un token CLI di Apache Airflow
- Utilizzo dell'API REST di Apache Airflow
- Riferimento ai comandi CLI di Apache Airflow

Prerequisiti

La sezione seguente descrive i passaggi preliminari necessari per utilizzare i comandi e gli script di questa sezione.

Accesso

- AWS accesso tramite account AWS Identity and Access Management (IAM) alla policy di autorizzazione di Amazon MWAA in. Politica di accesso all'interfaccia utente di Apache Airflow: AmazonMWAA Access WebServer
- AWS accesso tramite account AWS Identity and Access Management (IAM) alla policy di autorizzazione di Amazon MWAA. Politica completa di accesso all'API e alla console: FullApi AmazonMWAA Access

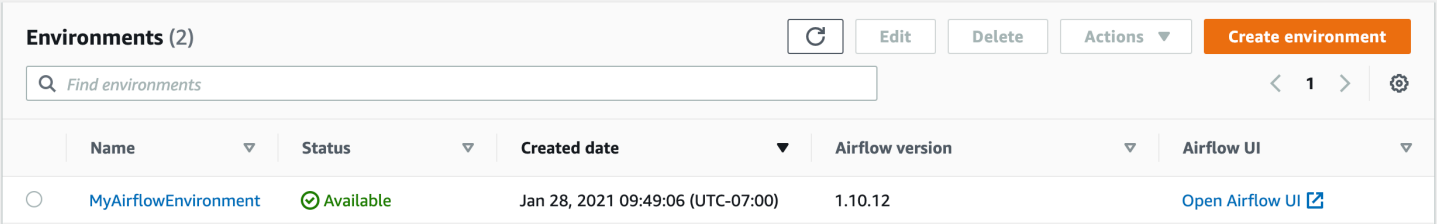
AWS CLI

Il AWS Command Line Interface (AWS CLI) è uno strumento open source che consente di interagire con i AWS servizi utilizzando i comandi nella shell della riga di comando. Per completare la procedura descritta in questa pagina, è necessario quanto segue:

- AWS CLI — Installa la versione 2.
- AWS CLI — Configurazione rapida con `aws configure`.

Apri l'interfaccia utente Airflow

L'immagine seguente mostra il collegamento all'interfaccia utente di Apache Airflow sulla console Amazon MWAA.



Accesso ad Apache Airflow

Hai bisogno Politica di accesso all'interfaccia utente di Apache Airflow: AmazonMWAA Access WebServer delle autorizzazioni per il tuo AWS account in AWS Identity and Access Management (IAM) per visualizzare l'interfaccia utente di Apache Airflow.

Per accedere all'interfaccia utente di Apache Airflow

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Open Airflow UI.

Creare un token di accesso al server web Apache Airflow

Puoi utilizzare i comandi in questa pagina per creare un token di accesso al server Web. Un token di accesso ti consente di accedere al tuo ambiente Amazon MWAA. Ad esempio, puoi ottenere un token e quindi distribuire i DAG in modo programmatico utilizzando le API di Amazon MWAA. La sezione

seguente include i passaggi per creare un token di accesso web Apache Airflow utilizzando uno script bash, una richiesta API POST o uno script Python. AWS CLI Il token restituito nella risposta è valido per 60 secondi.

Indice

- Prerequisiti
  - Accesso
  - AWS CLI
- Utilizzando il AWS CLI
- Utilizzando uno script bash
- Utilizzo di una richiesta API POST
- Usare uno script Python
- Fasi successive

Prerequisiti

La sezione seguente descrive i passaggi preliminari necessari per utilizzare i comandi e gli script in questa pagina.

Accesso

- AWS accesso tramite account AWS Identity and Access Management (IAM) alla policy di autorizzazione di Amazon MWAA in. Politica di accesso all'interfaccia utente di Apache Airflow: AmazonMWAA Access WebServer
- AWS accesso tramite account AWS Identity and Access Management (IAM) alla policy di autorizzazione di Amazon MWAA. Politica completa di accesso all'API e alla console: FullApi AmazonMWAA Access

AWS CLI

Il AWS Command Line Interface (AWS CLI) è uno strumento open source che consente di interagire con i AWS servizi utilizzando i comandi nella shell della riga di comando. Per completare la procedura descritta in questa pagina, è necessario quanto segue:

- AWS CLI — Installa la versione 2.
- AWS CLI — Configurazione rapida con `aws configure`.

Utilizzando il AWS CLI

L'esempio seguente utilizza il create-web-login-tokencomando in AWS CLI per creare un token di accesso web Apache Airflow.

```
aws mwaa create-web-login-token --name YOUR_ENVIRONMENT_NAME
```

Utilizzando uno script bash

L'esempio seguente utilizza uno script bash per chiamare il create-web-login-tokencomando in per creare un token AWS CLI di accesso web Apache Airflow.

1. Copia il contenuto del seguente esempio di codice e salvalo localmente come. `get-web-token.sh`

   ```
   #!/bin/bash
   HOST=YOUR_HOST_NAME
   YOUR_URL=https://$HOST/aws_mwaa/aws-console-sso?login=true#
   WEB_TOKEN=$(aws mwaa create-web-login-token --name YOUR_ENVIRONMENT_NAME --query
    WebToken --output text)
   echo $YOUR_URL$WEB_TOKEN
   ```

2. Sostituisci e con i segnaposto in *rosso.* YOUR_HOST_NAME YOUR_ENVIRONMENT_NAME Ad esempio, un nome host per una rete pubblica può avere il seguente aspetto (senza https://):

   ```
   123456a0-0101-2020-9e11-1b159eec9000.c2.us-east-1.airflow.amazonaws.com
   ```

3. (opzionale) Gli utenti macOS e Linux potrebbero dover eseguire il comando seguente per assicurarsi che lo script sia eseguibile.

   ```
   chmod +x get-web-token.sh
   ```

4. Esegui lo script seguente per ottenere un token di accesso Web.

   ```
   ./get-web-token.sh
   ```

5. Dovresti vedere quanto segue nel prompt dei comandi:

   ```
   https://123456a0-0101-2020-9e11-1b159eec9000.c2.us-east-1.airflow.amazonaws.com/
   aws_mwaa/aws-console-sso?login=true#{your-web-login-token}
   ```

Utilizzo di una richiesta API POST

L'esempio seguente utilizza una richiesta API POST per creare un token di accesso web Apache Airflow.

1. Copia il seguente URL e incollalo nel campo URL del tuo client API REST.

```
https://YOUR_HOST_NAME/aws_mwaa/aws-console-sso?login=true#WebToken
```

2. *Sostituisci i segnaposto in rosso con.* YOUR_HOST_NAME Ad esempio, un nome host per una rete pubblica può avere il seguente aspetto (senza https://):

```
123456a0-0101-2020-9e11-1b159eec9000.c2.us-east-1.airflow.amazonaws.com
```

3. Copia il seguente codice JSON e incollalo nel campo body del tuo client API REST.

```
{
  "name": "YOUR_ENVIRONMENT_NAME"
}
```

4. *Sostituisci il segnaposto in rosso con.* YOUR_ENVIRONMENT_NAME
5. Aggiungi coppie chiave-valore nel campo di autorizzazione. Ad esempio, se utilizzi Postman, scegli AWS Signature, quindi inserisci:
   - AWS_ACCESS_KEY_ID in AccessKey
   - AWS_SECRET_ACCESS_KEY in SecretKey
6. Dovrebbe essere visualizzata la seguente risposta:

```
{
  "webToken": "<Short-lived token generated for enabling access to the Apache
 Airflow Webserver UI>",
  "webServerHostname": "<Hostname for the WebServer of the environment>"
}
```

Usare uno script Python

L'esempio seguente utilizza il metodo boto3 create_web_login_token in uno script Python per creare un token di accesso web Apache Airflow. Puoi eseguire questo script al di fuori di Amazon MWAA.

L'unica cosa che devi fare è installare la libreria boto3. Potresti voler creare un ambiente virtuale per installare la libreria. Si presuppone che tu abbia configurato le credenziali di AWS autenticazione per il tuo account.

1. Copia il contenuto del seguente esempio di codice e salvalo localmente come. `create-web-login-token.py`

```
import boto3
mwaa = boto3.client('mwaa')
response = mwaa.create_web_login_token(
    Name="YOUR_ENVIRONMENT_NAME"
)
webServerHostName = response["WebServerHostname"]
webToken = response["WebToken"]
airflowUIUrl = 'https://{0}/aws_mwaa/aws-console-sso?
login=true#{1}'.format(webServerHostName, webToken)
print("Here is your Airflow UI URL: ")
print(airflowUIUrl)
```

2. *Sostituisci il segnaposto in rosso con.* YOUR_ENVIRONMENT_NAME
3. Esegui lo script seguente per ottenere un token di accesso web.

```
python3 create-web-login-token.py
```

Fasi successive

- Esplora il funzionamento dell'API Amazon MWAA utilizzato per creare un token di accesso Web all'indirizzo. CreateWebLoginToken

Creazione di un token CLI di Apache Airflow

Puoi utilizzare i comandi di questa pagina per generare un token CLI e quindi effettuare chiamate all'API Amazon Managed Workflows for Apache Airflow direttamente nella shell dei comandi. Ad esempio, puoi ottenere un token, quindi distribuire i DAG a livello di codice utilizzando le API Amazon MWAA. La sezione seguente include i passaggi per creare un token CLI di Apache Airflow utilizzando loAWS CLI script curl, uno script Python o uno script bash. Il token restituito nella risposta è valido per 60 secondi.

> **Note**
>
> IlAWS CLI token è inteso come sostituto delle azioni shell sincrone, non dei comandi API asincroni. Pertanto, la concorrenza disponibile è limitata. Per garantire che il server Web continui a rispondere agli utenti, si consiglia di non aprire una nuovaAWS CLI richiesta fino a quando quella precedente non viene completata correttamente.

Indice

- Prerequisiti
  - Accesso
  - AWS CLI
- Utilizzo di AWS CLI
- Usare uno script curl
- Usare uno script bash
- Usare uno script Python
- Fasi successive

Prerequisiti

La sezione seguente descrive i passaggi preliminari necessari per utilizzare i comandi e gli script di questa pagina.

Accesso

- AWSaccesso all'account inAWS Identity and Access Management (IAM) alla politica delle autorizzazioni Amazon MWAA inPolitica di accesso all'interfaccia utente di Apache Airflow: AmazonMWAA Access WebServer.
- AWSaccesso all'account inAWS Identity and Access Management (IAM) alla politica di autorizzazione Amazon MWAAPolitica completa di accesso all'API e alla console: FullApi AmazonMWAA Access.

AWS CLI

AWS Command Line Interface (AWS CLI) è uno strumento open source che consente di interagire con i servizi AWS utilizzando i comandi nella shell a riga di comando. Per completare le fasi di questa pagina, è necessario quanto segue:

- AWS CLI— Installare la versione 2.
- AWS CLI— Configurazione rapida con`aws configure`.

Utilizzo di AWS CLI

L'esempio seguente utilizza il create-cli-tokencomando inAWS CLI per creare un token CLI di Apache Airflow.

```
aws mwaa create-cli-token --name YOUR_ENVIRONMENT_NAME
```

Usare uno script curl

L'esempio seguente utilizza uno script curl per chiamare il create-web-login-tokencomando inAWS CLI per richiamare l'interfaccia a riga di comando di Apache Airflow tramite un endpoint sul server web Apache Airflow.

Apache Airflow v2

1. Copia l'istruzione curl dal tuo file di testo e incollala nella shell dei comandi.

   > **Note**
   >
   > Dopo averlo copiato negli appunti, potrebbe essere necessario utilizzare Modifica > Incolla dal menu della shell.

   ```
   CLI_JSON=$(aws mwaa --region YOUR_REGION create-cli-token --
   name YOUR_ENVIRONMENT_NAME) \
     && CLI_TOKEN=$(echo $CLI_JSON | jq -r '.CliToken') \
     && WEB_SERVER_HOSTNAME=$(echo $CLI_JSON | jq -r '.WebServerHostname') \
     && CLI_RESULTS=$(curl --request POST "https://$WEB_SERVER_HOSTNAME/aws_mwaa/
   cli" \
     --header "Authorization: Bearer $CLI_TOKEN" \
   ```

```
  --header "Content-Type: text/plain" \
  --data-raw "dags trigger YOUR_DAG_NAME") \
  && echo "Output:" \
  && echo $CLI_RESULTS | jq -r '.stdout' | base64 --decode \
  && echo "Errors:" \
  && echo $CLI_RESULTS | jq -r '.stderr' | base64 --decode
```

2. Sostituisci i segnapostiYOUR_REGION con laAWS regione per il tuo ambienteYOUR_DAG_NAME, eYOUR_ENVIRONMENT_NAME. Ad esempio, un nome host per una rete pubblica può avere il seguente aspetto (senza https://):

```
123456a0-0101-2020-9e11-1b159eec9000.c2.us-east-1.airflow.amazonaws.com
```

3. Nel prompt dei comandi:

```
{
  "stderr":"<STDERR of the CLI execution (if any), base64 encoded>",
  "stdout":"<STDOUT of the CLI execution, base64 encoded>"
}
```

Apache Airflow v1

1. Copia l'istruzione cURL dal tuo file di testo e incollala nella shell dei comandi.

> **Note**
>
> Dopo averlo copiato negli appunti, potrebbe essere necessario utilizzare Modifica > Incolla dal menu della shell.

```
CLI_JSON=$(aws mwaa --region YOUR_REGION create-cli-token --
name YOUR_ENVIRONMENT_NAME) \
  && CLI_TOKEN=$(echo $CLI_JSON | jq -r '.CliToken') \
  && WEB_SERVER_HOSTNAME=$(echo $CLI_JSON | jq -r '.WebServerHostname') \
  && CLI_RESULTS=$(curl --request POST "https://$WEB_SERVER_HOSTNAME/aws_mwaa/
cli" \
  --header "Authorization: Bearer $CLI_TOKEN" \
  --header "Content-Type: text/plain" \
  --data-raw "trigger_dag YOUR_DAG_NAME") \
  && echo "Output:" \
```

```
  && echo $CLI_RESULTS | jq -r '.stdout' | base64 --decode \
  && echo "Errors:" \
  && echo $CLI_RESULTS | jq -r '.stderr' | base64 --decode
```

2. Sostituisci i segnapostiYOUR_REGION con laAWS regione per il tuo ambienteYOUR_DAG_NAME, eYOUR_HOST_NAME. Ad esempio, un nome host per una rete pubblica può avere il seguente aspetto (senza https://):

```
123456a0-0101-2020-9e11-1b159eec9000.c2.us-east-1.airflow.amazonaws.com
```

3. Nel prompt dei comandi:

```
{
  "stderr":"<STDERR of the CLI execution (if any), base64 encoded>",
  "stdout":"<STDOUT of the CLI execution, base64 encoded>"
}
```

4. Sostituisci i segnaposti conYOUR_ENVIRONMENT_NAME eYOUR_DAG_NAME.

Usare uno script bash

L'esempio seguente utilizza uno script bash per chiamare il create-cli-tokencomando inAWS CLI per creare un token CLI di Apache Airflow.

Apache Airflow v2

1. Copiare il seguente esempio di codice e salvare localmente comeget-cli-token.sh.

```
# brew install jq
  aws mwaa create-cli-token --name YOUR_ENVIRONMENT_NAME | export CLI_TOKEN=$(jq
 -r .CliToken) && curl --request POST "https://YOUR_HOST_NAME/aws_mwaa/cli" \
      --header "Authorization: Bearer $CLI_TOKEN" \
      --header "Content-Type: text/plain" \
      --data-raw "dags trigger YOUR_DAG_NAME"
```

2. Sostituisci i segnaposti in *rosso* conYOUR_ENVIRONMENT_NAMEYOUR_HOST_NAME, eYOUR_DAG_NAME. Ad esempio, un nome host per una rete pubblica può avere il seguente aspetto (senza https://):

```
123456a0-0101-2020-9e11-1b159eec9000.c2.us-east-1.airflow.amazonaws.com
```

3. (opzionale) Gli utenti macOS e Linux potrebbero dover eseguire il seguente comando per assicurarsi che lo script sia eseguibile.

```
chmod +x get-cli-token.sh
```

4. Eseguire lo script seguente per creare un token CLI Apache Airflow.

```
./get-cli-token.sh
```

Apache Airflow v1

1. Copiare il seguente esempio di codice e salvare localmente come`get-cli-token.sh`.

```
# brew install jq
  aws mwaa create-cli-token --name YOUR_ENVIRONMENT_NAME | export CLI_TOKEN=$(jq
 -r .CliToken) && curl --request POST "https://YOUR_HOST_NAME/aws_mwaa/cli" \
      --header "Authorization: Bearer $CLI_TOKEN" \
      --header "Content-Type: text/plain" \
      --data-raw "trigger_dag YOUR_DAG_NAME"
```

2. Sostituisci i segnaposti in *rosso* con`YOUR_ENVIRONMENT_NAME`,`YOUR_HOST_NAME`, e`YOUR_DAG_NAME`. Ad esempio, un nome host per una rete pubblica può avere il seguente aspetto (senza https://):

```
123456a0-0101-2020-9e11-1b159eec9000.c2.us-east-1.airflow.amazonaws.com
```

3. (opzionale) Gli utenti macOS e Linux potrebbero dover eseguire il seguente comando per assicurarsi che lo script sia eseguibile.

```
chmod +x get-cli-token.sh
```

4. Eseguire lo script seguente per creare un token CLI Apache Airflow.

```
./get-cli-token.sh
```

Usare uno script Python

L'esempio seguente utilizza il metodo boto3 create_cli_token in uno script Python per creare un token CLI di Apache Airflow e attivare un DAG. Puoi eseguire questo script al di fuori di Amazon MWAA. L'unica cosa che rimane da fare è installare la libreria boto3. Potresti voler creare un ambiente virtuale per installare la libreria. Si presuppone che tu abbia configurato le credenziali diAWS autenticazione per il tuo account.

Apache Airflow v2

1. Copiare il seguente esempio di codice e salvare localmente come`create-cli-token.py`.

```
"""
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""
import boto3
import json
import requests
import base64

mwaa_env_name = 'YOUR_ENVIRONMENT_NAME'
dag_name = 'YOUR_DAG_NAME'
mwaa_cli_command = 'dags trigger'

client = boto3.client('mwaa')

mwaa_cli_token = client.create_cli_token(
    Name=mwaa_env_name
)
```

```
mwaa_auth_token = 'Bearer ' + mwaa_cli_token['CliToken']
mwaa_webserver_hostname = 'https://{0}/aws_mwaa/
cli'.format(mwaa_cli_token['WebServerHostname'])
raw_data = '{0} {1}'.format(mwaa_cli_command, dag_name)

mwaa_response = requests.post(
        mwaa_webserver_hostname,
        headers={
            'Authorization': mwaa_auth_token,
            'Content-Type': 'text/plain'
            },
        data=raw_data
        )

mwaa_std_err_message = base64.b64decode(mwaa_response.json()
['stderr']).decode('utf8')
mwaa_std_out_message = base64.b64decode(mwaa_response.json()
['stdout']).decode('utf8')

print(mwaa_response.status_code)
print(mwaa_std_err_message)
print(mwaa_std_out_message)
```

2. Sostituisci i segnaposti con`YOUR_ENVIRONMENT_NAME` e`YOUR_DAG_NAME`.
3. Eseguire lo script seguente per creare un token CLI Apache Airflow.

```
python3 create-cli-token.py
```

Apache Airflow v1

1. Copiare il seguente esempio di codice e salvare localmente come`create-cli-token.py`.

```
import boto3
import json
import requests
import base64

mwaa_env_name = 'YOUR_ENVIRONMENT_NAME'
dag_name = 'YOUR_DAG_NAME'
mwaa_cli_command = 'trigger_dag'
```

```
client = boto3.client('mwaa')

mwaa_cli_token = client.create_cli_token(
    Name=mwaa_env_name
)

mwaa_auth_token = 'Bearer ' + mwaa_cli_token['CliToken']
mwaa_webserver_hostname = 'https://{0}/aws_mwaa/
cli'.format(mwaa_cli_token['WebServerHostname'])
raw_data = '{0} {1}'.format(mwaa_cli_command, dag_name)

mwaa_response = requests.post(
        mwaa_webserver_hostname,
        headers={
            'Authorization': mwaa_auth_token,
            'Content-Type': 'text/plain'
            },
        data=raw_data
        )

mwaa_std_err_message = base64.b64decode(mwaa_response.json()
['stderr']).decode('utf8')
mwaa_std_out_message = base64.b64decode(mwaa_response.json()
['stdout']).decode('utf8')

print(mwaa_response.status_code)
print(mwaa_std_err_message)
print(mwaa_std_out_message)
```

2. Sostituisci i segnaposti con`YOUR_ENVIRONMENT_NAME` e`YOUR_DAG_NAME`.
3. Eseguire lo script seguente per creare un token CLI Apache Airflow.

```
python3 create-cli-token.py
```

Fasi successive

- Esplora l'operazione dell'API Amazon MWAA utilizzata per creare un token CLI all'indirizzo CreateCliToken.

Utilizzo dell'API REST di Apache Airflow

Amazon Managed Workflows for Apache Airflow (Amazon MWAA) supporta l'interazione con i tuoi ambienti Apache Airflow direttamente utilizzando l'API REST di Apache Airflow per ambienti che eseguono Apache Airflow v2.4.3 e versioni successive. Ciò consente di accedere e gestire gli ambienti Amazon MWAA in modo programmatico, fornendo un modo standardizzato per richiamare flussi di lavoro di orchestrazione dei dati, gestire i DAG e monitorare lo stato di vari componenti di Apache Airflow come il database di metadati, il trigger e lo scheduler.

Per supportare l'utilizzo diretto dell'API REST di Apache Airflow, Amazon MWAA offre la possibilità di scalare orizzontalmente la capacità del server Web per gestire l'aumento della domanda, che si tratti di richieste API REST, utilizzo dell'interfaccia a riga di comando (CLI) o più utenti simultanei dell'interfaccia utente (UI) Apache Airflow. Per ulteriori informazioni su come Amazon MWAA ridimensiona i server Web, consulta. the section called “Configurazione della scalabilità automatica del server Web”

Puoi utilizzare l'API REST di Apache Airflow per implementare i seguenti casi d'uso per i tuoi ambienti:

- Accesso programmatico: ora puoi avviare le esecuzioni di Apache Airflow DAG, gestire i set di dati e recuperare lo stato di vari componenti come il database dei metadati, i trigger e gli scheduler senza fare affidamento sull'interfaccia utente o sulla CLI di Apache Airflow.
- Integrazione con applicazioni e microservizi esterni: il supporto dell'API REST ti consente di creare soluzioni personalizzate che integrano i tuoi ambienti Amazon MWAA con altri sistemi. Ad esempio, puoi avviare flussi di lavoro in risposta a eventi provenienti da sistemi esterni, come lavori di database completati o registrazioni di nuovi utenti.
- Monitoraggio centralizzato: puoi creare dashboard di monitoraggio che aggregano lo stato dei tuoi DAG in più ambienti Amazon MWAA, abilitando il monitoraggio e la gestione centralizzati.

I seguenti argomenti mostrano come ottenere un token di accesso al server Web, quindi utilizzarlo per effettuare chiamate API all'API REST di Apache Airflow. Nell'esempio seguente, chiamerai l'API per avviare una nuova esecuzione DAG.

Per ulteriori informazioni sull'API REST di Apache Airflow, consulta The Apache Airflow REST API Reference.

Argomenti

- Crea un token di sessione del server Web

- Chiama l'API REST di Apache Airflow

Crea un token di sessione del server Web

Per creare un token di accesso al server Web, utilizzare la seguente funzione Python. Questa funzione chiama innanzitutto l'API Amazon MWAA per ottenere un token di accesso Web. Il token di accesso Web, che scade dopo 60 secondi, viene quindi sostituito con un token di sessione Web, che consente di accedere al server Web e utilizzare l'API REST di Apache Airflow.

> **Note**
>
> Il token di sessione scade dopo 12 ore.

```
def get_session_info(region, env_name):
    logging.basicConfig(level=logging.INFO)

    try:
        # Initialize MWAA client and request a web login token
        mwaa = boto3.client('mwaa', region_name=region)
        response = mwaa.create_web_login_token(Name=env_name)

        # Extract the web server hostname and login token
        web_server_host_name = response["WebServerHostname"]
        web_token = response["WebToken"]

        # Construct the URL needed for authentication
        login_url = f"https://{web_server_host_name}/aws_mwaa/login"
        login_payload = {"token": web_token}

        # Make a POST request to the MWAA login url using the login payload
        response = requests.post(
            login_url,
            data=login_payload,
            timeout=10
        )

        # Check if login was succesfull
        if response.status_code == 200:

            # Return the hostname and the session cookie
```

```
            return (
                web_server_host_name,
                response.cookies["session"]
            )
        else:
            # Log an error
            logging.error("Failed to log in: HTTP %d", response.status_code)
            return None
    except requests.RequestException as e:
         # Log any exceptions raised during the request to the MWAA login endpoint
        logging.error("Request failed: %s", str(e))
        return None
    except Exception as e:
        # Log any other unexpected exceptions
        logging.error("An unexpected error occurred: %s", str(e))
        return None
```

Chiama l'API REST di Apache Airflow

Una volta completata l'autenticazione, disponi delle credenziali per iniziare a inviare richieste agli endpoint dell'API. Nell'esempio seguente, usa l'endpoint. `/dags/`*`dag_id`*`/dag`

```
def trigger_dag(region, env_name, dag_name):
    """
    Triggers a DAG in a specified MWAA environment using the Airflow REST API.

    Args:
    region (str): AWS region where the MWAA environment is hosted.
    env_name (str): Name of the MWAA environment.
    dag_name (str): Name of the DAG to trigger.
    """

    logging.info(f"Attempting to trigger DAG {dag_name} in environment {env_name} at
 region {region}")

    # Retrieve the web server hostname and session cookie for authentication
    try:
        web_server_host_name, session_cookie = get_session_info(region, env_name)
        if not session_cookie:
            logging.error("Authentication failed, no session cookie retrieved.")
            return
    except Exception as e:
```

```
        logging.error(f"Error retrieving session info: {str(e)}")
        return

    # Prepare headers and payload for the request
    cookies = {"session": session_cookie}
    json_body = {"conf": {}}

    # Construct the URL for triggering the DAG
    url = f"https://{web_server_host_name}/api/v1/dags/{dag_id}/dagRuns"

    # Send the POST request to trigger the DAG
    try:
        response = requests.post(url, cookies=cookies, json=json_body)
        # Check the response status code to determine if the DAG was triggered
 successfully
        if response.status_code == 200:
            logging.info("DAG triggered successfully.")
        else:
            logging.error(f"Failed to trigger DAG: HTTP {response.status_code} -
 {response.text}")
    except requests.RequestException as e:
        logging.error(f"Request to trigger DAG failed: {str(e)}")

if __name__ == "__main__":
    logging.basicConfig(level=logging.INFO)

    # Check if the correct number of arguments is provided
    if len(sys.argv) != 4:
        logging.error("Incorrect usage. Proper format: python script_name.py {region}
 {env_name} {dag_name}")
        sys.exit(1)

    region = sys.argv[1]
    env_name = sys.argv[2]
    dag_name = sys.argv[3]

    # Trigger the DAG with the provided arguments
    trigger_dag(region, env_name, dag_name)
```

Riferimento ai comandi CLI di Apache Airflow

Questa pagina descrive i comandi CLI di Apache Airflow supportati e non supportati su Amazon Managed Workflows for Apache Airflow.

Indice

- Prerequisiti
  - Accesso
  - AWS CLI
- Cosa è cambiato nella v2
- Comandi CLI supportati
  - Comandi supportati
  - Utilizzo di comandi che analizzano i DAG
- Codice di esempio
  - Imposta, ottieni o elimina una variabile Apache Airflow v2
  - Aggiungere una configurazione quando si attiva un DAG
  - Esegui i comandi CLI su un tunnel SSH verso un host bastion
  - Esempi GitHub e tutorial AWS

Prerequisiti

La sezione seguente descrive i passaggi preliminari necessari per utilizzare i comandi e gli script in questa pagina.

Accesso

- AWS accesso tramite account AWS Identity and Access Management (IAM) alla policy di autorizzazione di Amazon MWAA in. Politica di accesso all'interfaccia utente di Apache Airflow: AmazonMWAA Access WebServer
- AWS accesso tramite account AWS Identity and Access Management (IAM) alla policy di autorizzazione di Amazon MWAA. Politica completa di accesso all'API e alla console: FullApi AmazonMWAA Access

AWS CLI

Il AWS Command Line Interface (AWS CLI) è uno strumento open source che consente di interagire con i AWS servizi utilizzando i comandi nella shell della riga di comando. Per completare la procedura descritta in questa pagina, è necessario quanto segue:

- AWS CLI — Installa la versione 2.

- AWS CLI — Configurazione rapida con `aws configure`.

Cosa è cambiato nella v2

- Novità: struttura dei comandi CLI Airflow. La CLI di Apache Airflow v2 è organizzata in modo che i comandi correlati siano raggruppati come sottocomandi, il che significa che è necessario aggiornare gli script di Apache Airflow v1 se si desidera eseguire l'aggiornamento ad Apache Airflow v2. Ad esempio, in Apache Airflow v1 è ora in Apache Airflow v2. `unpause dags unpause` Per ulteriori informazioni, consulta le modifiche alla CLI di Airflow in 2 nella guida di riferimento di Apache Airflow.

Comandi CLI supportati

La sezione seguente elenca i comandi CLI di Apache Airflow disponibili su Amazon MWAA.

Comandi supportati

Apache Airflow v2

Versioni secondarie	Comando
v2.0 e versioni successive	cheatsheet
v2.0 +	connessioni aggiungi
v2.0 +	eliminazione delle connessioni
v2.2+ (nota)	4 giorni di riempimento
v2.0 +	giorni eliminati
v2.2+ (nota)	elenco dei giorni
v2.0 +	dags list-jobs
v2.6 +	giorni list-import-errors
v2.2+ (nota)	dags list-run

Versioni secondarie	Comando
v2.2+ (nota)	dags successiva esecuzione
v2.0 +	giorni di pausa
v2.0 +	rapporto dags
v2.4 e versioni successive	i dag riserializzano
v2.0 +	mostra giorni
v2.0 +	stato dei giorni
v2.0 +	giorni di test
v2.0 +	i giorni si innescano
v2.0 +	giorni in pausa
v2.4 e versioni successive	db pulito
v2.0 +	comportamenti dei fornitori
v2.0 +	i fornitori ottengono
v2.0 +	ganci per fornitori
v2.0 +	link ai fornitori
v2.0 +	elenco dei fornitori
v2.8 +	notifiche ai fornitori
v2.6+	segreti dei fornitori
v2.7 +	trigger dei provider
v2.0 +	widget dei fornitori
v2.6 +	ruoli add-perms

Versioni secondarie	Comando
v2.6 +	ruoli del-perms
v2.6 +	i ruoli creano
v2.0 +	elenco dei ruoli
v2.0 +	compiti chiari
v2.0 +	attività fallite - deps
v2.0 +	elenco delle attività
v2.0 +	renderizzazione delle attività
v2.0 +	stato delle attività
v2.0 +	attività states-for-dag-run
v2.0 +	test delle attività
v2.0 +	le variabili si eliminano
v2.0 +	le variabili ottengono
v2.0 +	set di variabili
v2.0 +	elenco delle variabili
v2.0 +	versione

Utilizzo di comandi che analizzano i DAG

Se nel tuo ambiente è in esecuzione Apache Airflow v1.10.12 o v2.0.2, i comandi CLI che analizzano i DAG avranno esito negativo se il DAG utilizza plugin che dipendono dai pacchetti installati tramite: `requirements.txt`

Apache Airflow v2.0.2

- `dags backfill`

- `dags list`
- `dags list-runs`
- `dags next-execution`

È possibile utilizzare questi comandi CLI se i DAG non utilizzano plug-in che dipendono dai pacchetti installati tramite un. `requirements.txt`

Codice di esempio

La sezione seguente contiene esempi di diversi modi di utilizzare la CLI di Apache Airflow.

Imposta, ottieni o elimina una variabile Apache Airflow v2

È possibile utilizzare il seguente codice di esempio per impostare, ottenere o eliminare una variabile nel formato di. `<script> <mwaa env name> get | set | delete <variable> <variable value> </variable> </variable>`

```
[ $# -eq 0 ] && echo "Usage: $0 MWAA environment name " && exit

if [[ $2 == "" ]]; then
    dag="variables list"

elif  [ $2 == "get" ] ||  [ $2 == "delete" ] ||  [ $2 == "set" ]; then
    dag="variables $2 $3 $4 $5"

else
    echo "Not a valid command"
    exit 1
fi

CLI_JSON=$(aws mwaa --region $AWS_REGION create-cli-token --name $1) \
    && CLI_TOKEN=$(echo $CLI_JSON | jq -r '.CliToken') \
    && WEB_SERVER_HOSTNAME=$(echo $CLI_JSON | jq -r '.WebServerHostname') \
    && CLI_RESULTS=$(curl --request POST "https://$WEB_SERVER_HOSTNAME/aws_mwaa/cli" \
    --header "Authorization: Bearer $CLI_TOKEN" \
    --header "Content-Type: text/plain" \
    --data-raw "$dag" ) \
    && echo "Output:" \
    && echo $CLI_RESULTS | jq -r '.stdout' | base64 --decode \
    && echo "Errors:" \
    && echo $CLI_RESULTS | jq -r '.stderr' | base64 --decode
```

Aggiungere una configurazione quando si attiva un DAG

È possibile utilizzare il seguente codice di esempio con Apache Airflow v1 e Apache Airflow v2 per aggiungere una configurazione quando si attiva un DAG, ad esempio. `airflow trigger_dag 'dag_name' –conf '{"key":"value"}'`

```
import boto3
import json
import requests
import base64

mwaa_env_name = 'YOUR_ENVIRONMENT_NAME'
dag_name = 'YOUR_DAG_NAME'
key = "YOUR_KEY"
value = "YOUR_VALUE"
conf = "{\"" + key + "\":\"" + value + "\"}"

client = boto3.client('mwaa')

mwaa_cli_token = client.create_cli_token(
  Name=mwaa_env_name
)

mwaa_auth_token = 'Bearer ' + mwaa_cli_token['CliToken']
mwaa_webserver_hostname = 'https://{0}/aws_mwaa/
cli'.format(mwaa_cli_token['WebServerHostname'])
raw_data = "trigger_dag {0} -c '{1}'".format(dag_name, conf)

mwaa_response = requests.post(
      mwaa_webserver_hostname,
      headers={
          'Authorization': mwaa_auth_token,
          'Content-Type': 'text/plain'
          },
      data=raw_data
      )

mwaa_std_err_message = base64.b64decode(mwaa_response.json()['stderr']).decode('utf8')
mwaa_std_out_message = base64.b64decode(mwaa_response.json()['stdout']).decode('utf8')

print(mwaa_response.status_code)
print(mwaa_std_err_message)
print(mwaa_std_out_message)
```

Esegui i comandi CLI su un tunnel SSH verso un host bastion

L'esempio seguente mostra come eseguire i comandi CLI Airflow utilizzando un proxy tunnel SSH su un host Linux Bastion.

Usare curl

1.
```
ssh -D 8080 -f -C -q -N YOUR_USER@YOUR_BASTION_HOST
```
2.
```
curl -x socks5h://0:8080 --request POST https://YOUR_HOST_NAME/aws_mwaa/cli --
header YOUR_HEADERS --data-raw YOUR_CLI_COMMAND
```

Esempi GitHub e tutorial AWS

- Utilizzo dei parametri e delle variabili di Apache Airflow v2.0.2 in Amazon Managed Workflows for Apache Airflow
- Interazione con Apache Airflow v1.10.12 su Amazon MWAA tramite la riga di comando
- Comandi interattivi con Apache Airflow v1.10.12 su Amazon MWAA e Bash Operator su GitHub

Gestione delle connessioni ad Apache Airflow

Questa sezione descrive i diversi modi per configurare una connessione Apache Airflow for Apache Apache Airflow for Apache Apache Airflow.

Argomenti

- Panoramica delle variabili e delle connessioni di Apache Airflow
- Pacchetti del provider Apache Airflow installati in ambienti Amazon MWAA
- Panoramica dei tipi di connessione
- Configurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager

Panoramica delle variabili e delle connessioni di Apache Airflow

In alcuni casi, potresti voler specificare connessioni o variabili aggiuntive per un ambiente, ad esempio unAWS profilo, o aggiungere il tuo ruolo di esecuzione in un oggetto di connessione nel metastore di Apache Airflow, quindi fare riferimento alla connessione dall'interno di un DAG.

- Apache Airflow Apache Apache Airflow Flow In un'installazione di Apache Airflow autogestita, si impostano le opzioni di configurazione di Apache Airflow in`airflow.cfg`.

  ```
  [secrets]
  backend = airflow.providers.amazon.aws.secrets.secrets_manager.SecretsManagerBackend
  backend_kwargs = {"connections_prefix" : "airflow/connections", "variables_prefix" :
   "airflow/variables"}
  ```

- Apache Airflow su Amazon MWAA. Su Amazon MWAA, devi aggiungere queste impostazioni di configurazione come opzioni di configurazione di Apache Airflow sulla console Amazon MWAA. Le opzioni di configurazione di Apache Airflow sono scritte come variabili di ambiente nell'ambiente e sostituiscono tutte le altre configurazioni esistenti per la stessa impostazione.

Pacchetti del provider Apache Airflow installati in ambienti Amazon MWAA

Amazon MWAA installa gli extra del provider per i tipi di connessione Apache Airflow v2 e versioni successive quando crei un nuovo ambiente. L'installazione di pacchetti provider consente di

visualizzare un tipo di connessione nell'interfaccia utente di Apache Airflow. Significa anche che non è necessario specificare questi pacchetti come dipendenza da Python nel file. `requirements.txt` Questa pagina elenca i pacchetti del provider Apache Airflow installati da Amazon MWAA per tutti gli ambienti Apache Airflow v2.

> **Note**
>
> Per Apache Airflow v2 e versioni successive, Amazon MWAA installa Watchtower versione 2.0.1 dopo l'esecuzione`pip3 install -r requirements.txt`, per garantire che la compatibilità con la registrazione non CloudWatch venga sostituita da altre installazioni di librerie Python.

Indice

- Pacchetti provider per connessioni Apache Airflow v2.8.1
- Pacchetti provider per connessioni Apache Airflow v2.7.2
- Pacchetti provider per connessioni Apache Airflow v2.6.3
- Pacchetti provider per connessioni Apache Airflow v2.5.1
- Pacchetti provider per connessioni Apache Airflow v2.4.3
- Pacchetti provider per connessioni Apache Airflow v2.2.2
- Pacchetti provider per connessioni Apache Airflow v2.0.2
- Specificare pacchetti di provider più recenti

Pacchetti provider per connessioni Apache Airflow v2.8.1

Quando crei un ambiente Amazon MWAA in Apache Airflow v2.8.1, Amazon MWAA installa i seguenti pacchetti di provider utilizzati per le connessioni Apache Airflow.

> **Note**
>
> Puoi specificare l'ultima versione supportata di per aggiornare questo provider. `apache-airflow-providers-amazon` Per ulteriori informazioni su come specificare le versioni più recenti, vedere. the section called “Specificare pacchetti di provider più recenti”

Tipo di connessione	Pacchetto
AWS Connessione	apache-airflow-providers-amazon[aiobotocore]==8.16.0
Connessione Postgres	apache-airflow-providers-postgres==5.10.0
Connessione FTP	apache-airflow-providers-ftp==3.7.0
Connessione Celery	apache-airflow-providers-celery==3.5.1
Connessione HTTP	apache-airflow-providers-http==4.8.0
Connessione IMAP	apache-airflow-providers-imap==3.5.0
SQL comune	apache-airflow-providers-common-sql==1.10.0
Connessione SQLite	apache-airflow-providers-sqlite==3.7.0

Pacchetti provider per connessioni Apache Airflow v2.7.2

Quando crei un ambiente Amazon MWAA in Apache Airflow v2.7.2, Amazon MWAA installa i seguenti pacchetti di provider utilizzati per le connessioni Apache Airflow.

> **Note**
>
> Puoi specificare l'ultima versione supportata di per aggiornare questo provider. `apache-airflow-providers-amazon` Per ulteriori informazioni su come specificare le versioni più recenti, vedere. the section called “Specificare pacchetti di provider più recenti”

Tipo di connessione	Pacchetto
AWS Connessione	apache-airflow-providers-amazon[aiobotocore]==8.7.1
Connessione Postgres	apache-airflow-providers-postgres==5.6.1

Tipo di connessione	Pacchetto
Connessione FTP	apache-airflow-providers-ftp==3.5.2
Connessione Celery	apache-airflow-providers-celery==3.3.4
Connessione HTTP	apache-airflow-providers-http==4.5.2
Connessione IMAP	apache-airflow-providers-imap==3.3.2
SQL comune	apache-airflow-providers-common-sql==1.7.2
Connessione SQLite	apache-airflow-providers-sqlite==3.4.3

Pacchetti provider per connessioni Apache Airflow v2.6.3

Quando crei un ambiente Amazon MWAA in Apache Airflow v2.6.3, Amazon MWAA installa i seguenti pacchetti di provider utilizzati per le connessioni Apache Airflow.

> **Note**
>
> Puoi specificare l'ultima versione supportata di per aggiornare questo provider. `apache-airflow-providers-amazon` Per ulteriori informazioni su come specificare le versioni più recenti, vedere. the section called “Specificare pacchetti di provider più recenti”

Tipo di connessione	Pacchetto
AWS Connessione	apache-airflow-providers-amazon[aiobotocore]==8.2.0
Connessione Postgres	apache-airflow-providers-postgres==5.5.1
Connessione FTP	apache-airflow-providers-ftp==3.4.2
Connessione Celery	apache-airflow-providers-celery==3.2.1
Connessione HTTP	apache-airflow-providers-http==4.4.2

Tipo di connessione	Pacchetto
Connessione IMAP	apache-airflow-providers-imap==3.2.2
SQL comune	apache-airflow-providers-common-sql==1.5.2
Connessione SQLite	apache-airflow-providers-sqlite==3.4.2

Pacchetti provider per connessioni Apache Airflow v2.5.1

Quando crei un ambiente Amazon MWAA in Apache Airflow v2.5.1, Amazon MWAA installa i seguenti pacchetti di provider utilizzati per le connessioni Apache Airflow.

> **Note**
>
> Puoi specificare l'ultima versione supportata di per aggiornare questo provider. `apache-airflow-providers-amazon` Per ulteriori informazioni su come specificare le versioni più recenti, vedere. the section called “Specificare pacchetti di provider più recenti”

Tipo di connessione	Pacchetto
AWS Connessione	apache-airflow-providers-amazon==7.1.0
Connessione Postgres	apache-airflow-providers-postgres==5.4.0
Connessione FTP	apache-airflow-providers-ftp==3.3.0
Connessione Celery	apache-airflow-providers-celery==3.1.0
Connessione HTTP	apache-airflow-providers-http==4.1.1
Connessione IMAP	apache-airflow-providers-imap==3.1.1
SQL comune	apache-airflow-providers-common-sql==1.3.3
Connessione SQLite	apache-airflow-providers-sqlite==3.3.1

Pacchetti provider per connessioni Apache Airflow v2.4.3

Quando crei un ambiente Amazon MWAA in Apache Airflow v2.4.3, Amazon MWAA installa i seguenti pacchetti di provider utilizzati per le connessioni Apache Airflow.

Tipo di connessione	Pacchetto
AWS Connessione	apache-airflow-providers-amazon==6.0.0
Connessione Postgres	apache-airflow-providers-postgres==5.2.2
Connessione FTP	apache-airflow-providers-ftp==3.1.0
Connessione Celery	apache-airflow-providers-celery==3.0.0
Connessione HTTP	apache-airflow-providers-http==4.0.0
Connessione IMAP	apache-airflow-providers-imap==3.0.0
SQL comune	apache-airflow-providers-common-sql==1.2.0
Connessione SQLite	apache-airflow-providers-sqlite==3.2.1

Pacchetti provider per connessioni Apache Airflow v2.2.2

Quando crei un ambiente Amazon MWAA in Apache Airflow v2.2.2, Amazon MWAA installa i seguenti pacchetti di provider utilizzati per le connessioni Apache Airflow.

Tipo di connessione	Pacchetto
AWS Connessione	apache-airflow-providers-amazon==2.4.0
Connessione Postgres	apache-airflow-providers-postgres==2.3.0
Connessione FTP	apache-airflow-providers-ftp==2.0.1
Connessione Celery	apache-airflow-providers-celery==2.1.0
Connessione HTTP	apache-airflow-providers-http==2.0.1

Tipo di connessione	Pacchetto
Connessione IMAP	apache-airflow-providers-imap==2.0.1
Connessione SQLite	apache-airflow-providers-sqlite==2.0.1

Pacchetti provider per connessioni Apache Airflow v2.0.2

Quando crei un ambiente Amazon MWAA in Apache Airflow v2.0.2, Amazon MWAA installa i seguenti pacchetti di provider utilizzati per le connessioni Apache Airflow.

Tipo di connessione	Pacchetto
Connessione Tableau	apache-airflow-providers-tableau==1.0.0
Connessione Databricks	apache-airflow-providers-databricks==1.0.1
Connessione SSH	apache-airflow-providers-ssh==1.3.0
Connessione Postgres	apache-airflow-providers-postgres==1.0.2
Connessione Docker	apache-airflow-providers-docker==1.2.0
Collegamento Oracle	apache-airflow-providers-oracle==1.1.0
Connessione Presto	apache-airflow-providers-presto==1.0.2
Connessione SFTP	apache-airflow-providers-sftp==1.2.0

Specificare pacchetti di provider più recenti

A partire da Apache Airflow v2.7.2, il file dei requisiti deve includere una dichiarazione. `--constraint` Se non fornisci un vincolo, Amazon MWAA te ne specificherà uno per garantire che i pacchetti elencati nei tuoi requisiti siano compatibili con la versione di Apache Airflow che stai utilizzando.

I file di vincoli di Apache Airflow specificano le versioni del provider disponibili al momento del rilascio di Apache Airflow. In molti casi, tuttavia, i provider più recenti sono compatibili con quella versione di

Apache Airflow. Poiché è necessario utilizzare i vincoli, per specificare una versione più recente di un pacchetto provider, è possibile modificare il file dei vincoli per una versione specifica del provider:

1. Scaricate il file dei vincoli specifici della versione da https://raw.githubusercontent.com/apache/airflow/constraints-2.7.2/constraints-3.11.txt "
2. Modificate la `apache-airflow-providers-amazon` versione nel file dei vincoli con la versione che desiderate utilizzare.
3. Salva il file dei vincoli modificato nella cartella Amazon S3 dags del tuo ambiente Amazon MWAA, ad esempio come `constraints-3.11-updated.txt`
4. Specificate i vostri requisiti come illustrato di seguito.

```
--constraint "/usr/local/airflow/dags/constraints-3.11-updated.txt"

apache-airflow-providers-amazon==version-number
```

> **Note**
>
> Se utilizzi un server Web privato, ti consigliamo di impacchettare le librerie richieste come file WHL utilizzando Amazon MWAA local-runner.

Panoramica dei tipi di connessione

Apache Airflow memorizza le connessioni come stringa URI di connessione. Fornisce un modello di connessioni nell'interfaccia utente di Apache Airflow per generare la stringa URI di connessione, indipendentemente dal tipo di connessione. Se un modello di connessione non è disponibile nell'interfaccia utente di Apache Airflow, è possibile utilizzare un modello di connessione alternativo per generare questa stringa URI di connessione, ad esempio utilizzando il modello di connessione HTTP. La differenza principale è il prefisso URI, ad esempio`my-conn-type://`, che i provider di Apache Airflow in genere ignorano per una connessione. Questa pagina descrive come utilizzare i modelli di connessione nell'interfaccia utente di Apache Airflow in modo intercambiabile per diversi tipi di connessione.

> **Warning**
>
> Non sovrascrivere la `aws_default`connessione in Amazon MWAA. Amazon MWAA utilizza questa connessione per eseguire una serie di attività critiche, come la raccolta dei registri

delle attività. La sovrascrittura di questa connessione potrebbe causare la perdita di dati e l'interruzione della disponibilità dell'ambiente.

Argomenti

- Esempio di stringa URI di connessione
- Modello di connessione
- Esempio di utilizzo di un modello di connessione HTTP per una connessione Jdbc

Esempio di stringa URI di connessione

L'esempio seguente mostra una stringa URI di connessione per il tipo di connessione MySQL.

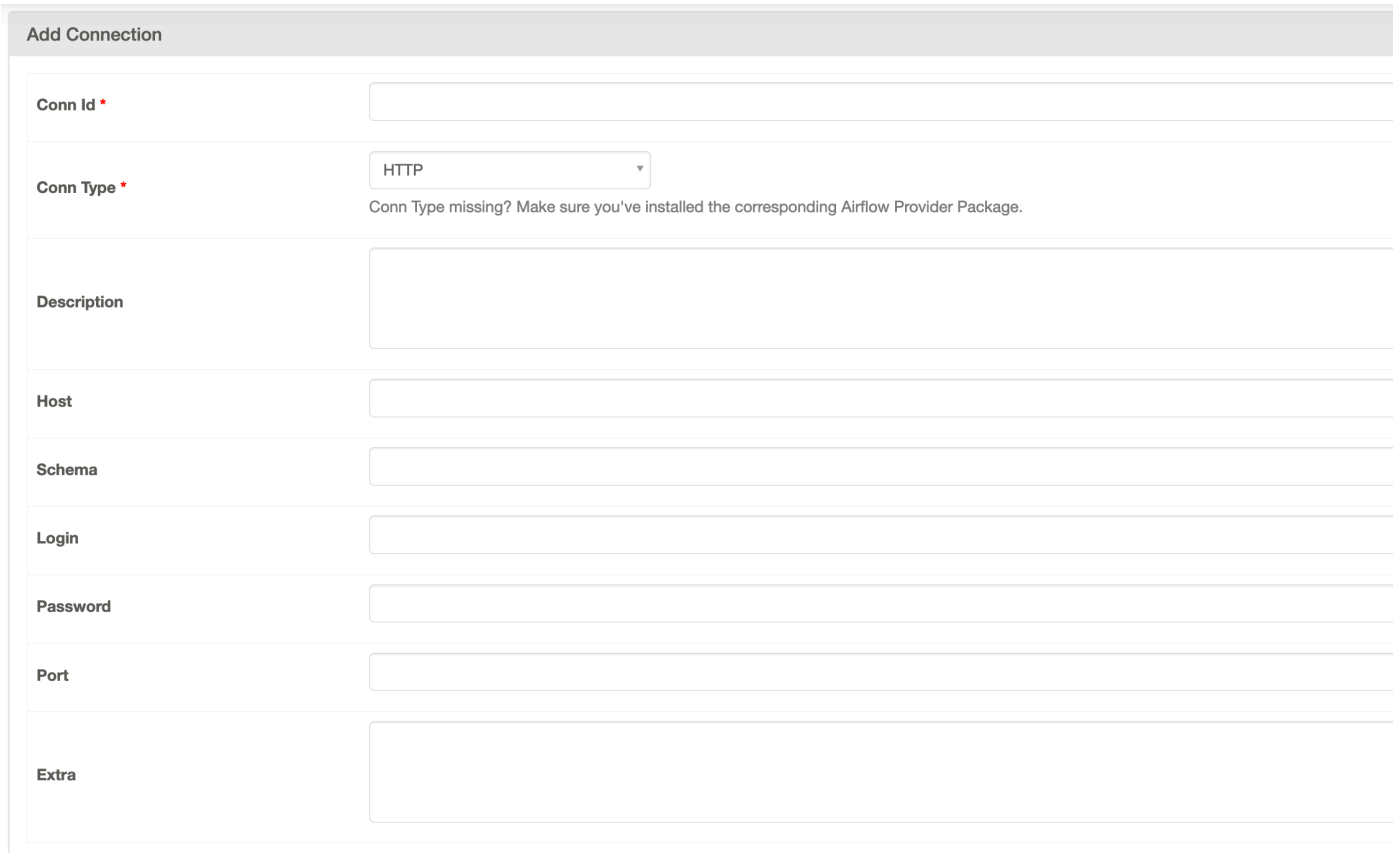
```
'mysql://288888a0-50a0-888-9a88-1a111aaa0000.a1.us-east-1.airflow.amazonaws.com
%2Fhome?role_arn=arn%3Aaws%3Aiam%3A%3A001122332255%3Arole%2Fservice-role%2FAmazonMWAA-
MyAirflowEnvironment-iAaaaA&region_name=us-east-1'
```

Modello di connessione

L'esempio seguente mostra il modello di connessione HTTP nell'interfaccia utente di Apache Airflow.

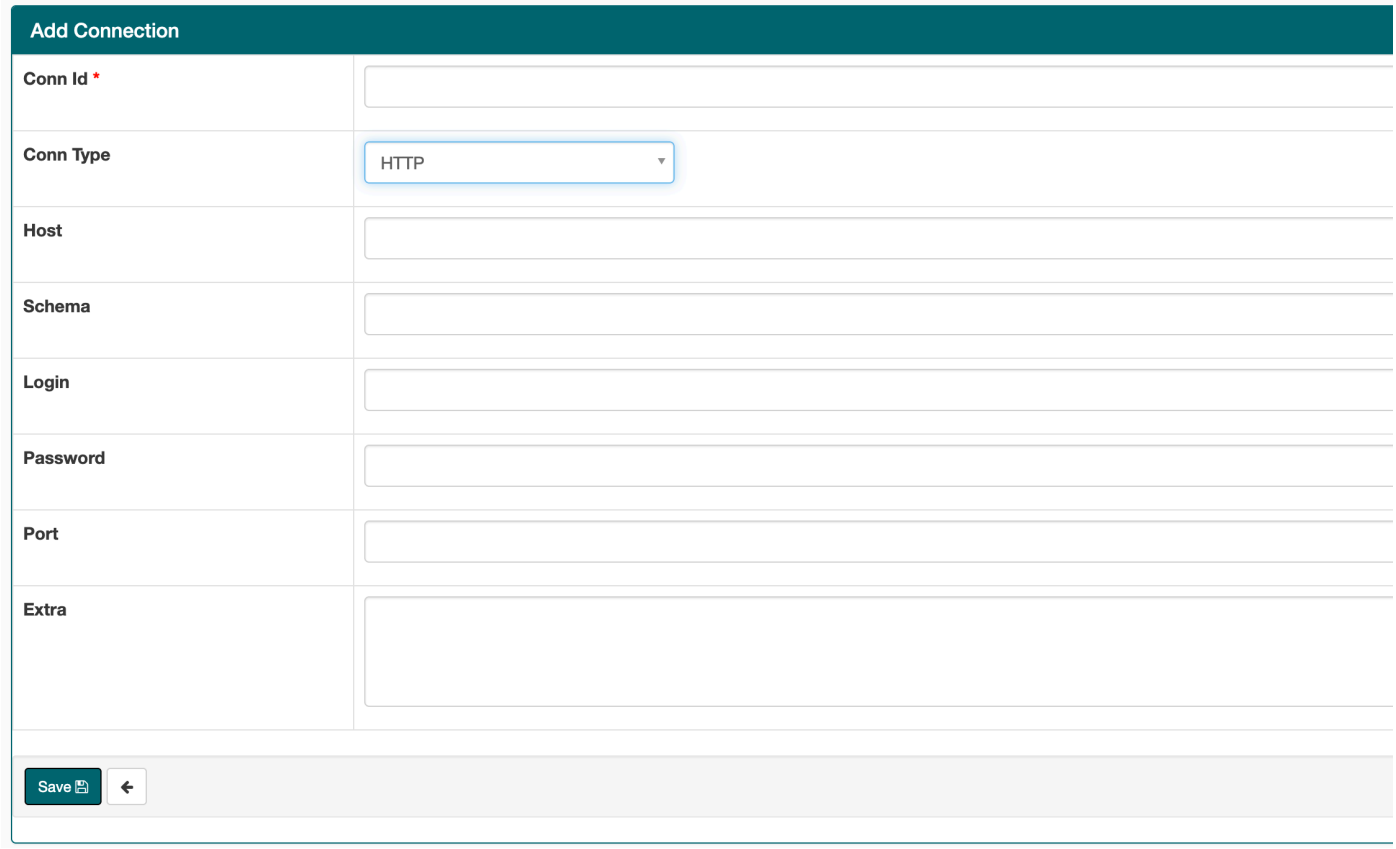
Apache Airflow v2

L'esempio seguente mostra il modello di connessione HTTP per Apache Airflow v2 nell'interfaccia utente di Apache Airflow.



Apache Airflow v1

L'esempio seguente mostra il modello di connessione HTTP per Apache Airflow v1 nell'interfaccia utente di Apache Airflow.



Esempio di utilizzo di un modello di connessione HTTP per una connessione Jdbc

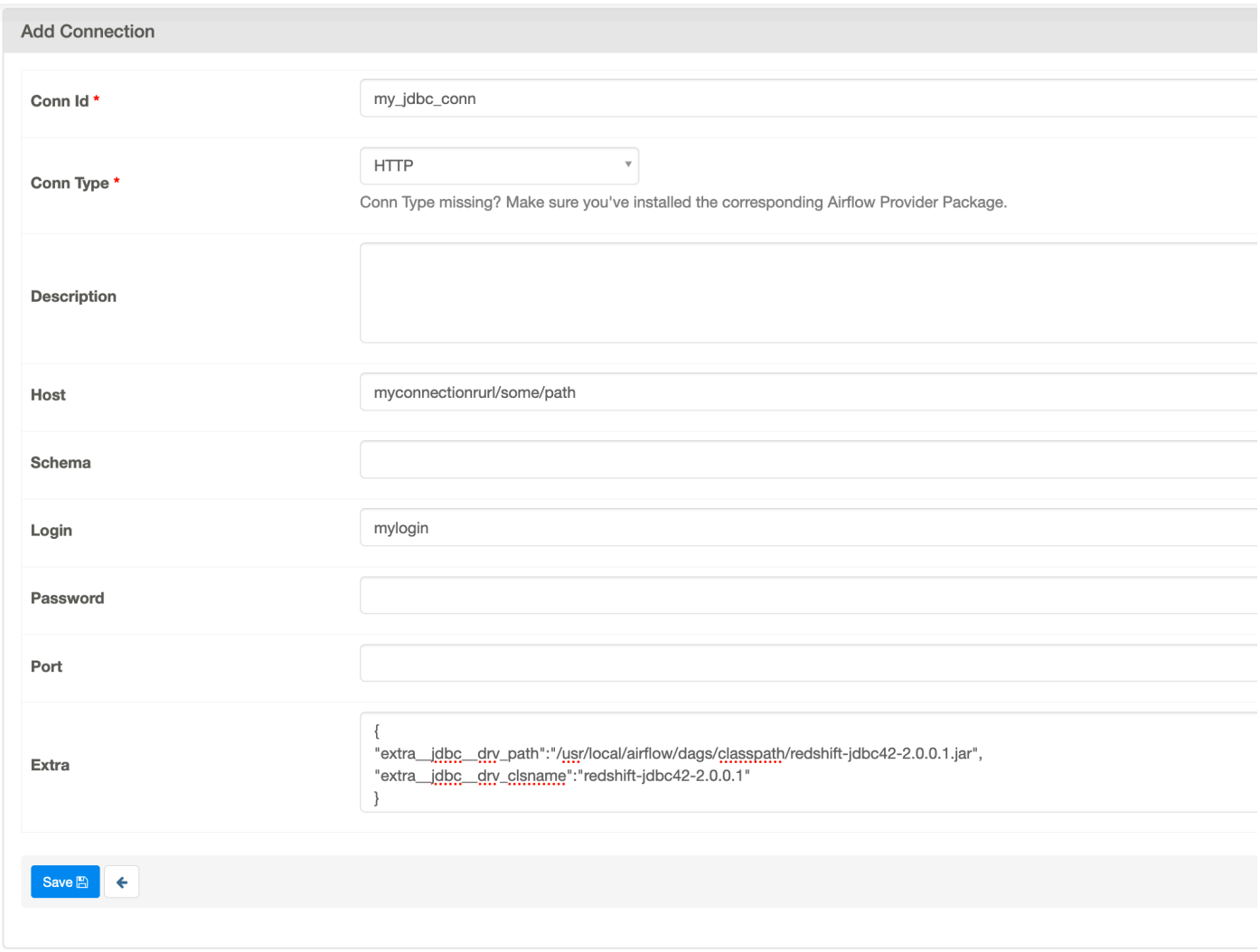
L'esempio seguente mostra come utilizzare il modello di connessione HTTP per un tipo di connessione Jdbc in Apache Airflow v2.0.2 e gli stessi valori nel modello di connessione Jdbc per Apache Airflow v1.10.12 nell'interfaccia utente di Apache Airflow.

Apache Airflow v2

L'esempio seguente mostra la stringa URI di connessione generata da Apache Airflow per l'esempio in questa sezione.

```
http://myconnectionurl/some/path&login=mylogin&extra__jdbc__dry__path=usr/local/
airflow/dags/classpath/redshif-
jdbc42-2.0.0.1.jar&extra__jdbc__dry__clsname=redshift-jdbc42-2.0.0.1
```

L'esempio seguente mostra come utilizzare il modello di connessione HTTP per una connessione Jdbc per Apache Airflow v2 nell'interfaccia utente di Apache Airflow.

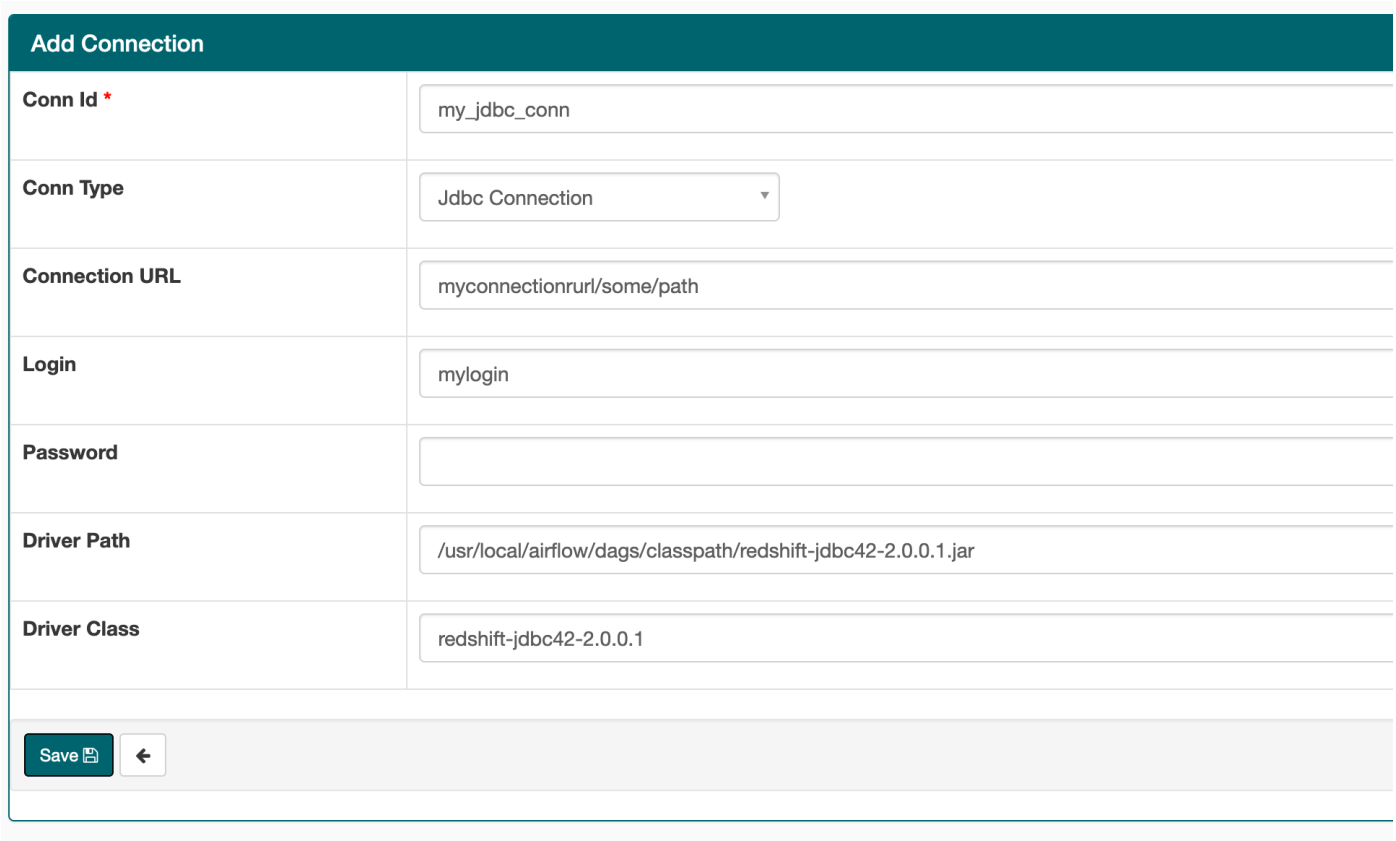


Apache Airflow v1

L'esempio seguente mostra la stringa URI di connessione generata da Apache Airflow per l'esempio in questa sezione.

```
jdbc://myconnectionurl/some/path&login=mylogin&extra__jdbc__dry__path=usr/local/
airflow/dags/classpath/redshif-
jdbc42-2.0.0.1.jar&extra__jdbc__dry__clsname=redshift-jdbc42-2.0.0.1
```

L'esempio seguente mostra il modello di connessione Jdbc per Apache Airflow v1.10.12 nell'interfaccia utente di Apache Airflow.



Configurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager

AWS Secrets Manager è un backend Apache Airflow alternativo supportato in un ambiente Amazon Managed Workflows for Apache Airflow. Questa guida mostra come archiviare in modo sicuro i segreti per le variabili di Apache Airflow e una connessione Apache Airflow su Amazon Managed Workflows for Apache Airflow. AWS Secrets Manager

> **Note**
>
> - Ti verranno addebitati i costi per i segreti che crei. Per ulteriori informazioni sui prezzi di Secrets Manager, consulta AWS Prezzi.

Indice

- Fase uno: fornire ad Amazon MWAA l'autorizzazione ad accedere alle chiavi segrete di Secrets Manager

- Fase due: creare il backend Secrets Manager come opzione di configurazione di Apache Airflow
- Fase tre: generare una stringa URI di connessione Apache Airflow AWS
- Fase quattro: Aggiungere le variabili in Secrets Manager
- Fase cinque: aggiungere la connessione in Secrets Manager
- Codice di esempio
- Risorse
- Fasi successive

Fase uno: fornire ad Amazon MWAA l'autorizzazione ad accedere alle chiavi segrete di Secrets Manager

Il ruolo di esecuzione per il tuo ambiente Amazon MWAA richiede l'accesso in lettura alla chiave segreta. AWS Secrets Manager La seguente policy IAM consente l'accesso in lettura/scrittura utilizzando la policy gestita. AWS SecretsManagerReadWrite

Per allegare la policy al tuo ruolo di esecuzione

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli il tuo ruolo di esecuzione nel riquadro Autorizzazioni.
4. Scegli Collega policy.
5. Digita `SecretsManagerReadWrite` nel campo di testo delle politiche di filtro.
6. Scegli Collega policy.

Se non desideri utilizzare una politica di autorizzazione AWS gestita, puoi aggiornare direttamente il ruolo di esecuzione del tuo ambiente per consentire qualsiasi livello di accesso alle tue risorse Secrets Manager. Ad esempio, la seguente dichiarazione politica concede l'accesso in lettura a tutti i segreti creati in una AWS regione specifica in Secrets Manager.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "secretsmanager:GetResourcePolicy",
                "secretsmanager:GetSecretValue",
                "secretsmanager:DescribeSecret",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-west-2:012345678910:secret:*"
        },
        {
            "Effect": "Allow",
            "Action": "secretsmanager:ListSecrets",
            "Resource": "*"
        }
    ]
}
```

Fase due: creare il backend Secrets Manager come opzione di configurazione di Apache Airflow

La sezione seguente descrive come creare un'opzione di configurazione Apache Airflow sulla console Amazon MWAA per il backend. AWS Secrets Manager Se utilizzi un'impostazione di configurazione con lo stesso nome in`airflow.cfg`, la configurazione creata nei passaggi seguenti avrà la precedenza e sostituirà le impostazioni di configurazione.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.
4. Seleziona Successivo.
5. Scegli Aggiungi configurazione personalizzata nel riquadro delle opzioni di configurazione Airflow. Aggiungi le seguenti coppie chiave-valore:

    a. **`secrets.backend`**: **`airflow.providers.amazon.aws.secrets.secrets_manager.SecretsManagerBacken`**

    b. **`secrets.backend_kwargs`**: **`{"connections_prefix" : "airflow/connections", "variables_prefix" : "airflow/variables"}`** Questo configura Apache Airflow per cercare stringhe e variabili di connessione in e percorsi. `airflow/connections/*` `airflow/variables/*`

    Puoi utilizzare un modello di ricerca per ridurre il numero di chiamate API che Amazon MWAA effettua a Secrets Manager per tuo conto. Se non specifichi un modello di ricerca,

Apache Airflow cerca tutte le connessioni e le variabili nel backend configurato. Specificando uno schema, si restringono i possibili percorsi di visualizzazione di Apache Airflow. Ciò consente di ridurre i costi quando si utilizza Secrets Manager con Amazon MWAA.

Per specificare un modello di ricerca, specifica i parametri and. `connections_lookup_pattern` `variables_lookup_pattern` Questi parametri accettano una RegEx stringa come input. Ad esempio, per cercare segreti che iniziano `contest`, inserisci quanto segue per`secrets.backend_kwargs`:

```
{
  "connections_prefix": "airflow/connections",
  "connections_lookup_pattern": "^test",
  "variables_prefix" : "airflow/variables",
  "variables_lookup_pattern": "^test"
}
```

> **Note**
>
> Per utilizzare `connections_lookup_pattern` e`variables_lookup_pattern`, è necessario installare la `apache-airflow-providers-amazon` versione 7.3.0 o successiva. Per ulteriori informazioni sull'aggiornamento dei pacchetti del provider alle versioni più recenti, vedere. the section called “Specificare pacchetti di provider più recenti”

6. Selezionare Salva.

Fase tre: generare una stringa URI di connessione Apache Airflow AWS

Per creare una stringa di connessione, utilizzate il tasto «tab» sulla tastiera per indentare le coppie chiave-valore nell'oggetto Connection. Consigliamo inoltre di creare una variabile per l'`extra`oggetto nella sessione di shell. La sezione seguente illustra i passaggi per generare una stringa URI di connessione Apache Airflow per un ambiente Amazon MWAA utilizzando Apache Airflow o uno script Python.

Apache Airflow CLI

La seguente sessione di shell utilizza la CLI Airflow locale per generare una stringa di connessione. Se non hai installato la CLI, ti consigliamo di usare lo script Python.

1. Apri una sessione di shell Python:

```
python3
```

2. Immetti il comando seguente:

```
>>> import json
```

3. Immetti il comando seguente:

```
>>> from airflow.models.connection import Connection
```

4. Crea una variabile nella tua sessione di shell per l'`extra`oggetto. *Sostituisci i valori di esempio in YOUR_EXECUTION_ROLE_ARN con il ruolo di esecuzione ARN e la regione in YOUR_REGION (ad esempio).* `us-east-1`

```
>>> extra=json.dumps({'role_arn': 'YOUR_EXECUTION_ROLE_ARN', 'region_name':
 'YOUR_REGION'})
```

5. Create l'oggetto di connessione. Sostituisci il valore di esempio `myconn` con il nome della connessione Apache Airflow.

```
>>> myconn = Connection(
```

6. Usa il tasto «tab» sulla tastiera per indentare ciascuna delle seguenti coppie chiave-valore nell'oggetto di connessione. *Sostituisci i valori del campione in rosso.*

   a. Specificate il tipo di AWS connessione:

   ```
   ... conn_id='aws',
   ```

   b. Specificate l'opzione del database Apache Airflow:

   ```
   ... conn_type='mysql',
   ```

   c. Specificare l'URL dell'interfaccia utente di Apache Airflow su Amazon MWAA:

   ```
   ... host='288888a0-50a0-888-9a88-1a111aaa0000.a1.us-
   east-1.airflow.amazonaws.com/home',
   ```

d. Specificare l'ID della chiave di AWS accesso (nome utente) per accedere ad Amazon MWAA:

```
... login='YOUR_AWS_ACCESS_KEY_ID',
```

e. Specificare la chiave di accesso AWS segreta (password) per accedere ad Amazon MWAA:

```
... password='YOUR_AWS_SECRET_ACCESS_KEY',
```

f. Specificare la variabile di sessione della `extra` shell:

```
... extra=extra
```

g. Chiudi l'oggetto di connessione.

```
... )
```

7. Stampa la stringa URI di connessione:

```
>>> myconn.get_uri()
```

Dovresti vedere la stringa URI di connessione nella risposta:

```
'mysql://288888a0-50a0-888-9a88-1a111aaa0000.a1.us-east-1.airflow.amazonaws.com
%2Fhome?role_arn=arn%3Aaws%3Aiam%3A%3A001122332255%3Arole%2Fservice-role
%2FAmazonMWAA-MyAirflowEnvironment-iAaaaA&region_name=us-east-1'
```

Python script

Il seguente script Python non richiede la CLI Apache Airflow.

1. Copia il contenuto del seguente esempio di codice e salvalo localmente come. `mwaa_connection.py`

```
import urllib.parse

conn_type = 'YOUR_DB_OPTION'
host = 'YOUR_MWAA_AIRFLOW_UI_URL'
port = 'YOUR_PORT'
```

```
login = 'YOUR_AWS_ACCESS_KEY_ID'
password = 'YOUR_AWS_SECRET_ACCESS_KEY'
role_arn = urllib.parse.quote_plus('YOUR_EXECUTION_ROLE_ARN')
region_name = 'YOUR_REGION'

conn_string = '{0}://{1}:{2}@{3}:{4}?
role_arn={5}&region_name={6}'.format(conn_type, login, password, host, port,
 role_arn, region_name)
print(conn_string)
```

2. *Sostituisci i segnaposti in rosso.*
3. Eseguite lo script seguente per generare una stringa di connessione.

```
python3 mwaa_connection.py
```

Fase quattro: Aggiungere le variabili in Secrets Manager

La sezione seguente descrive come creare il segreto per una variabile in Secrets Manager.

Per creare il segreto

1. Apri la AWS Secrets Manager console.
2. Scegli Archivia un nuovo segreto.
3. Scegli Altro tipo di segreto.
4. Nel riquadro Specificare le coppie chiave/valore da memorizzare in questo riquadro segreto, scegli Testo normale.
5. Aggiungi il valore della variabile come testo semplice nel seguente formato.

```
"YOUR_VARIABLE_VALUE"
```

Ad esempio, per specificare un numero intero:

```
14
```

Ad esempio, per specificare una stringa:

```
"mystring"
```

6. Per la chiave di crittografia, scegli un'opzione AWS KMS chiave dall'elenco a discesa.
7. Inserisci un nome nel campo di testo per Nome segreto nel seguente formato.

   ```
   airflow/variables/YOUR_VARIABLE_NAME
   ```

   Per esempio:

   ```
   airflow/variables/test-variable
   ```
8. Seleziona Successivo.
9. Nella pagina Configura segreto, nel riquadro Nome e descrizione segreti, procedi come segue.

   a. Per Nome segreto, fornisci un nome per il tuo segreto.

   b. (Facoltativo) Per Descrizione, fornisci una descrizione del tuo segreto.

   Seleziona Successivo.
10. In Configura rotazione (facoltativo), lascia le opzioni predefinite e scegli Avanti.
11. Ripeti questi passaggi in Secrets Manager per tutte le variabili aggiuntive che desideri aggiungere.
12. Nella pagina di revisione, controlla il tuo segreto, quindi scegli Store.

Fase cinque: aggiungere la connessione in Secrets Manager

La sezione seguente descrive come creare il segreto per l'URI della stringa di connessione in Secrets Manager.

Per creare il segreto

1. Apri la AWS Secrets Manager console.
2. Scegli Archivia un nuovo segreto.
3. Scegli Altro tipo di segreto.
4. Nel riquadro Specificare le coppie chiave/valore da memorizzare in questo riquadro segreto, scegli Testo normale.
5. Aggiungi la stringa URI di connessione come testo semplice nel seguente formato.

```
YOUR_CONNECTION_URI_STRING
```

Per esempio:

```
mysql://288888a0-50a0-888-9a88-1a111aaa0000.a1.us-east-1.airflow.amazonaws.com
%2Fhome?role_arn=arn%3Aaws%3Aiam%3A%3A001122332255%3Arole%2Fservice-role
%2FAmazonMWAA-MyAirflowEnvironment-iAaaaA&region_name=us-east-1
```

> **Warning**
>
> Apache Airflow analizza ciascuno dei valori nella stringa di connessione. Non è necessario utilizzare virgolette singole o doppie, altrimenti analizzerà la connessione come una singola stringa.

6. Per la chiave di crittografia, scegli un'opzione AWS KMS chiave dall'elenco a discesa.
7. Inserisci un nome nel campo di testo per Nome segreto nel seguente formato.

```
airflow/connections/YOUR_CONNECTION_NAME
```

Per esempio:

```
airflow/connections/myconn
```

8. Seleziona Successivo.
9. Nella pagina Configura segreto, nel riquadro Nome e descrizione segreti, procedi come segue.
   a. Per Nome segreto, fornisci un nome per il tuo segreto.
   b. (Facoltativo) Per Descrizione, fornisci una descrizione del tuo segreto.

   Seleziona Successivo.
10. In Configura rotazione (facoltativo), lascia le opzioni predefinite e scegli Avanti.
11. Ripeti questi passaggi in Secrets Manager per tutte le variabili aggiuntive che desideri aggiungere.
12. Nella pagina di revisione, controlla il tuo segreto, quindi scegli Store.

Codice di esempio

- Scopri come utilizzare la chiave segreta per la connessione Apache Airflow (`myconn`) in questa pagina utilizzando il codice di esempio all'indirizzo. Utilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow
- Scopri come usare la chiave segreta per la variabile Apache Airflow (`test-variable`) in questa pagina utilizzando il codice di esempio all'indirizzo. Utilizzo di una chiave segreta inAWS Secrets Managerper una variabile Apache Airflow

Risorse

- Per ulteriori informazioni sulla configurazione dei segreti di Secrets Manager utilizzando la console e il AWS CLI, vedere Create a secret nella Guida per l'AWS Secrets Manager utente.
- Usa uno script Python per migrare un grande volume di variabili e connessioni Apache Airflow a Secrets Manager in Move your Apache Airflow connections and variable to. AWS Secrets Manager

Fasi successive

- Scopri come generare un token per accedere all'interfaccia utente di Apache Airflow. Accesso ad Apache Airflow

Gestione degli ambienti Amazon MWAA

La console Amazon Managed Workflows for Apache Airflow contiene opzioni integrate per configurare l'accesso privato o pubblico all'interfaccia utente di Apache Airflow. Contiene inoltre opzioni integrate per configurare le dimensioni dell'ambiente, quando scalare i dipendenti e opzioni di configurazione di Apache Airflow che consentono di ignorare le configurazioni di Apache Airflow che normalmente sono accessibili solo in. `airflow.cfg` Questa guida descrive come utilizzare queste configurazioni sulla console Amazon MWAA.

Argomenti

- Configurazione della classe di ambiente Amazon MWAA
- Configurazione della scalabilità automatica dei lavoratori Amazon MWAA
- Configurazione della scalabilità automatica del server Web Amazon MWAA
- Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA
- Aggiornamento della versione di Apache Airflow
- Utilizzo di uno script di avvio con Amazon MWAA

Configurazione della classe di ambiente Amazon MWAA

La classe di ambiente scelta per il tuo ambiente Amazon MWAA determina la dimensione dei AWS Fargate contenitori AWS gestiti su cui viene eseguito Celery Executor e del database di metadati AWS Amazon Aurora PostgreSQL gestito in cui gli scheduler Apache Airflow creano istanze di attività. Questa pagina descrive ogni classe di ambiente Amazon MWAA e i passaggi per aggiornare la classe di ambiente sulla console Amazon MWAA.

Sections

- Funzionalità ambientali
- Apache Airflow Scheduler

Funzionalità ambientali

La sezione seguente contiene le attività simultanee predefinite di Apache Airflow, la memoria ad accesso casuale (RAM) e le unità di elaborazione centralizzate virtuali (vCPU) per ogni classe di

ambiente. Le attività simultanee elencate presuppongono che la contemporanea delle attività non superi la capacità di Apache Airflow Worker nell'ambiente.

Nella tabella seguente, la capacità DAG si riferisce alle definizioni DAG, non alle esecuzioni, e presuppone che i DAG siano dinamici in un singolo file Python e scritti con le best practice di Apache Airflow.

Le esecuzioni delle attività dipendono dal numero di operazioni pianificate contemporaneamente e presuppone che il numero di esecuzioni DAG impostate per l'avvio alla stessa ora non superi il valore predefinito `max_dagruns_per_loop_to_schedule`, nonché le dimensioni e il numero di lavoratori, come descritto in questo argomento.

mw1.small

- Capacità fino a 50 DAG
- 5 attività simultanee (per impostazione predefinita)
- 1 vCPU
- 2 GB DI RAM

mw1.medium

- Capacità fino a 200 DAG
- 10 attività simultanee (per impostazione predefinita)
- 2 vCPU
- 4 GB DI RAM

mw1.large

- Capacità fino a 1000 DAG
- 20 attività simultanee (per impostazione predefinita)
- 4 vCPU
- 8 GB RAM

mw1.xlarge

- Capacità fino a 2000 DAG

- 40 attività simultanee (per impostazione predefinita)
- 8 vCPU
- 24 GB DI RAM

mw1.2xlarge

- Capacità fino a 4000 DAG
- 80 attività simultanee (per impostazione predefinita)
- 16 vCPU
- 48 GB DI RAM

È possibile utilizzarlo `celery.worker_autoscale` per aumentare le attività per lavoratore. Per ulteriori informazioni, consulta the section called “Esempio di utilizzo ad alte prestazioni”.

Apache Airflow Scheduler

La sezione seguente contiene le opzioni di pianificazione di Apache Airflow disponibili su Amazon MWAA e il modo in cui il numero di scheduler influisce sul numero di trigger.

In Apache Airflow, un trigger gestisce le attività che rimanda finché non vengono soddisfatte determinate condizioni specificate utilizzando un trigger. In Amazon MWAA, il trigger viene eseguito insieme allo scheduler per la stessa attività di Fargate. L'aumento del numero di pianificatori aumenta di conseguenza il numero di trigger disponibili, ottimizzando il modo in cui l'ambiente gestisce le attività differite. Ciò garantisce una gestione efficiente delle attività, pianificandone tempestivamente l'esecuzione quando le condizioni sono soddisfatte.

Apache Airflow v2

- v2 - Accetta tra a. 2 5 L'impostazione predefinita è 2.

Configurazione della scalabilità automatica dei lavoratori Amazon MWAA

Il meccanismo di scalabilità automatica aumenta automaticamente il numero di lavoratori Apache Airflow in risposta alle attività in esecuzione e in coda nel tuo ambiente Amazon Managed Workflows

for Apache Airflow e elimina lavoratori aggiuntivi quando non ci sono più attività in coda o in esecuzione. Questa pagina descrive come configurare la scalabilità automatica specificando il numero massimo di worker Apache Airflow eseguiti nel tuo ambiente utilizzando la console Amazon MWAA.

> **Note**
>
> Amazon MWAA utilizza i parametri di Apache Airflow per determinare quando sono necessari altri lavoratori Celery Executor e, se necessario, aumenta il numero di lavoratori Fargate fino al valore specificato da. `max-workers` Man mano che i lavoratori aggiuntivi completano il lavoro e il carico di lavoro diminuisce, Amazon MWAA li rimuove, tornando così al valore impostato da. `min-workers`
> Se i lavoratori intraprendono nuove attività durante il downscaling, Amazon MWAA conserva la risorsa Fargate e non rimuove il lavoratore. Per ulteriori informazioni, consulta Come funziona la scalabilità automatica di Amazon MWAA.

Sections

- Come funziona la scalabilità dei lavoratori
- Utilizzo della console Amazon MWAA
- Esempio di utilizzo ad alte prestazioni
- Risoluzione dei problemi relativi alle attività bloccate nello stato di esecuzione
- Fasi successive

Come funziona la scalabilità dei lavoratori

Usi `RunningTasks` e `QueuedTasks` parametri di Amazon MWAA, dove (attività in esecuzione + attività in coda)/(attività per lavoratore) = (lavoratori richiesti). Se il numero richiesto di lavoratori è superiore al numero attuale di lavoratori, Amazon MWAA aggiungerà i container per lavoratori Fargate a quel valore, fino al valore massimo specificato da. `max-workers`

Man mano che il carico di lavoro diminuisce e la somma dei `QueuedTasks` parametri `RunningTasks` e si riduce, Amazon MWAA richiede a Fargate di ridurre il numero di lavoratori per l'ambiente. Tutti i lavoratori che stanno ancora completando il lavoro rimangono protetti durante il downscaling fino al completamento del lavoro. A seconda del carico di lavoro, le attività possono essere messe in coda mentre i lavoratori effettuano la scalata.

Utilizzo della console Amazon MWAA

Puoi scegliere il numero massimo di lavoratori che possono essere eseguiti contemporaneamente nel tuo ambiente sulla console Amazon MWAA. Per impostazione predefinita, puoi specificare un valore massimo fino a 25.

Per configurare il numero di lavoratori

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.
4. Seleziona Successivo.
5. Nel riquadro Classe Ambiente, inserisci un valore in Numero massimo di lavoratori.
6. Selezionare Salva.

> **Note**
>
> Possono essere necessari alcuni minuti prima che le modifiche abbiano effetto sull'ambiente.

Esempio di utilizzo ad alte prestazioni

La sezione seguente descrive il tipo di configurazioni che è possibile utilizzare per abilitare alte prestazioni e parallelismo in un ambiente.

Apache Airflow locale

In genere, in una piattaforma Apache Airflow on-premise, è necessario configurare il parallelismo delle attività, la scalabilità automatica e le impostazioni di concorrenza nel file: `airflow.cfg`

- `core.parallelism`— Il numero massimo di istanze di attività che possono essere eseguite contemporaneamente per scheduler.
- `core.dag_concurrency`— La concorrenza massima per i DAG (non per i lavoratori).
- `celery.worker_autoscale`— Il numero massimo e minimo di attività che possono essere eseguite contemporaneamente su qualsiasi lavoratore.

Ad esempio, se `core.parallelism` fosse impostato su `100` e `core.dag_concurrency` fosse impostato su7, sarebbe comunque possibile eseguire un totale di `14` attività contemporaneamente solo se si disponesse di 2 DAG. Tuttavia, ogni DAG è impostato per eseguire solo sette attività contemporaneamente (in`core.dag_concurrency`), anche se il parallelismo complessivo è impostato su (in). `100` `core.parallelism`

In un ambiente Amazon MWAA

In un ambiente Amazon MWAA, puoi configurare queste impostazioni direttamente sulla console Amazon MWAA utilizzando Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAAConfigurazione della classe di ambiente Amazon MWAA, e il meccanismo di scalabilità automatica Maximum worker count. Sebbene non `core.dag_concurrency` sia disponibile nell'elenco a discesa come opzione di configurazione Apache Airflow sulla console Amazon MWAA, puoi aggiungerla come opzione di configurazione Apache Airflow personalizzata.

Supponiamo che quando hai creato il tuo ambiente, hai scelto le seguenti impostazioni:

1. La classe di ambiente mw1.small che controlla il numero massimo di attività simultanee che ogni lavoratore può eseguire per impostazione predefinita e la vCPU dei contenitori.
2. L'impostazione predefinita di **`10`** Workers in Maximum worker count.
3. Un'opzione di configurazione di Apache Airflow per `celery.worker_autoscale` più `5,5` attività per lavoratore.

Ciò significa che puoi eseguire 50 attività simultanee nel tuo ambiente. Tutte le attività oltre 50 verranno messe in coda e attenderanno il completamento delle attività in esecuzione.

Esegui più attività simultanee. È possibile modificare l'ambiente per eseguire più attività contemporaneamente utilizzando le seguenti configurazioni:

1. Aumenta il numero massimo di attività simultanee che ogni lavoratore può eseguire per impostazione predefinita e la vCPU dei contenitori scegliendo `mw1.medium` la classe di ambiente (10 attività simultanee per impostazione predefinita).
2. Aggiungi `celery.worker_autoscale` come opzione di configurazione Apache Airflow.
3. Aumenta il numero massimo di lavoratori. In questo esempio, l'aumento del numero massimo di lavoratori da `10` a `20` raddoppierebbe il numero di attività simultanee che l'ambiente può eseguire.

Specificare il numero minimo di lavoratori. È inoltre possibile specificare il numero minimo e massimo di Apache Airflow Workers eseguiti nel proprio ambiente utilizzando AWS Command Line Interface ()AWS CLI. Per esempio:

```
aws mwaa update-environment --max-workers 10 --min-workers 10 --
name YOUR_ENVIRONMENT_NAME
```

Per ulteriori informazioni, consultate il comando update-environment in. AWS CLI

Risoluzione dei problemi relativi alle attività bloccate nello stato di esecuzione

In rari casi, Apache Airflow potrebbe pensare che ci siano attività ancora in esecuzione. Per risolvere questo problema, è necessario cancellare l'attività bloccata nell'interfaccia utente di Apache Airflow. Per ulteriori informazioni, consulta l'argomento sulla risoluzione dei problemi. Vedo che le mie attività sono bloccate o non vengono completate

Fasi successive

- Scopri di più sulle best practice che consigliamo per ottimizzare le prestazioni del tuo ambienteOttimizzazione delle prestazioni per Apache Airflow su Amazon MWAA.

Configurazione della scalabilità automatica del server Web Amazon MWAA

Per gli ambienti che eseguono Apache Airflow Apache Airflow v2.2.2 e versioni successive, Amazon MWAA ridimensiona dinamicamente i server Web per gestire carichi di lavoro fluttuanti, il che a sua volta previene problemi di prestazioni durante i picchi di carico. Ridimensionando automaticamente il numero di server Web in base all'utilizzo della CPU e al numero di connessioni attive, Amazon MWAA garantisce che l'ambiente Apache Airflow possa soddisfare senza problemi l'aumento della domanda, sia che provenga dalle richieste API REST, dall'utilizzo della CLI o da più utenti simultanei dell'interfaccia utente Apache Airflow.

Sections

- Come funziona la scalabilità dei server web
- Utilizzo della console Amazon MWAA

Come funziona la scalabilità dei server web

Amazon MWAA utilizza la metrica del contenitore e la metrica del bilanciamento del carico per determinare se è necessario scalare i server Web in base alla quantità di traffico. `CPUUtilization` `ActiveConnectionCount` Se `CPUUtilization` è superiore a 70 o `ActiveConnectionCount` superiore a 15, Amazon MWAA aggiungerà altri contenitori di server Web Fargate fino al valore massimo specificato da. `MaxWebservers`

Man mano che il traffico diminuisce `CPUUtilization` e `ActiveConnectionCount` i valori e diminuiscono, Amazon MWAA richiede a Fargate di ridimensionare i contenitori del server Web per l'ambiente al valore minimo impostato da. `MinimumWebservers`

Utilizzo della console Amazon MWAA

Puoi scegliere il numero di server Web che possono essere eseguiti contemporaneamente nel tuo ambiente sulla console Amazon MWAA. Per impostazione predefinita, il numero minimo di server Web è due e il numero massimo di server Web è cinque.

Per configurare il numero di server Web

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.
4. Seleziona Successivo.
5. Nel riquadro della classe Ambiente, immettete un valore in Numero massimo di server web.
6. Quindi, immettete un valore in Numero minimo di server web.
7. Selezionare Salva.

> **Note**
>
> Possono essere necessari alcuni minuti prima che le modifiche abbiano effetto sull'ambiente.

Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA

Le opzioni di configurazione di Apache Airflow possono essere collegate al tuo ambiente Amazon Managed Workflows for Apache Airflow come variabili di ambiente. Puoi scegliere dall'elenco a discesa suggerito o specificare opzioni di configurazione personalizzate per la tua versione di Apache Airflow sulla console Amazon MWAA. Questa pagina descrive le opzioni di configurazione di Apache Airflow disponibili e come utilizzarle per sovrascrivere le impostazioni di configurazione di Apache Airflow nel tuo ambiente.

Indice

- Prerequisiti
- Come funziona
- Utilizzo delle opzioni di configurazione per caricare i plugin in Apache Airflow v2
- Panoramica delle opzioni di configurazione
  - Opzioni di configurazione di Apache Airflow
  - Riferimento Apache Airflow
  - Utilizzo della console Amazon MWAA
- Informazioni di riferimento sulla configurazione
  - Configurazioni e-mail
  - Configurazioni delle attività
  - Configurazioni dello scheduler
  - Configurazioni dei lavoratori
  - Configurazioni del server Web
  - Configurazioni Triggerer
- Esempi e codice di esempio
  - Esempio DAG
  - Esempio di impostazioni di notifica e-mail
- Fasi successive

Prerequisiti

Avrai bisogno di quanto segue prima di completare i passaggi di questa pagina.

- Autorizzazioni: al tuo AWS account deve essere stato concesso dall'amministratore l'accesso alla politica di controllo degli FullConsoleaccessi di AmazonMWAA per il tuo ambiente. Inoltre, il tuo ambiente Amazon MWAA deve essere autorizzato dal tuo ruolo di esecuzione ad accedere alle AWS risorse utilizzate dal tuo ambiente.
- Accesso: se è necessario accedere agli archivi pubblici per installare le dipendenze direttamente sul server Web, l'ambiente deve essere configurato con l'accesso al server Web di rete pubblica. Per ulteriori informazioni, consulta the section called “Modalità di accesso Apache Airflow”.
- Configurazione Amazon S3 : il bucket Amazon S3 utilizzato per archiviare i DAG`plugins.zip`, i plug-in personalizzati e le dipendenze `requirements.txt` Python deve essere configurato con Public Access Blocked e Versioning Enabled.

Come funziona

Quando crei un ambiente, Amazon MWAA allega le impostazioni di configurazione specificate nella console Amazon MWAA nelle opzioni di configurazione Airflow come variabili di ambiente al contenitore per il tuo ambiente. AWS Fargate Se utilizzi un'impostazione con lo stesso nome in`airflow.cfg`, le opzioni specificate nella console Amazon MWAA sostituiscono i valori in. `airflow.cfg`

Sebbene per impostazione predefinita non le esponiamo `airflow.cfg` nell'interfaccia utente Apache Airflow di un ambiente Amazon MWAA, puoi modificare le opzioni di configurazione di Apache Airflow direttamente sulla console Amazon MWAA, inclusa l'impostazione per esporre le configurazioni. `webserver.expose_config`

Utilizzo delle opzioni di configurazione per caricare i plugin in Apache Airflow v2

Per impostazione predefinita, in Apache Airflow v2, i plugin sono configurati per essere caricati «pigramente» utilizzando l'impostazione. `core.lazy_load_plugins : True` Se utilizzi plug-in personalizzati in Apache Airflow v2, devi aggiungere un'opzione di configurazione di Apache Airflow per caricare `core.lazy_load_plugins : False` i plug-in all'inizio di ogni processo Airflow per sovrascrivere l'impostazione predefinita.

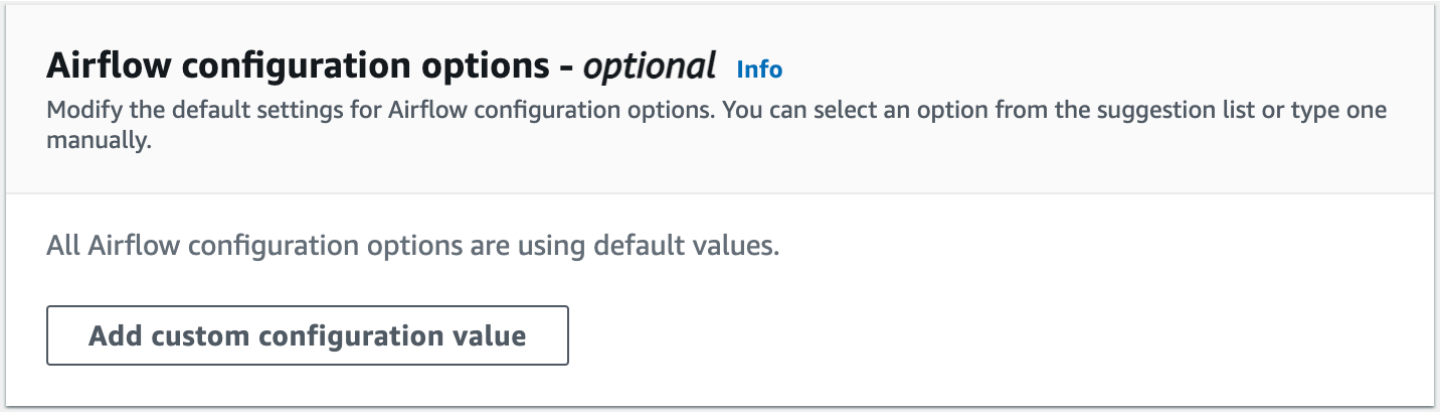
Panoramica delle opzioni di configurazione

Quando aggiungi una configurazione sulla console Amazon MWAA, Amazon MWAA scrive la configurazione come variabile di ambiente.

- Opzioni elencate. Puoi scegliere una delle impostazioni di configurazione disponibili per la tua versione di Apache Airflow nell'elenco a discesa. Ad esempio,:. `dag_concurrency 16` L'impostazione di configurazione viene tradotta nel contenitore Fargate dell'ambiente come `AIRFLOW__CORE__DAG_CONCURRENCY : 16`
- Opzioni personalizzate. Puoi anche specificare opzioni di configurazione Airflow che non sono elencate per la tua versione di Apache Airflow nell'elenco a discesa. Ad esempio,:. `foo.user YOUR_USER_NAME` L'impostazione di configurazione viene tradotta nel contenitore Fargate dell'ambiente come `AIRFLOW__FOO__USER : YOUR_USER_NAME`

Opzioni di configurazione di Apache Airflow

L'immagine seguente mostra dove è possibile personalizzare le opzioni di configurazione di Apache Airflow sulla console Amazon MWAA.



Riferimento Apache Airflow

Per un elenco delle opzioni di configurazione supportate da Apache Airflow, consulta Configuration Reference nella guida di riferimento di Apache Airflow. Per visualizzare le opzioni per la versione di Apache Airflow in esecuzione su Amazon MWAA, seleziona la versione dall'elenco a discesa.

Utilizzo della console Amazon MWAA

La procedura seguente illustra i passaggi per aggiungere un'opzione di configurazione Airflow al tuo ambiente.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.
4. Seleziona Successivo.
5. Scegli Aggiungi configurazione personalizzata nel riquadro delle opzioni di configurazione Airflow.
6. Scegli una configurazione dall'elenco a discesa e inserisci un valore, oppure digita una configurazione personalizzata e inserisci un valore.
7. Scegli Aggiungi configurazione personalizzata per ogni configurazione che desideri aggiungere.
8. Selezionare Salva.

Informazioni di riferimento sulla configurazione

La sezione seguente contiene l'elenco delle configurazioni Apache Airflow disponibili nell'elenco a discesa sulla console Amazon MWAA.

Configurazioni e-mail

L'elenco seguente mostra le opzioni di configurazione delle notifiche e-mail Airflow disponibili su Amazon MWAA.

Si consiglia di utilizzare la porta 587 per il traffico SMTP. Per impostazione predefinita, AWS blocca il traffico SMTP in uscita sulla porta 25 di tutte le istanze Amazon EC2. Se desideri inviare traffico in uscita sulla porta 25, puoi richiedere la rimozione di questa restrizione.

Apache Airflow v2

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
v2	email.email_backend	L'utilità Apache Airflow utilizzata per le notifiche e-mail in email_backend.	airflow.utils.email.send_email_smtp
v2	smtp.smtp_host	Il nome del server in uscita utilizzato per	localhost

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
		l'indirizzo e-mail in smtp_host.	
v2	smtp.smtp_starttls	Transport Layer Security (TLS) viene utilizzato per crittografare le e-mail su Internet in smtp_starttls.	False
v2	smtp.smtp_ssl	Secure Sockets Layer (SSL) viene utilizzato per connettere il server e il client di posta elettronica in smtp_ssl.	True
v2	porta smtp.smtp_	La porta TCP (Transmission Control Protocol) designata al server in smtp_port.	587
v2	smtp.smtp_mail_from	L'indirizzo e-mail in uscita in smtp_mail_from.	myemail@domain.com

Configurazioni delle attività

L'elenco seguente mostra le configurazioni disponibili nell'elenco a discesa per le attività Airflow su Amazon MWAA.

Apache Airflow v2

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
v2	core.default_task_retries	Il numero di volte in cui riprovare un'attività di Apache Airflow in default_task_retries.	3
v2	core.parallelismo	Il numero massimo di istanze di attività che possono essere eseguite contemporaneamente nell'intero ambiente in parallelo (parallelismo).	40

Configurazioni dello scheduler

L'elenco seguente mostra le configurazioni dello scheduler Apache Airflow disponibili nell'elenco a discesa su Amazon MWAA.

Apache Airflow v2

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
v2	scheduler.catchup_by_default	Indica allo scheduler di creare un'esecuzione DAG per «recuperare» l'intervallo di tempo specifico in catchup_by_default.	False

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
v2	scheduler.scheduler_zombie_task_threshold	Indica allo scheduler se contrassegnare l'istanza dell'attività come non riuscita e riprogrammarla in scheduler_zombie_task_threshold.	300

Configurazioni dei lavoratori

L'elenco seguente mostra le configurazioni Airflow Worker disponibili nell'elenco a discesa su Amazon MWAA.

Apache Airflow v2

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
v2	celery.worker_autoscale	Il numero massimo e minimo di attività che possono essere eseguite contemporaneamente su qualsiasi lavoratore utilizzando Celery Executor in worker_autoscale. Il valore deve essere separato da virgole nel seguente ordine:. `max_concurrency,min_concurrency`	16,12

Configurazioni del server Web

L'elenco seguente mostra le configurazioni del server Web Airflow disponibili nell'elenco a discesa su Amazon MWAA.

Apache Airflow v2

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
v2	webserver.default_ui_timezone	L'impostazione datetime predefinita dell'interfaccia utente di Apache Airflow in default_ui_timezone. **Note** L'impostazione dell'`default_ui_timezone` opzione non modifica il fuso orario in cui è pianificata l'esecuzione dei DAG. Per modificare il fuso orario dei tuoi DAG, puoi utilizzare un plug-in personalizzato. Per ulteriori informazioni,	America/New_York

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
		consulta the section called “Modifica del fuso orario di un DAG”.	

Configurazioni Triggerer

L'elenco seguente mostra le configurazioni dei trigger di Apache Airflow disponibili su Amazon MWAA.

Apache Airflow v2

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
v2.7	mwaa.triggerer_enabled	Utilizzato per attivare e disattivare il trigger su Amazon MWAA. Per impostazione predefinita, questo valore è impostato su `True`. Se impostato su`False`, Amazon MWAA non avvierà alcun processo di attivazione sugli scheduler.	True
v2.7	triggerer.default_capacity	Definisce il numero di trigger che ogni trigger può eseguire in parallelo. Su	125

Versione Airflow	Opzione di configurazione del flusso d'aria	Descrizione	Valore di esempio
		Amazon MWAA, questa capacità è impostata per ogni trigger e per ogni scheduler poiché entrambi i componenti funzionano insieme. L'impostazione predefinita per scheduler è`60`,,`125`, e `1000` per istanze piccole `250``500`, medie e grandi, xlarge e 2xlarge, rispettivamente.	

Esempi e codice di esempio

Esempio DAG

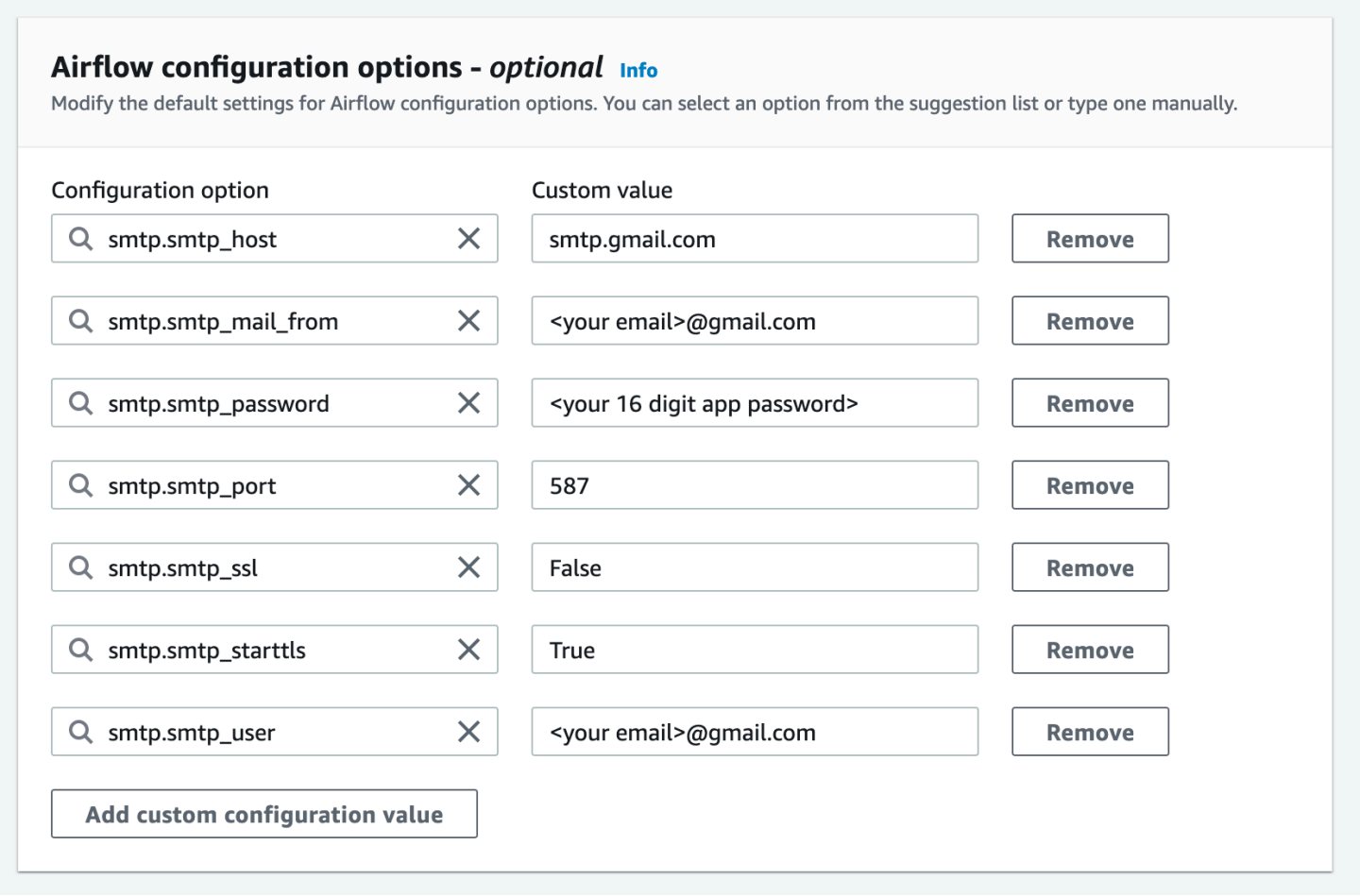
È possibile utilizzare il seguente DAG per stampare le opzioni di configurazione di `email_backend` Apache Airflow. Per eseguirlo in risposta agli eventi di Amazon MWAA, copia il codice nella cartella DAg del tuo ambiente sul bucket di storage Amazon S3.

```
from airflow.decorators import dag
from datetime import datetime

def print_var(**kwargs):
  email_backend = kwargs['conf'].get(section='email', key='email_backend')
  print("email_backend")
  return email_backend

@dag(
  dag_id="print_env_variable_example",
```

```
  schedule_interval=None,
  start_date=datetime(yyyy, m, d),
  catchup=False,
)
def print_variable_dag():
  email_backend_test = PythonOperator(
  task_id="email_backend_test",
  python_callable=print_var,
  provide_context=True
)

print_variable_test = print_variable_dag()
```

Esempio di impostazioni di notifica e-mail

Le seguenti opzioni di configurazione di Apache Airflow possono essere utilizzate per un account di posta elettronica Gmail.com utilizzando una password dell'app. Per ulteriori informazioni, consulta Accedere utilizzando le password delle app nella guida di riferimento dell'assistenza di Gmail.



Fasi successive

- Scopri come caricare la tua cartella DAG nel tuo bucket Amazon S3 in. Aggiunta o aggiornamento di DAG

Aggiornamento della versione di Apache Airflow

Amazon MWAA supporta aggiornamenti di versioni minori. Ciò significa che puoi aggiornare il tuo ambiente dalla versione a. `x.4.z` `x.5.z` Per eseguire un aggiornamento della versione principale, ad esempio dalla versione **1**`.y.z` a**2**`.y.z`, è necessario creare un nuovo ambiente e migrare le risorse. Per ulteriori informazioni sull'aggiornamento a una nuova versione principale di Apache Airflow, consulta Migrazione a un nuovo ambiente Amazon MWAA nella Amazon MWAA Migration Guide.

Durante il processo di aggiornamento, Amazon MWAA acquisisce un'istantanea dei metadati dell'ambiente, aggiorna i lavoratori, gli scheduler e il server Web alla nuova versione di Apache Airflow e infine ripristina il database di metadati utilizzando lo snapshot.

> **Note**
>
> Non puoi effettuare il downgrade della versione di Apache Airflow per il tuo ambiente.

Prima di eseguire l'aggiornamento, assicurati che i DAG e le altre risorse del flusso di lavoro siano compatibili con la nuova versione di Apache Airflow a cui stai effettuando l'aggiornamento. Se utilizzi `requirements.txt` a per gestire le dipendenze, devi anche assicurarti che le dipendenze specificate nei requisiti siano compatibili con la nuova versione.

Argomenti

- Aggiorna le risorse del tuo flusso di lavoro
- Specificare la nuova versione

Aggiorna le risorse del tuo flusso di lavoro

Ogni volta che modifichi le versioni di Apache Airflow, assicurati di fare riferimento all'`--constraint`URL corretto nel tuo. `requirements.txt`

> **Warning**
>
> La specificazione di requisiti incompatibili con la versione di Apache Airflow di destinazione durante un aggiornamento potrebbe comportare un lungo processo di rollback alla versione precedente di Apache Airflow con la versione dei requisiti precedente.

Per migrare le risorse del flusso di lavoro

1. Crea un fork del aws-mwaa-local-runnerrepository e clona una copia del runner locale di Amazon MWAA.
2. Effettua il checkout nel ramo del aws-mwaa-local-runner repository corrispondente alla versione a cui stai effettuando l'aggiornamento.
3. Usa lo strumento CLI Amazon MWAA local runner per creare l'immagine Docker ed eseguire Apache Airflow localmente. Per ulteriori informazioni, consulta il file README del runner locale nel repository. GitHub
4. Per aggiornare le tue`requirements.txt`, segui le best practice consigliate nella sezione Managing Python dependencies, nella Amazon MWAA User Guide.
5. (Facoltativo) Per accelerare il processo di aggiornamento, pulisci il database dei metadati dell'ambiente. L'aggiornamento degli ambienti con una grande quantità di metadati può richiedere molto più tempo.
6. Dopo aver testato con successo le risorse del flusso di lavoro, copia i DAG e i plug-in nel bucket Amazon S3 del tuo ambiente. `requirements.txt`

Ora sei pronto per modificare l'ambiente, specificare una nuova versione di Apache Airflow e avviare la procedura di aggiornamento.

Specificare la nuova versione

Dopo aver completato l'aggiornamento delle risorse del flusso di lavoro per garantire la compatibilità con la nuova versione di Apache Airflow, procedi come segue per modificare i dettagli dell'ambiente e specificare la versione di Apache Airflow a cui desideri eseguire l'aggiornamento.

> **Note**
>
> Quando si esegue un aggiornamento, tutte le attività attualmente in esecuzione nell'ambiente vengono terminate durante la procedura. La procedura di aggiornamento può richiedere fino a due ore, durante le quali l'ambiente non sarà disponibile.

Per specificare una nuova versione utilizzando la console

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Dall'elenco Ambienti, scegli l'ambiente che desideri aggiornare.
3. Nella pagina dell'ambiente, scegli Modifica per modificare l'ambiente.
4. Nella sezione Dettagli sull'ambiente, per la versione Airflow, scegli il nuovo numero di versione di Apache Airflow a cui desideri aggiornare l'ambiente dall'elenco a discesa.
5. Scegli Avanti finché non ti trovi nella pagina Rivedi e salva.
6. Nella pagina Rivedi e salva, rivedi le modifiche, quindi scegli Salva.

Quando si applicano le modifiche, l'ambiente avvia la procedura di aggiornamento. Durante questo periodo, lo stato dell'ambiente indica le azioni intraprese da Amazon MWAA e l'esito positivo della procedura.

In uno scenario di aggiornamento riuscito, lo stato verrà visualizzato `UPDATING` non `CREATING_SNAPSHOT` appena Amazon MWAA acquisirà un backup dei metadati. Infine, lo stato tornerà prima al termine della procedura e poi a quello `UPDATING` in `AVAILABLE` cui è stata completata.

Se l'aggiornamento dell'ambiente non riesce, verrà visualizzato lo stato dell'ambiente`ROLLING_BACK`. Se il rollback ha esito positivo, viene innanzitutto visualizzato lo stato`UPDATE_FAILED`, a indicare che l'aggiornamento non è riuscito ma che l'ambiente è disponibile. Se il rollback fallisce, verrà visualizzato lo stato`UNAVAILABLE`, a indicare che non è possibile accedere all'ambiente.

Utilizzo di uno script di avvio con Amazon MWAA

Uno script di avvio è uno script shell (`.sh`) ospitato nel bucket Amazon S3 del tuo ambiente in modo simile ai tuoi DAG, requisiti e plugin. Amazon MWAA esegue questo script durante l'avvio su ogni singolo componente di Apache Airflow (worker, scheduler e server Web) prima di installare i requisiti

e inizializzare il processo Apache Airflow. Utilizza uno script di avvio per effettuare le seguenti operazioni:

- Runtime di installazione: installa i runtime Linux richiesti dai flussi di lavoro e dalle connessioni.
- Configurazione delle variabili di ambiente: imposta le variabili di ambiente per ogni componente di Apache Airflow. Sovrascrivi variabili comuni come`PATH`, `PYTHONPATH` e. `LD_LIBRARY_PATH`
- Gestisci chiavi e token: trasferisci i token di accesso per gli archivi personalizzati `requirements.txt` e configura le chiavi di sicurezza.

I seguenti argomenti descrivono come configurare uno script di avvio per installare i runtime Linux, impostare le variabili di ambiente e risolvere i problemi correlati utilizzando Logs. CloudWatch

Argomenti

- Configurare uno script di avvio
- Installa i runtime Linux utilizzando uno script di avvio
- Imposta le variabili di ambiente utilizzando uno script di avvio

Configurare uno script di avvio

Per utilizzare uno script di avvio con il tuo ambiente Amazon MWAA esistente, carica un `.sh` file nel bucket Amazon S3 del tuo ambiente. Quindi, per associare lo script all'ambiente, specifica quanto segue nei dettagli dell'ambiente:

- Il percorso URL di Amazon S3 allo script: il percorso relativo allo script ospitato nel tuo bucket, ad esempio, `s3://mwaa-environment/startup.sh`
- L'ID della versione Amazon S3 dello script: la versione dello script di shell di avvio nel tuo bucket Amazon S3. Devi specificare l'ID della versione che Amazon S3 assegna al file ogni volta che aggiorni lo script. Gli ID di versione sono stringhe opache Unicode, con codifica UTF-8, pronte per l'URL e lunghe non più di 1.024 byte, ad esempio. `3sL4kqtJlcpXroDTDmJ+rmSpXd3dIbrHY+MTRCxf3vjVBH40Nr8X8gdRQBpUMLUo`

Per completare i passaggi descritti in questa sezione, utilizzate lo script di esempio seguente. Lo script restituisce il valore assegnato a`MWAA_AIRFLOW_COMPONENT`. Questa variabile di ambiente identifica ogni componente di Apache Airflow su cui viene eseguito lo script.

Copia il codice e salvalo localmente come. `startup.sh`

```
#!/bin/sh

echo "Printing Apache Airflow component"
echo $MWAA_AIRFLOW_COMPONENT
```

Quindi, carica lo script nel tuo bucket Amazon S3.

AWS Management Console

Per caricare uno script di shell (console)

1. Accedere alla AWS Management Console e aprire la console Amazon S3 all'indirizzo https://console.aws.amazon.com/s3/.
2. Dall'elenco dei bucket, scegli il nome del bucket associato al tuo ambiente.
3. Nella scheda Objects (Oggetti), scegliere Upload (Carica).
4. Nella pagina di caricamento, trascina e rilascia lo script di shell che hai creato.
5. Scegli Carica.

Lo script viene visualizzato nell'elenco degli oggetti. Amazon S3 crea un nuovo ID di versione per il file. Se aggiorni lo script e lo carichi di nuovo utilizzando lo stesso nome di file, al file viene assegnato un nuovo ID di versione.

AWS CLI

Per creare e caricare uno script di shell (CLI)

1. Apri un nuovo prompt dei comandi ed esegui il comando Amazon `ls` S3 per elencare e identificare il bucket associato al tuo ambiente.

   ```
   $ aws s3 ls
   ```

2. Passa alla cartella in cui hai salvato lo script di shell. Utilizzalo `cp` in una nuova finestra di richiesta per caricare lo script nel tuo bucket. Sostituisci *`your-s3-bucket`* con le tue informazioni.

   ```
   $ aws s3 cp startup.sh s3://your-s3-bucket/startup.sh
   ```

   In caso di successo, Amazon S3 restituisce il percorso URL dell'oggetto:

```
upload: ./startup.sh to s3://your-s3-bucket/startup.sh
```

3. Usa il seguente comando per recuperare l'ID della versione più recente dello script.

```
$ aws s3api list-object-versions --bucket your-s3-bucket --prefix startup --
query 'Versions[?IsLatest].[VersionId]' --output text
```

```
BbdVMmBRjtestta1EsVnbybZp1Wqh1J4
```

Questo ID di versione viene specificato quando si associa lo script a un ambiente.

Ora associate lo script al vostro ambiente.

AWS Management Console

Per associare lo script a un ambiente (console)

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Seleziona la riga per l'ambiente che desideri aggiornare, quindi scegli Modifica.
3. Nella pagina Specificare i dettagli, per il file di script di avvio, facoltativo, inserisci l'URL di Amazon S3 per lo script, ad esempio:. `s3://your-mwaa-bucket/startup-sh.`
4. Scegli la versione più recente dall'elenco a discesa o sfoglia S3 per trovare lo script.
5. Scegli Avanti, quindi procedi alla pagina Rivedi e salva.
6. Rivedi le modifiche, quindi scegli Salva.

Gli aggiornamenti dell'ambiente possono richiedere dai 10 ai 30 minuti. Amazon MWAA esegue lo script di avvio al riavvio di ogni componente dell'ambiente.

AWS CLI

Per associare lo script a un ambiente (CLI)

- Apri un prompt dei comandi e utilizzalo `update-environment` per specificare l'URL e l'ID di versione di Amazon S3 per lo script.

```
$ aws mwaa update-environment \
```

```
    --name your-mwaa-environment \
    --startup-script-s3-path startup.sh \
    --startup-script-s3-object-version BbdVMmBRjtestta1EsVnbybZp1Wqh1J4
```

In caso di successo, Amazon MWAA restituisce l'Amazon Resource Name (ARN) per l'ambiente:

```
arn:aws::airflow:us-west-2:123456789012:environment/your-mwaa-environment
```

L'aggiornamento dell'ambiente può richiedere dai 10 ai 30 minuti. Amazon MWAA esegue lo script di avvio al riavvio di ogni componente dell'ambiente.

Infine, recupera gli eventi di registro per verificare che lo script funzioni come previsto. Quando attivi la registrazione per ogni componente di Apache Airflow, Amazon MWAA crea un nuovo gruppo di log e un nuovo flusso di log. Per ulteriori informazioni, consulta Tipi di log di Apache Airflow.

AWS Management Console

Per controllare il flusso di log di Apache Airflow (console)

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli il tuo ambiente.
3. Nel riquadro Monitoraggio, scegli il gruppo di log per il quale desideri visualizzare i log, ad esempio il gruppo di log di Airflow scheduler.
4. Nella CloudWatch console, dall'elenco Log stream, scegliete uno stream con il seguente prefisso:. `startup_script_exection_ip`
5. Nel riquadro Registra eventi, vedrai l'output del comando che stampa il valore per. `MWAA_AIRFLOW_COMPONENT` Ad esempio, per i registri dello scheduler, otterrai quanto segue:

```
Printing Apache Airflow component
scheduler
Finished running startup script. Execution time: 0.004s.
Running verification
Verification completed
```

È possibile ripetere i passaggi precedenti per visualizzare i registri dei lavoratori e dei server Web.

Installa i runtime Linux utilizzando uno script di avvio

Utilizza uno script di avvio per aggiornare il sistema operativo di un componente Apache Airflow e installa librerie di runtime aggiuntive da utilizzare con i flussi di lavoro. Ad esempio, lo script seguente viene eseguito `yum update` per aggiornare il sistema operativo.

Quando si esegue `yum update` uno script di avvio, è necessario escludere l'utilizzo di Python `--exclude=python*` come mostrato nell'esempio. Affinché il tuo ambiente funzioni, Amazon MWAA installa una versione specifica di Python compatibile con il tuo ambiente. Pertanto, non è possibile aggiornare la versione Python dell'ambiente utilizzando uno script di avvio.

```
#!/bin/sh

echo "Updating operating system"
sudo yum update -y --exclude=python*
```

Per installare i runtime su un componente specifico di Apache Airflow, usa le istruzioni condizionali `MWAA_AIRFLOW_COMPONENT` e `eif`. `fi` Questo esempio esegue un singolo comando per installare la `libaio` libreria sullo scheduler e sul worker, ma non sul server web.

> **Important**
>
> - Se è stato configurato un server Web privato, è necessario utilizzare la seguente condizione o fornire tutti i file di installazione localmente per evitare i timeout di installazione.
> - `sudo`Da utilizzare per eseguire operazioni che richiedono privilegi amministrativi.

```
#!/bin/sh

if [[ "${MWAA_AIRFLOW_COMPONENT}" != "webserver" ]]
then
     sudo yum -y install libaio
fi
```

Puoi usare uno script di avvio per controllare la versione di Python.

```
#!/bin/sh

export PYTHON_VERSION_CHECK=`python -c 'import sys; version=sys.version_info[:3];
 print("{0}.{1}.{2}".format(*version))'`
echo "Python version is $PYTHON_VERSION_CHECK"
```

Amazon MWAA non supporta l'override della versione Python predefinita, poiché ciò potrebbe causare incompatibilità con le librerie Apache Airflow installate.

Imposta le variabili di ambiente utilizzando uno script di avvio

Usa gli script di avvio per impostare le variabili di ambiente e modificare le configurazioni di Apache Airflow. Quanto segue definisce una nuova variabile,. `ENVIRONMENT_STAGE` È possibile fare riferimento a questa variabile in un DAG o nei moduli personalizzati.

```
#!/bin/sh

export ENVIRONMENT_STAGE="development"
echo "$ENVIRONMENT_STAGE"
```

Utilizzate gli script di avvio per sovrascrivere le variabili di sistema o di Apache Airflow più comuni. Ad esempio, si imposta `LD_LIBRARY_PATH` di indicare a Python di cercare i binari nel percorso specificato. Ciò consente di fornire file binari personalizzati per i flussi di lavoro utilizzando i plug-in:

```
#!/bin/sh

export LD_LIBRARY_PATH=/usr/local/airflow/plugins/your-custom-binary
```

Variabili d'ambiente riservate

Amazon MWAA riserva un set di variabili ambientali critiche. Se sovrascrivi una variabile riservata, Amazon MWAA la ripristina ai valori predefiniti. Di seguito sono elencate le variabili riservate:

- `MWAA__AIRFLOW__COMPONENT`— Utilizzato per identificare il componente Apache Airflow con uno dei seguenti valori:`scheduler`,`worker`, o. `webserver`
- `AIRFLOW__WEBSERVER__SECRET_KEY`— La chiave segreta utilizzata per firmare in modo sicuro i cookie di sessione nel server web Apache Airflow.
- `AIRFLOW__CORE__FERNET_KEY`— La chiave utilizzata per la crittografia e la decrittografia dei dati sensibili memorizzati nel database dei metadati, ad esempio le password di connessione.

- AIRFLOW_HOME— Il percorso della home directory di Apache Airflow in cui i file di configurazione e i file DAG sono archiviati localmente.
- AIRFLOW__CELERY__BROKER_URL— L'URL del broker di messaggi utilizzato per la comunicazione tra lo scheduler Apache Airflow e i nodi di lavoro Celery.
- AIRFLOW__CELERY__RESULT_BACKEND— L'URL del database utilizzato per memorizzare i risultati delle attività di Celery.
- AIRFLOW__CORE__EXECUTOR— La classe executor che Apache Airflow dovrebbe usare. In Amazon MWAA si tratta di CeleryExecutor
- AIRFLOW__CORE__LOAD_EXAMPLES— Utilizzato per attivare o disattivare il caricamento di DAG di esempio.
- AIRFLOW__METRICS__METRICS_BLOCK_LIST— Utilizzato per gestire i parametri di Apache Airflow emessi e acquisiti da Amazon MWAA in. CloudWatch
- SQL_ALCHEMY_CONN— La stringa di connessione per il database RDS per PostgreSQL utilizzata per archiviare i metadati Apache Airflow in Amazon MWAA.
- AIRFLOW__CORE__SQL_ALCHEMY_CONN— Utilizzata per lo stesso scopo, ma seguendo la nuova convenzione di denominazione di Apache SQL_ALCHEMY_CONN Airflow.
- AIRFLOW__CELERY__DEFAULT_QUEUE— La coda predefinita per le attività di Celery in Apache Airflow.
- AIRFLOW__OPERATORS__DEFAULT_QUEUE— La coda predefinita per le attività che utilizzano operatori Apache Airflow specifici.
- AIRFLOW_VERSION— La versione Apache Airflow installata nell'ambiente Amazon MWAA.
- AIRFLOW_CONN_AWS_DEFAULT— Le AWS credenziali predefinite utilizzate per l'integrazione con altri servizi in. AWS
- AWS_DEFAULT_REGION— Imposta la AWS regione predefinita utilizzata con le credenziali predefinite per l'integrazione con altri AWS servizi.
- AWS_REGION— Se definita, questa variabile di ambiente sostituisce i valori nella variabile di ambiente AWS_DEFAULT_REGION e nell'area di impostazione del profilo.
- PYTHONUNBUFFERED— Utilizzato per inviare stdout e trasmettere i stderr log ai container.
- AIRFLOW__METRICS__STATSD_ALLOW_LIST— Utilizzato per configurare un elenco consentito di prefissi separati da virgole per inviare le metriche che iniziano con gli elementi dell'elenco.
- AIRFLOW__METRICS__STATSD_ON— Attiva l'invio di metriche a. StatsD
- AIRFLOW__METRICS__STATSD_HOST— Utilizzato per connettersi al demone. StatSD
- AIRFLOW__METRICS__STATSD_PORT— Utilizzato per connettersi al demone. StatSD

- AIRFLOW__METRICS__STATSD_PREFIX— Utilizzato per connettersi al demone. StatSD
- AIRFLOW__CELERY__WORKER_AUTOSCALE— Imposta la concorrenza massima e minima.
- AIRFLOW__CORE__DAG_CONCURRENCY— Imposta il numero di istanze di attività che possono essere eseguite contemporaneamente dallo scheduler in un DAG.
- AIRFLOW__CORE__MAX_ACTIVE_TASKS_PER_DAG— Imposta il numero massimo di attività attive per DAG.
- AIRFLOW__CORE__PARALLELISM— Definisce il numero massimo di istanze di attività che possono essere eseguite contemporaneamente.
- AIRFLOW__SCHEDULER__PARSING_PROCESSES— Imposta il numero massimo di processi analizzati dallo scheduler per pianificare i DAG.
- AIRFLOW__CELERY_BROKER_TRANSPORT_OPTIONS__VISIBILITY_TIMEOUT— Definisce il numero di secondi che un lavoratore attende per confermare l'attività prima che il messaggio venga recapitato a un altro lavoratore.
- AIRFLOW__CELERY_BROKER_TRANSPORT_OPTIONS__REGION— Imposta la AWS regione per il trasporto di Celery sottostante.
- AIRFLOW__CELERY_BROKER_TRANSPORT_OPTIONS__PREDEFINED_QUEUES— Imposta la coda per il trasporto di Celery sottostante.
- AIRFLOW_SCHEDULER_ALLOWED_RUN_ID_PATTERN— Utilizzato per verificare la validità dell'input per il `run_id` parametro quando si attiva un DAG.
- AIRFLOW__WEBSERVER__BASE_URL— L'URL del server Web utilizzato per ospitare l'interfaccia utente di Apache Airflow.

Variabili d'ambiente non riservate

È possibile utilizzare uno script di avvio per sovrascrivere le variabili di ambiente non riservate. Di seguito sono elencate alcune di queste variabili comuni:

- PATH— specifica un elenco di directory in cui il sistema operativo cerca file e script eseguibili. Quando un comando viene eseguito nella riga di comando, il sistema controlla le directory per trovare ed eseguire il comando. PATH Quando si creano operatori o attività personalizzati in Apache Airflow, potrebbe essere necessario affidarsi a script o eseguibili esterni. Se le directory contenenti questi file non si trovano nella PATH variabile specificata, le attività non vengono eseguite quando il sistema non è in grado di individuarle. Aggiungendo le directory appropriatePATH, le attività di Apache Airflow possono trovare ed eseguire gli eseguibili richiesti.

- PYTHONPATH— Utilizzato dall'interprete Python per determinare in quali directory cercare i moduli e i pacchetti importati. È un elenco di directory che è possibile aggiungere al percorso di ricerca predefinito. Ciò consente all'interprete di trovare e caricare librerie Python non incluse nella libreria standard o installate nelle directory di sistema. Usa questa variabile per aggiungere i tuoi moduli e pacchetti Python personalizzati e usarli con i tuoi DAG.
- LD_LIBRARY_PATH— Una variabile di ambiente utilizzata dal linker e dal loader dinamico in Linux per trovare e caricare librerie condivise. Specifica un elenco di directory contenenti librerie condivise, nelle quali viene effettuata la ricerca prima delle directory di libreria di sistema predefinite. Utilizzate questa variabile per specificare i file binari personalizzati.
- CLASSPATH— Utilizzata da Java Runtime Environment (JRE) e Java Development Kit (JDK) per individuare e caricare classi, librerie e risorse Java in fase di esecuzione. È un elenco di directory, file JAR e archivi ZIP che contengono codice Java compilato.

Utilizzo dei DAG su Amazon MWAA

Per eseguire Directed Acyclic Graphs (DAG) in un ambiente Amazon Managed Workflows for Apache Airflow, copia i file nel bucket di storage Amazon S3 collegato al tuo ambiente, quindi comunica ad Amazon MWAA dove si trovano i tuoi DAG e i file di supporto sulla console Amazon MWAA. Amazon MWAA si occupa della sincronizzazione dei DAG tra lavoratori, pianificatori e server Web. Questa guida descrive come aggiungere o aggiornare i DAG e installare plugin personalizzati e dipendenze Python in un ambiente Amazon MWAA.

Argomenti

- Panoramica dei bucket Amazon S3
- Aggiunta o aggiornamento di DAG
- Installazione di plugin personalizzati
- Installazione delle dipendenze in Python
- Eliminazione di file su Amazon S3

Panoramica dei bucket Amazon S3

Un bucket Amazon S3 per un ambiente Amazon MWAA deve avere l'accesso pubblico bloccato. Per impostazione predefinita, tutte le risorse Amazon S3 (bucket, oggetti e risorse secondarie correlate (ad esempio, la configurazione del ciclo di vita) sono private.

- Solo il proprietario della risorsa, l'AWSaccount che ha creato il bucket, può accedere alla risorsa. Il proprietario della risorsa (ad esempio l'amministratore) può concedere le autorizzazioni di accesso ad altri scrivendo una politica di controllo degli accessi.
- La politica di accesso che configuri deve avere l'autorizzazione per aggiungere DAG, plug-in personalizzati `plugins.zip` e dipendenze Python nel tuo `requirements.txt` bucket Amazon S3. Per un esempio di politica che contiene le autorizzazioni richieste, consulta AmazonMWAA. FullConsoleAccess

Un bucket Amazon S3 per un ambiente Amazon MWAA deve avere il controllo delle versioni abilitato. Quando il controllo delle versioni del bucket Amazon S3 è abilitato, ogni volta che viene creata una nuova versione, viene creata una nuova copia.

- Il controllo delle versioni è abilitato per i plug-in personalizzati in a e le dipendenze Python in un `plugins.zip` bucket `requirements.txt` Amazon S3.
- È necessario specificare la versione di `plugins.zip` e `requirements.txt` sulla console Amazon MWAA ogni volta che questi file vengono aggiornati sul bucket Amazon S3.

Aggiunta o aggiornamento di DAG

I grafici aciclici diretti (DAG) sono definiti all'interno di un file Python che definisce la struttura del DAG come codice. Puoi utilizzare la console Amazon S3AWS CLI, oppure la console per caricare i DAG nel tuo ambiente. Questa pagina descrive i passaggi per aggiungere o aggiornare i DAG Apache Airflow nel tuo ambiente Amazon Managed Workflows for Apache Airflow utilizzando la`dags` cartella nel tuo bucket Amazon S3.

Sezioni

- Prerequisiti
- Come funziona
- Cosa è cambiato nella versione 2
- Test dei DAG utilizzando l'utilità CLI di Amazon MWAA
- Caricamento del codice DAG in Amazon S3
- Specificare il percorso della cartella DAG sulla console Amazon MWAA (la prima volta)
- Visualizzazione delle modifiche nell'interfaccia utente di Apache Airflow
- Fasi successive

Prerequisiti

Avrai bisogno di quanto segue prima di poter completare i passaggi in questa pagina.

- Autorizzazioni: al tuoAWS account deve essere stato concesso dall'amministratore l'accesso alla politica di controllo degliFullConsoleAccess accessi di AmazonMWAA per il tuo ambiente. Inoltre, il tuo ambiente Amazon MWAA deve essere autorizzato dal tuo ruolo di esecuzione per accedere alleAWS risorse utilizzate dal tuo ambiente.
- Accesso: se è necessario l'accesso ai repository pubblici per installare le dipendenze direttamente sul server Web, l'ambiente deve essere configurato con l'accesso al server Web della rete pubblica. Per ulteriori informazioni, consulta the section called “Modalità di accesso Apache Airflow”.

- Configurazione Amazon S3: il bucket Amazon S3 utilizzato per archiviare i DAG, i plug-in personalizzati e le dipendenze Python`requirements.txt` deve essere configurato con accesso pubblico bloccato e controllo delle versioni abilitato.`plugins.zip`

Come funziona

Un grafico aciclico diretto (DAG) è definito all'interno di un singolo file Python che definisce la struttura del DAG come codice. Consiste in quanto segue:

- Una definizione DAG.
- Operatori che descrivono come eseguire il DAG e le attività da eseguire.
- Relazioni con gli operatori che descrivono l'ordine in cui eseguire le attività.

Per eseguire una piattaforma Apache Airflow in un ambiente Amazon MWAA, è necessario copiare la definizione DAG nella`dags` cartella del bucket di archiviazione. Ad esempio, la cartella DAG nel bucket di archiviazione potrebbe avere il seguente aspetto:

Example Cartella DAG

```
dags/
# dag_def.py
```

Amazon MWAA sincronizza automaticamente gli oggetti nuovi e modificati dal bucket Amazon S3 con lo scheduler Amazon MWAA e la`/usr/local/airflow/dags` cartella dei container di lavoro ogni 30 secondi, preservando la gerarchia dei file di origine di Amazon S3, indipendentemente dal tipo di file. Il tempo impiegato dai nuovi DAG per apparire nell'interfaccia utente di Apache Airflow è controllato da`scheduler.dag_dir_list_interval`. Le modifiche ai DAG esistenti verranno rilevate nel ciclo di elaborazione DAG successivo.

> **Note**
>
> Non è necessario includere il file`airflow.cfg` di configurazione nella cartella DAG. Puoi sovrascrivere le configurazioni predefinite di Apache Airflow dalla console Amazon MWAA. Per ulteriori informazioni, consulta Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA.

Cosa è cambiato nella versione 2

- Novità: operatori, hook ed esecutori. Le istruzioni di importazione nei tuoi DAG e i plugin personalizzati che specifichi in un MWAA`plugins.zip` su Amazon sono cambiati tra Apache Airflow v1 e Apache Airflow v2. Ad esempio,`from airflow.contrib.hooks.aws_hook import AwsHook` in Apache Airflow v1 è cambiato`from airflow.providers.amazon.aws.hooks.base_aws import AwsBaseHook` in Apache Airflow v2. Per ulteriori informazioni, consulta Python API Reference nella guida di riferimento di Apache Airflow.

Test dei DAG utilizzando l'utilità CLI di Amazon MWAA

- L'utilità di interfaccia a riga di comando (CLI) replica localmente un ambiente Amazon Managed Workflows for Apache Airflow.
- La CLI crea localmente un'immagine del contenitore Docker simile a un'immagine di produzione Amazon MWAA. Ciò consente di eseguire un ambiente Apache Airflow locale per sviluppare e testare DAG, plug-in personalizzati e dipendenze prima della distribuzione su Amazon MWAA.
- Per eseguire la CLI, vedi aws-mwaa-local-runnerattivaGitHub.

Caricamento del codice DAG in Amazon S3

Puoi usare la console Amazon S3 o ilAWS Command Line Interface (AWS CLI) per caricare il codice DAG nel bucket Amazon S3 nel bucket Amazon S3. I passaggi seguenti presuppongono che tu stia caricando il codice (`.py`) in una cartella denominata`dags` nel tuo bucket Amazon S3.

Utilizzo di AWS CLI

AWS Command Line Interface (AWS CLI) è uno strumento open source che consente di interagire con i servizi AWS utilizzando i comandi nella shell a riga di comando. Per completare le fasi di questa pagina, è necessario quanto segue:

- AWS CLI— Installare la versione 2
- AWS CLI— Configurazione rapida con`aws configure`.

Per caricare utilizzando ilAWS CLI

1. Utilizza il seguente comando per elencare tutti i bucket Amazon S3.

```
aws s3 ls
```

2. Utilizza il seguente comando per elencare i file e le cartelle nel bucket Amazon S3 per il tuo ambiente.

```
aws s3 ls s3://YOUR_S3_BUCKET_NAME
```

3. Il comando seguente carica un`dag_def.py` file in un`dags` cartella.

```
aws s3 cp dag_def.py s3://YOUR_S3_BUCKET_NAME/dags/
```

   Se una cartella denominata`dags` non esiste già nel bucket Amazon S3, questo comando crea la`dags` cartella e carica il file denominato`dag_def.py` nella nuova cartella.

Utilizzo della console Amazon S3

La console Amazon S3 è un'interfaccia utente basata sul Web che consente di creare e gestire le risorse nel bucket Amazon S3. I passaggi seguenti presuppongono che tu abbia una cartella DAG denominata`dags`.

Per eseguire il caricamento utilizzando la console Amazon S3

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Seleziona il link del bucket S3 nel codice DAG nel riquadro S3 per aprire il bucket di archiviazione sulla console Amazon S3.
4. Scegliere la cartella `dags`.
5. Scegliere Upload (Carica).
6. Scegli Aggiungi file.
7. Seleziona la tua copia locale`dag_def.py`, scegli Carica.

Specificare il percorso della cartella DAG sulla console Amazon MWAA (la prima volta)

I seguenti passaggi presuppongono che tu stia specificando il percorso di una cartella denominata bucket Amazon S3`dags`.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli l'ambiente in cui eseguire i DAG.
3. Scegliere Edit (Modifica).
4. Nel codice DAG nel riquadro Amazon S3, scegli Sfoglia S3 accanto al campo della cartella DAG.
5. Seleziona la tua`dags` cartella.
6. Scegliere Choose (Scegli).
7. Scegli Avanti, Aggiorna ambiente.

Visualizzazione delle modifiche nell'interfaccia utente di Apache Airflow

Accesso ad Apache Airflow

Hai bisognoPolitica di accesso all'interfaccia utente di Apache Airflow: AmazonMWAA Access WebServer delle autorizzazioni per il tuoAWS account inAWS Identity and Access Management (IAM) per visualizzare l'interfaccia utente di Apache Airflow.

Per accedere all'interfaccia utente di Apache Airflow

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegliete Open Airflow UI.

Fasi successive

- Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.

Installazione di plugin personalizzati

Amazon Managed Workflows for Apache Airflow supporta il gestore di plugin integrato di Apache Airflow, che consente di utilizzare operatori, hook, sensori o interfacce Apache Airflow personalizzati. Questa pagina descrive i passaggi per installare i plug-in personalizzati Apache Airflow sul tuo ambiente Amazon MWAA utilizzando un file. `plugins.zip`

Indice

- Prerequisiti
- Come funziona
- Cosa è cambiato nella v2
- Panoramica dei plugin personalizzati
  - Limiti di directory e dimensioni dei plugin personalizzati
- Esempi di plugin personalizzati
  - Esempio di utilizzo di una struttura di directory piatta in plugins.zip
  - Esempio di utilizzo di una struttura di directory annidata in plugins.zip
- Creazione di un file plugins.zip
  - Fase uno: testare i plugin personalizzati utilizzando l'utilità CLI di Amazon MWAA
  - Fase due: creare il file plugins.zip
- Caricamento plugins.zip su Amazon S3
  - Utilizzo di AWS CLI
  - Utilizzo della console Amazon S3
- Installazione di plugin personalizzati nel tuo ambiente
  - Specificazione del percorso plugins.zip sulla console Amazon MWAA (la prima volta)
  - Specificazione della plugins.zip versione sulla console Amazon MWAA
- Esempi di casi d'uso per plugins.zip
- Fasi successive

Prerequisiti

Avrai bisogno di quanto segue prima di completare i passaggi di questa pagina.

- Autorizzazioni: al tuo AWS account deve essere stato concesso dall'amministratore l'accesso alla politica di controllo degli accessi di AmazonMWAA per il tuo FullConsoleAccess ambiente. Inoltre, il tuo ambiente Amazon MWAA deve essere autorizzato dal tuo ruolo di esecuzione ad accedere alle AWS risorse utilizzate dal tuo ambiente.
- Accesso: se è necessario accedere agli archivi pubblici per installare le dipendenze direttamente sul server Web, l'ambiente deve essere configurato con l'accesso al server Web di rete pubblica. Per ulteriori informazioni, consulta the section called “Modalità di accesso Apache Airflow”.
- Configurazione Amazon S3 : il bucket Amazon S3 utilizzato per archiviare i DAG`plugins.zip`, i plug-in personalizzati e le dipendenze `requirements.txt` Python deve essere configurato con Public Access Blocked e Versioning Enabled.

Come funziona

Per eseguire plugin personalizzati nel tuo ambiente, devi fare tre cose:

1. Crea un `plugins.zip` file localmente.
2. Carica il `plugins.zip` file locale nel tuo bucket Amazon S3.
3. Specificare la versione di questo file nel campo File Plugins sulla console Amazon MWAA.

> **Note**
>
> Se è la prima volta che carichi un `plugins.zip` file nel tuo bucket Amazon S3, devi anche specificare il percorso del file sulla console Amazon MWAA. Devi completare questo passaggio solo una volta.

Cosa è cambiato nella v2

- Novità: operatori, ganci ed esecutori. Le istruzioni di importazione nei tuoi DAG e i plug-in personalizzati specificati in un MWAA su `plugins.zip` Amazon sono cambiati tra Apache Airflow v1 e Apache Airflow v2. Ad esempio, in Apache Airflow v1 è cambiato `from airflow.contrib.hooks.aws_hook import AwsHook` in Apache Airflow v2. `from airflow.providers.amazon.aws.hooks.base_aws import AwsBaseHook` Per saperne di più, consulta Python API Reference nella guida di riferimento di Apache Airflow.

- Novità: importazioni nei plugin. L'importazione di operatori, sensori, hook aggiunti tramite plugin non `airflow.{operators,sensors,hooks}.<plugin_name>` è più supportata. Queste estensioni devono essere importate come normali moduli Python. Nella versione 2 e successive, l'approccio consigliato consiste nel metterle nella directory DAG e creare e utilizzare un file.airflowignore per escluderle dall'analisi come DAG. Per ulteriori informazioni, consulta Gestione dei moduli e creazione di un operatore personalizzato nella guida di riferimento di Apache Airflow.

Panoramica dei plugin personalizzati

Il gestore di plugin integrato di Apache Airflow può integrare funzionalità esterne al suo interno semplicemente trascinando i file in una cartella. `$AIRFLOW_HOME/plugins` Consente di utilizzare operatori, hook, sensori o interfacce Apache Airflow personalizzati. La sezione seguente fornisce un esempio di strutture di directory piatte e annidate in un ambiente di sviluppo locale e le istruzioni di importazione risultanti, che determinano la struttura delle directory all'interno di un plugins.zip.

Limiti di directory e dimensioni dei plugin personalizzati

Apache Airflow Scheduler e Workers cercano plugin personalizzati durante l'avvio sul contenitore Fargate gestito da AWS Fargate per il vostro ambiente in. `/usr/local/airflow/plugins/*`

- Struttura delle directory. La struttura delle cartelle (at`/*`) si basa sul contenuto del `plugins.zip` file. Ad esempio, se la directory è `plugins.zip` `operators` contenuta come directory di primo livello, la directory verrà estratta nell'ambiente `/usr/local/airflow/plugins/operators` in cui si trova.
- Limite di dimensione. Consigliamo un `plugins.zip` file inferiore a 1 GB. Maggiore è la dimensione di un `plugins.zip` file, maggiore è il tempo di avvio in un ambiente. Sebbene Amazon MWAA non limiti esplicitamente la dimensione di un `plugins.zip` file, se le dipendenze non possono essere installate entro dieci minuti, il servizio Fargate andrà in timeout e tenterà di ripristinare l'ambiente a uno stato stabile.

> **Note**
>
> Per gli ambienti che utilizzano Apache Airflow v1.10.12 o Apache Airflow v2.0.2, Amazon MWAA limita il traffico in uscita sul server Web Apache Airflow e non consente di installare plugin o dipendenze Python direttamente sul server Web. A partire da Apache Airflow v2.2.2, Amazon MWAA può installare plugin e dipendenze direttamente sul server Web.

Esempi di plugin personalizzati

La sezione seguente utilizza il codice di esempio contenuto nella guida di riferimento di Apache Airflow per mostrare come strutturare l'ambiente di sviluppo locale.

Esempio di utilizzo di una struttura di directory piatta in plugins.zip

Apache Airflow v2

L'esempio seguente mostra un `plugins.zip` file con una struttura di directory piatta per Apache Airflow v2.

Example directory piatta con plugins.zip PythonVirtualenvOperator

L'esempio seguente mostra l'albero di primo livello di un file plugins.zip per il plugin PythonVirtualenvOperator personalizzato inCreazione di un plugin personalizzato per Apache AirflowPythonVirtualenvOperator.

```
### virtual_python_plugin.py
```

Example plugins/virtual_python_plugin.py

L'esempio seguente mostra il plugin PythonVirtualenvOperator personalizzato.

```
"""
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""
from airflow.plugins_manager import AirflowPlugin
import airflow.utils.python_virtualenv
```

```
from typing import List

def _generate_virtualenv_cmd(tmp_dir: str, python_bin: str, system_site_packages:
 bool) -> List[str]:
    cmd = ['python3','/usr/local/airflow/.local/lib/python3.7/site-packages/
virtualenv', tmp_dir]
    if system_site_packages:
        cmd.append('--system-site-packages')
    if python_bin is not None:
        cmd.append(f'--python={python_bin}')
    return cmd

airflow.utils.python_virtualenv._generate_virtualenv_cmd=_generate_virtualenv_cmd

class VirtualPythonPlugin(AirflowPlugin):
    name = 'virtual_python_plugin'
```

Apache Airflow v1

L'esempio seguente mostra un `plugins.zip` file con una struttura di directory piatta per Apache Airflow v1.

Example directory piatta con plugins.zip PythonVirtualenvOperator

L'esempio seguente mostra l'albero di primo livello di un file plugins.zip per il plugin PythonVirtualenvOperator personalizzato inCreazione di un plugin personalizzato per Apache AirflowPythonVirtualenvOperator.

```
### virtual_python_plugin.py
```

Example plugins/virtual_python_plugin.py

L'esempio seguente mostra il plugin PythonVirtualenvOperator personalizzato.

```
from airflow.plugins_manager import AirflowPlugin
from airflow.operators.python_operator import PythonVirtualenvOperator

def _generate_virtualenv_cmd(self, tmp_dir):
    cmd = ['python3','/usr/local/airflow/.local/lib/python3.7/site-packages/
virtualenv', tmp_dir]
    if self.system_site_packages:
        cmd.append('--system-site-packages')
    if self.python_version is not None:
```

```
        cmd.append('--python=python{}'.format(self.python_version))
    return cmd
PythonVirtualenvOperator._generate_virtualenv_cmd=_generate_virtualenv_cmd

class EnvVarPlugin(AirflowPlugin):
    name = 'virtual_python_plugin'
```

Esempio di utilizzo di una struttura di directory annidata in plugins.zip

Apache Airflow v2

L'esempio seguente mostra un `plugins.zip` file con directory separate per e una `sensors` directory per `hooks` Apache Airflow v2. `operators`

Example plugins.zip

```
__init__.py
my_airflow_plugin.py
hooks/
|-- __init__.py
|-- my_airflow_hook.py
operators/
|-- __init__.py
|-- my_airflow_operator.py
|-- hello_operator.py
sensors/
|-- __init__.py
|-- my_airflow_sensor.py
```

L'esempio seguente mostra le istruzioni di importazione nel DAG (cartella DAG) che utilizza i plugin personalizzati.

Example dags/your_dag.py

```
from airflow import DAG
from datetime import datetime, timedelta
from operators.my_airflow_operator import MyOperator
from sensors.my_airflow_sensor import MySensor
from operators.hello_operator import HelloOperator

default_args = {
```

```
 'owner': 'airflow',
 'depends_on_past': False,
 'start_date': datetime(2018, 1, 1),
 'email_on_failure': False,
 'email_on_retry': False,
 'retries': 1,
 'retry_delay': timedelta(minutes=5),
}


with DAG('customdag',
   max_active_runs=3,
   schedule_interval='@once',
   default_args=default_args) as dag:

 sens = MySensor(
  task_id='taskA'
 )

 op = MyOperator(
  task_id='taskB',
  my_field='some text'
 )

 hello_task = HelloOperator(task_id='sample-task', name='foo_bar')



 sens >> op >> hello_task
```

Example plugins/my_airflow_plugin.py

```
from airflow.plugins_manager import AirflowPlugin
from hooks.my_airflow_hook import *
from operators.my_airflow_operator import *

class PluginName(AirflowPlugin):

    name = 'my_airflow_plugin'

    hooks = [MyHook]
    operators = [MyOperator]
    sensors = [MySensor]
```

I seguenti esempi mostrano ciascuna delle istruzioni di importazione necessarie nei file dei plugin personalizzati.

Example hooks/my_airflow_hook.py

```
from airflow.hooks.base import BaseHook


class MyHook(BaseHook):

    def my_method(self):
        print("Hello World")
```

Example sensors/my_airflow_sensor.py

```
from airflow.sensors.base import BaseSensorOperator
from airflow.utils.decorators import apply_defaults


class MySensor(BaseSensorOperator):

    @apply_defaults
    def __init__(self,
                 *args,
                 **kwargs):
        super(MySensor, self).__init__(*args, **kwargs)

    def poke(self, context):
        return True
```

Example operators/my_airflow_operator.py

```
from airflow.operators.bash import BaseOperator
from airflow.utils.decorators import apply_defaults
from hooks.my_airflow_hook import MyHook


class MyOperator(BaseOperator):

    @apply_defaults
    def __init__(self,
                 my_field,
```

```
                *args,
                **kwargs):
        super(MyOperator, self).__init__(*args, **kwargs)
        self.my_field = my_field

    def execute(self, context):
        hook = MyHook('my_conn')
        hook.my_method()
```

Example operators/hello_operator.py

```
from airflow.models.baseoperator import BaseOperator
from airflow.utils.decorators import apply_defaults

class HelloOperator(BaseOperator):

    @apply_defaults
    def __init__(
            self,
            name: str,
            **kwargs) -> None:
        super().__init__(**kwargs)
        self.name = name

    def execute(self, context):
        message = "Hello {}".format(self.name)
        print(message)
        return message
```

Segui i passaggi descritti in Testare i plug-in personalizzati utilizzando l'utilità CLI di Amazon MWAA e quindi Creazione di un file plugins.zip per comprimere i contenuti all'interno della directory. `plugins` Ad esempio, `cd plugins`.

Apache Airflow v1

L'esempio seguente mostra un `plugins.zip` file con directory separate per e una directory per `sensors` Apache hooks `operators` Airflow v1.10.12.

Example plugins.zip

```
__init__.py
my_airflow_plugin.py
```

```
hooks/
    |-- __init__.py
    |-- my_airflow_hook.py
operators/
    |-- __init__.py
    |-- my_airflow_operator.py
    |-- hello_operator.py
sensors/
    |-- __init__.py
    |-- my_airflow_sensor.py
```

L'esempio seguente mostra le istruzioni di importazione nel DAG (cartella DAG) che utilizza i plugin personalizzati.

Example dags/your_dag.py

```
from airflow import DAG
from datetime import datetime, timedelta
from operators.my_operator import MyOperator
from sensors.my_sensor import MySensor
from operators.hello_operator import HelloOperator

default_args = {
 'owner': 'airflow',
 'depends_on_past': False,
 'start_date': datetime(2018, 1, 1),
 'email_on_failure': False,
 'email_on_retry': False,
 'retries': 1,
 'retry_delay': timedelta(minutes=5),
}


with DAG('customdag',
   max_active_runs=3,
   schedule_interval='@once',
   default_args=default_args) as dag:

 sens = MySensor(
  task_id='taskA'
 )

 op = MyOperator(
```

```
 task_id='taskB',
 my_field='some text'
)

hello_task = HelloOperator(task_id='sample-task', name='foo_bar')



sens >> op >> hello_task
```

Example plugins/my_airflow_plugin.py

```
from airflow.plugins_manager import AirflowPlugin
from hooks.my_airflow_hook import *
from operators.my_airflow_operator import *
from utils.my_utils import *

class PluginName(AirflowPlugin):

    name = 'my_airflow_plugin'

    hooks = [MyHook]
    operators = [MyOperator]
    sensors = [MySensor]
```

I seguenti esempi mostrano ciascuna delle istruzioni di importazione necessarie nei file dei plugin personalizzati.

Example hooks/my_airflow_hook.py

```
from airflow.hooks.base_hook import BaseHook


class MyHook(BaseHook):

    def my_method(self):
        print("Hello World")
```

Example sensors/my_airflow_sensor.py

```
from airflow.sensors.base_sensor_operator import BaseSensorOperator
```

```
from airflow.utils.decorators import apply_defaults


class MySensor(BaseSensorOperator):

    @apply_defaults
    def __init__(self,
                 *args,
                 **kwargs):
        super(MySensor, self).__init__(*args, **kwargs)

    def poke(self, context):
        return True
```

Example operators/my_airflow_operator.py

```
from airflow.operators.bash_operator import BaseOperator
from airflow.utils.decorators import apply_defaults
from hooks.my_hook import MyHook


class MyOperator(BaseOperator):

    @apply_defaults
    def __init__(self,
                 my_field,
                 *args,
                 **kwargs):
        super(MyOperator, self).__init__(*args, **kwargs)
        self.my_field = my_field

    def execute(self, context):
        hook = MyHook('my_conn')
        hook.my_method()
```

Example operators/hello_operator.py

```
from airflow.models.baseoperator import BaseOperator
from airflow.utils.decorators import apply_defaults

class HelloOperator(BaseOperator):

    @apply_defaults
```

```
    def __init__(
            self,
            name: str,
            **kwargs) -> None:
        super().__init__(**kwargs)
        self.name = name

    def execute(self, context):
        message = "Hello {}".format(self.name)
        print(message)
        return message
```

Segui i passaggi descritti in Testare i plug-in personalizzati utilizzando l'utilità CLI di Amazon MWAA e quindi Creazione di un file plugins.zip per comprimere i contenuti all'interno della directory. `plugins` Ad esempio, `cd plugins`.

Creazione di un file plugins.zip

I passaggi seguenti descrivono i passaggi consigliati per creare un file plugins.zip localmente.

Fase uno: testare i plugin personalizzati utilizzando l'utilità CLI di Amazon MWAA

- L'utilità CLI (Command Line Interface) replica localmente un ambiente Amazon Managed Workflows for Apache Airflow.
- La CLI crea localmente un'immagine del contenitore Docker simile a un'immagine di produzione Amazon MWAA. Ciò consente di eseguire un ambiente Apache Airflow locale per sviluppare e testare DAG, plug-in personalizzati e dipendenze prima della distribuzione su Amazon MWAA.
- Per eseguire la CLI, vedi on. aws-mwaa-local-runner GitHub

Fase due: creare il file plugins.zip

È possibile utilizzare un'utilità di archiviazione ZIP integrata o qualsiasi altra utilità ZIP (come 7zip) per creare un file.zip.

> **Note**
>
> L'utilità zip integrata per il sistema operativo Windows può aggiungere sottocartelle quando si crea un file con estensione zip. Ti consigliamo di verificare il contenuto del file plugins.zip

prima di caricarlo nel tuo bucket Amazon S3 per assicurarti che non siano state aggiunte altre directory.

1. Cambia le directory nella cartella locale dei plugin Airflow. Per esempio:

```
myproject$ cd plugins
```

2. Esegui il comando seguente per assicurarti che i contenuti abbiano autorizzazioni eseguibili (solo macOS e Linux).

```
plugins$ chmod -R 755 .
```

3. Comprimi il contenuto all'interno della cartella`plugins`.

```
plugins$ zip -r plugins.zip .
```

Caricamento **`plugins.zip`** su Amazon S3

Puoi utilizzare la console Amazon S3 o il AWS Command Line Interface (AWS CLI) per caricare un `plugins.zip` file nel tuo bucket Amazon S3.

Utilizzo di AWS CLI

AWS Command Line Interface (AWS CLI) è uno strumento open source che consente di interagire con i servizi AWS utilizzando i comandi nella shell a riga di comando. Per completare i passaggi indicati in questa pagina, è necessario quanto segue:

- AWS CLI— Installa la versione 2.
- AWS CLI— Configurazione rapida con `aws configure`.

Per caricare utilizzando il AWS CLI

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il `plugins.zip` file. Per esempio:

```
cd plugins
```

2. Usa il seguente comando per elencare tutti i tuoi bucket Amazon S3.

```
aws s3 ls
```

3. Usa il seguente comando per elencare i file e le cartelle nel bucket Amazon S3 per il tuo ambiente.

```
aws s3 ls s3://YOUR_S3_BUCKET_NAME
```

4. Usa il seguente comando per caricare il `plugins.zip` file nel bucket Amazon S3 per il tuo ambiente.

```
aws s3 cp plugins.zip s3://YOUR_S3_BUCKET_NAME/plugins.zip
```

Utilizzo della console Amazon S3

La console Amazon S3 è un'interfaccia utente basata sul Web che consente di creare e gestire le risorse nel bucket Amazon S3.

Per caricare utilizzando la console Amazon S3

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Seleziona il link del bucket S3 nel codice DAG nel riquadro S3 per aprire il bucket di archiviazione sulla console Amazon S3.
4. Scegli Carica.
5. Scegli Aggiungi file.
6. Seleziona la copia locale del tuo`plugins.zip`, scegli Carica.

Installazione di plugin personalizzati nel tuo ambiente

Questa sezione descrive come installare i plugin personalizzati caricati nel bucket Amazon S3 specificando il percorso del file plugins.zip e specificando la versione del file plugins.zip ogni volta che il file zip viene aggiornato.

Specificazione del percorso **`plugins.zip`** sulla console Amazon MWAA (la prima volta)

Se è la prima volta che carichi un `plugins.zip` file nel tuo bucket Amazon S3, devi anche specificare il percorso del file sulla console Amazon MWAA. Devi completare questo passaggio solo una volta.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.
4. Nel riquadro del codice DAG di Amazon S3, scegli Browse S3 accanto al file Plugins (campo opzionale).
5. Seleziona il `plugins.zip` file nel tuo bucket Amazon S3.
6. Scegliere Choose (Scegli).
7. Scegli Avanti, Aggiorna ambiente.

Specificazione della **`plugins.zip`** versione sulla console Amazon MWAA

È necessario specificare la versione del `plugins.zip` file sulla console Amazon MWAA ogni volta che si carica una nuova versione del file `plugins.zip` nel bucket Amazon S3.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.
4. Nel riquadro del codice DAG di Amazon S3, scegli `plugins.zip` una versione nell'elenco a discesa.
5. Seleziona Avanti.

Esempi di casi d'uso per plugins.zip

- Scopri come creare un plugin personalizzato inPlugin personalizzato con Apache Hive e Hadoop.
- Scopri come creare un plug-in personalizzato inPlugin personalizzato per la patchPythonVirtualenvOperator .
- Scopri come creare un plug-in personalizzato inPlugin personalizzato con Oracle.

- Scopri come creare un plug-in personalizzato inthe section called “Modifica del fuso orario di un DAG”.

Fasi successive

- Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando on. aws-mwaa-local-runner GitHub

Installazione delle dipendenze in Python

Una dipendenza Python è qualsiasi pacchetto o distribuzione non incluso nell'installazione base di Apache Airflow per la tua versione di Apache Airflow nel tuo ambiente Amazon Managed Workflows for Apache Airflow. Questa pagina descrive i passaggi per installare le dipendenze Python di Apache Airflow nell'ambiente Amazon MWAA utilizzando un file `requirements.txt` nel bucket Amazon S3.

Indice

- Prerequisiti
- Come funziona
- Panoramica delle dipendenze in Python
  - Limiti di posizione e dimensione delle dipendenze in Python
- Creazione di un file requirements.txt
  - Fase uno: testare le dipendenze di Python utilizzando l'utilità CLI di Amazon MWAA
  - Fase due: creare il requirements.txt
- Caricamento requirements.txt su Amazon S3
  - Usando il AWS CLI
  - Utilizzo della console Amazon S3
- Installazione delle dipendenze Python nel proprio ambiente
  - Specificazione del percorso requirements.txt sulla console Amazon MWAA (la prima volta)
  - Specificazione della requirements.txt versione sulla console Amazon MWAA
- Visualizzazione dei log per il requirements.txt
- Fasi successive

Prerequisiti

Avrai bisogno di quanto segue prima di completare i passaggi di questa pagina.

- Autorizzazioni: al tuo AWS account deve essere stato concesso dall'amministratore l'accesso alla politica di controllo degli FullConsoleaccessi di AmazonMWAA per il tuo ambiente. Inoltre, il tuo ambiente Amazon MWAA deve essere autorizzato dal tuo ruolo di esecuzione ad accedere alle AWS risorse utilizzate dal tuo ambiente.
- Accesso: se è necessario accedere agli archivi pubblici per installare le dipendenze direttamente sul server Web, l'ambiente deve essere configurato con l'accesso al server Web di rete pubblica. Per ulteriori informazioni, consulta the section called “Modalità di accesso Apache Airflow”.
- Configurazione Amazon S3 : il bucket Amazon S3 utilizzato per archiviare i DAG`plugins.zip`, i plug-in personalizzati e le dipendenze `requirements.txt` Python deve essere configurato con Public Access Blocked e Versioning Enabled.

Come funziona

Su Amazon MWAA, installi tutte le dipendenze Python caricando un file nel `requirements.txt` tuo bucket Amazon S3, quindi specificando la versione del file sulla console Amazon MWAA ogni volta che aggiorni il file. Amazon MWAA viene eseguito `pip3 install -r requirements.txt` per installare le dipendenze Python sullo scheduler Apache Airflow e su ciascuno dei worker.

Per eseguire le dipendenze di Python nel tuo ambiente, devi fare tre cose:

1. Crea un `requirements.txt` file localmente.
2. Carica il file locale `requirements.txt` nel tuo bucket Amazon S3.
3. Specificare la versione di questo file nel campo File dei requisiti sulla console Amazon MWAA.

> **Note**
>
> Se è la prima volta che crei e carichi un `requirements.txt` file nel tuo bucket Amazon S3, devi anche specificare il percorso del file sulla console Amazon MWAA. Devi completare questo passaggio solo una volta.

Panoramica delle dipendenze in Python

Puoi installare gli extra di Apache Airflow e altre dipendenze Python dalle dipendenze Python Package Index (.orgPyPi), Python wheels (`.whl`) o Python ospitate su un repository privato conforme a /PEP-503 nel tuo ambiente. PyPi

Limiti di posizione e dimensione delle dipendenze in Python

Apache Airflow Scheduler e Workers cercano plugin personalizzati durante l'avvio sul contenitore Fargate gestito da AWS Fargate per il vostro ambiente in. `/usr/local/airflow/plugins`

- Limite di dimensione. Consigliamo un `requirements.txt` file che faccia riferimento a librerie la cui dimensione combinata è inferiore a 1 GB. Più librerie Amazon MWAA deve installare, più lungo è il tempo di avvio in un ambiente. Sebbene Amazon MWAA non limiti esplicitamente la dimensione delle librerie installate, se le dipendenze non possono essere installate entro dieci minuti, il servizio Fargate andrà in timeout e tenterà di ripristinare l'ambiente a uno stato stabile.

Creazione di un file requirements.txt

I passaggi seguenti descrivono i passaggi consigliati per creare un file requirements.txt localmente.

Fase uno: testare le dipendenze di Python utilizzando l'utilità CLI di Amazon MWAA

- L'utilità CLI (Command Line Interface) replica localmente un ambiente Amazon Managed Workflows for Apache Airflow.
- La CLI crea localmente un'immagine del contenitore Docker simile a un'immagine di produzione Amazon MWAA. Ciò consente di eseguire un ambiente Apache Airflow locale per sviluppare e testare DAG, plug-in personalizzati e dipendenze prima della distribuzione su Amazon MWAA.
- Per eseguire la CLI, consulta aws-mwaa-local-runner su. GitHub

Fase due: creare il **`requirements.txt`**

La sezione seguente descrive come specificare le dipendenze Python dall'indice dei pacchetti Python in un file. `requirements.txt`

Apache Airflow v2

1. Esegui il test localmente. Aggiungi altre librerie in modo iterativo per trovare la giusta combinazione di pacchetti e le relative versioni, prima di creare un `requirements.txt` file. Per eseguire l'utilità CLI di Amazon MWAA, consulta aws-mwaa-local-runner su. GitHub
2. Consulta gli extra del pacchetto Apache Airflow. Per visualizzare un elenco dei pacchetti installati per Apache Airflow v2 su Amazon MWAA, consulta Amazon MWAA local runner sul sito Web. `requirements.txt` GitHub
3. Aggiungi una dichiarazione di vincoli. Aggiungi il file dei vincoli per il tuo ambiente Apache Airflow v2 nella parte superiore del file. `requirements.txt` I file di vincoli di Apache Airflow specificano le versioni del provider disponibili al momento del rilascio di Apache Airflow.

   A partire da Apache Airflow v2.7.2, il file dei requisiti deve includere una dichiarazione. `--constraint` Se non fornisci un vincolo, Amazon MWAA te ne specificherà uno per garantire che i pacchetti elencati nei tuoi requisiti siano compatibili con la versione di Apache Airflow che stai utilizzando.

   Nell'esempio seguente, sostituisci *`{environment-version}`* con il numero di versione del tuo ambiente e *`{Python-version} con la versione`* di Python compatibile con il tuo ambiente.

   Per informazioni sulla versione di Python compatibile con il tuo ambiente Apache Airflow, consulta Versioni di Apache Airflow.

   ```
   --constraint "https://raw.githubusercontent.com/apache/airflow/
   constraints-{Airflow-version}/constraints-{Python-version}.txt"
   ```

   Se il file dei vincoli determina che il `xyz==1.0` pacchetto non è compatibile con altri pacchetti dell'ambiente, non `pip3 install` riuscirà a impedire l'installazione di librerie incompatibili nell'ambiente. Se l'installazione di qualsiasi pacchetto non riesce, è possibile visualizzare i log degli errori per ogni componente di Apache Airflow (lo scheduler, il worker e il server web) nel flusso di log corrispondente su Logs. CloudWatch Per ulteriori informazioni sui tipi di registro, vedere. the section called “Visualizzazione dei registri Airflow”
4. Pacchetti Apache Airflow. Aggiungi gli extra del pacchetto e la versione (). == Questo aiuta a evitare che pacchetti con lo stesso nome, ma con una versione diversa, vengano installati nell'ambiente.

```
apache-airflow[package-extra]==2.5.1
```

5. Librerie Python. Aggiungi il nome del pacchetto e la versione (==) nel tuo `requirements.txt` file. In questo modo si evita l'applicazione automatica di future interruzioni del PyPisito .org.

```
library == version
```

Example Boto3 e psycopg2-binary

Questo esempio viene fornito a scopo dimostrativo. Le librerie boto e psycopg2-binary sono incluse nell'installazione di base di Apache Airflow v2 e non devono essere specificate in un file. `requirements.txt`

```
boto3==1.17.54
boto==2.49.0
botocore==1.20.54
psycopg2-binary==2.8.6
```

Se viene specificato un pacchetto senza una versione, Amazon MWAA installa la versione più recente del pacchetto da .org. PyPi Questa versione può entrare in conflitto con altri pacchetti presenti nel tuo. `requirements.txt`

Apache Airflow v1

1. Esegui il test localmente. Aggiungi altre librerie in modo iterativo per trovare la giusta combinazione di pacchetti e le relative versioni, prima di creare un `requirements.txt` file. Per eseguire l'utilità CLI di Amazon MWAA, consulta aws-mwaa-local-runner su. GitHub
2. Consulta gli extra del pacchetto Airflow. Consulta l'elenco dei pacchetti disponibili per Apache Airflow v1.10.12 all'indirizzo https://raw.githubusercontent.com/apache/airflow/constraints-1.10.12/constraints-3.7.txt.
3. Aggiungere il file dei vincoli. Aggiungi il file dei vincoli per Apache Airflow v1.10.12 nella parte superiore del file. `requirements.txt` Se il file dei vincoli determina che il `xyz==1.0` pacchetto non è compatibile con altri pacchetti presenti nell'ambiente, non `pip3 install` riuscirà a impedire l'installazione di librerie incompatibili nell'ambiente.

```
--constraint "https://raw.githubusercontent.com/apache/airflow/
constraints-1.10.12/constraints-3.7.txt"
```

4. Pacchetti Apache Airflow v1.10.12. Aggiungi gli extra del pacchetto Airflow e la versione Apache Airflow v1.10.12 (). == Questo aiuta a prevenire l'installazione di pacchetti con lo stesso nome, ma con una versione diversa, nell'ambiente.

```
apache-airflow[package]==1.10.12
```

Example Secure Shell (SSH)

Il seguente `requirements.txt` file di esempio installa SSH per Apache Airflow v1.10.12.

```
apache-airflow[ssh]==1.10.12
```

5. Librerie Python. Aggiungi il nome del pacchetto e la versione (==) nel tuo `requirements.txt` file. In questo modo si evita l'applicazione automatica di future interruzioni del PyPisito .org.

```
library == version
```

Example Boto3

Il seguente `requirements.txt` file di esempio installa la libreria Boto3 per Apache Airflow v1.10.12.

```
boto3 == 1.17.4
```

Se viene specificato un pacchetto senza una versione, Amazon MWAA installa la versione più recente del pacchetto da .org. PyPi Questa versione può entrare in conflitto con altri pacchetti presenti nel tuo. `requirements.txt`

Caricamento **`requirements.txt`** su Amazon S3

Puoi utilizzare la console Amazon S3 o il AWS Command Line Interface (AWS CLI) per caricare un `requirements.txt` file nel tuo bucket Amazon S3.

Usando il AWS CLI

Il AWS Command Line Interface (AWS CLI) è uno strumento open source che consente di interagire con i AWS servizi utilizzando i comandi nella shell della riga di comando. Per completare la procedura descritta in questa pagina, è necessario quanto segue:

- AWS CLI — Installa la versione 2.
- AWS CLI — Configurazione rapida con `aws configure`.

Per caricare utilizzando il AWS CLI

1. Usa il seguente comando per elencare tutti i tuoi bucket Amazon S3.

```
aws s3 ls
```

2. Usa il seguente comando per elencare i file e le cartelle nel bucket Amazon S3 per il tuo ambiente.

```
aws s3 ls s3://YOUR_S3_BUCKET_NAME
```

3. Il comando seguente carica un `requirements.txt` file in un bucket Amazon S3.

```
aws s3 cp requirements.txt s3://YOUR_S3_BUCKET_NAME/requirements.txt
```

Utilizzo della console Amazon S3

La console Amazon S3 è un'interfaccia utente basata sul Web che consente di creare e gestire le risorse nel bucket Amazon S3.

Per caricare utilizzando la console Amazon S3

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Seleziona il link del bucket S3 nel codice DAG nel riquadro S3 per aprire il bucket di archiviazione sulla console Amazon S3.
4. Scegli Carica.
5. Scegli Aggiungi file.

6. Seleziona la copia locale del tuo`requirements.txt`, scegli Carica.

Installazione delle dipendenze Python nel proprio ambiente

Questa sezione descrive come installare le dipendenze caricate nel bucket Amazon S3 specificando il percorso del file requirements.txt e specificando la versione del file requirements.txt ogni volta che viene aggiornato.

Specificazione del percorso **`requirements.txt`** sulla console Amazon MWAA (la prima volta)

Se è la prima volta che crei e carichi un `requirements.txt` file nel tuo bucket Amazon S3, devi anche specificare il percorso del file sulla console Amazon MWAA. Devi completare questo passaggio solo una volta.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.
4. Nel riquadro del codice DAG di Amazon S3, scegli Browse S3 accanto al file dei requisiti (campo opzionale).
5. Seleziona il `requirements.txt` file nel tuo bucket Amazon S3.
6. Scegliere Choose (Scegli).
7. Scegli Avanti, Aggiorna ambiente.

È possibile iniziare a utilizzare i nuovi pacchetti subito dopo il completamento dell'aggiornamento dell'ambiente.

Specificazione della **`requirements.txt`** versione sulla console Amazon MWAA

È necessario specificare la versione del `requirements.txt` file sulla console Amazon MWAA ogni volta che si carica una nuova versione del file `requirements.txt` nel bucket Amazon S3.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.

4. Nel riquadro del codice DAG di Amazon S3, scegli `requirements.txt` una versione nell'elenco a discesa.
5. Scegli Avanti, Aggiorna ambiente.

È possibile iniziare a utilizzare i nuovi pacchetti subito dopo il completamento dell'aggiornamento dell'ambiente.

Visualizzazione dei log per il **`requirements.txt`**

Puoi visualizzare i log di Apache Airflow per Scheduler, pianificando i flussi di lavoro e analizzando la cartella. `dags` I passaggi seguenti descrivono come aprire il gruppo di log per Scheduler sulla console Amazon MWAA e visualizzare i log di Apache Airflow sulla console Logs. CloudWatch

Per visualizzare i log di un **`requirements.txt`**

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli il gruppo di log dello scheduler Airflow nel riquadro Monitoraggio.
4. Scegli il `requirements_install_ip` log in Log Streams.
5. Dovresti vedere l'elenco dei pacchetti che sono stati installati nell'ambiente all'indirizzo`/usr/local/airflow/.local/bin`. Per esempio:

```
Collecting appdirs==1.4.4 (from -r /usr/local/airflow/.local/bin (line 1))
Downloading https://files.pythonhosted.org/
packages/3b/00/2344469e2084fb28kjdsfiuyweb47389789vxbmnbjhsdgf5463acd6cf5e3db69324/
appdirs-1.4.4-py2.py3-none-any.whl
Collecting astroid==2.4.2 (from -r /usr/local/airflow/.local/bin (line 2))
```

6. Controlla l'elenco dei pacchetti e verifica se qualcuno di questi ha riscontrato un errore durante l'installazione. Se qualcosa è andato storto, potresti visualizzare un errore simile al seguente:

```
2021-03-05T14:34:42.731-07:00
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
```

Fasi successive

- Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando aws-mwaa-local-runner on. GitHub

Eliminazione di file su Amazon S3

Questa pagina descrive come funziona il controllo delle versioni in un bucket Amazon S3 per un ambiente Amazon Managed Workflows for Apache Airflow e i passaggi per eliminare un DAG o un file. `plugins.zip` `requirements.txt`

Indice

- Prerequisiti
- Panoramica del controllo delle versioni
- Come funziona
- Eliminazione di un DAG su Amazon S3
- Rimozione di un file requirements.txt o plugins.zip «attuale» da un ambiente
- Eliminazione di una versione «non corrente» (precedente) di requirements.txt o plugins.zip
- Utilizzo dei cicli di vita per eliminare le versioni «non correnti» (precedenti) ed eliminare automaticamente i marker
- Esempio di politica del ciclo di vita per eliminare le versioni «non correnti» di requirements.txt ed eliminare automaticamente i marker
- Fasi successive

Prerequisiti

Avrai bisogno di quanto segue prima di completare i passaggi di questa pagina.

- Autorizzazioni: al tuo AWS account deve essere stato concesso dall'amministratore l'accesso alla politica di controllo degli accessi di AmazonMWAA per il tuo FullConsoleAccess ambiente. Inoltre, il tuo ambiente Amazon MWAA deve essere autorizzato dal tuo ruolo di esecuzione ad accedere alle risorse utilizzate dal tuo ambiente. AWS
- Accesso: se è necessario accedere agli archivi pubblici per installare le dipendenze direttamente sul server Web, l'ambiente deve essere configurato con l'accesso al server Web di rete pubblica. Per ulteriori informazioni, consulta the section called “Modalità di accesso Apache Airflow”.

- Configurazione Amazon S3 : il bucket Amazon S3 utilizzato per archiviare i DAG`plugins.zip`, i plug-in personalizzati e le dipendenze `requirements.txt` Python deve essere configurato con Public Access Blocked e Versioning Enabled.

Panoramica del controllo delle versioni

Il bucket `requirements.txt` and `plugins.zip` in Amazon S3 ha una versione. Quando il controllo delle versioni del bucket Amazon S3 è abilitato per un oggetto e un elemento (ad esempio, plugins.zip) viene eliminato da un bucket Amazon S3, il file non viene eliminato completamente. Ogni volta che un elemento viene eliminato su Amazon S3, viene creata una nuova copia del file, ovvero un file di errore 404 (oggetto non trovato) /0k che dice «Non sono qui». Amazon S3 lo chiama marker di eliminazione. Un marker di eliminazione è una versione «nulla» del file con un nome chiave (o chiave) e un ID di versione come qualsiasi altro oggetto.

Ti consigliamo di eliminare periodicamente le versioni dei file e i marker di eliminazione per ridurre i costi di storage per il tuo bucket Amazon S3. Per eliminare completamente le versioni «non correnti» (precedenti) dei file, devi eliminare le versioni dei file e quindi il marker di eliminazione relativo alla versione.

Come funziona

Amazon MWAA esegue un'operazione di sincronizzazione sul tuo bucket Amazon S3 ogni trenta secondi. Ciò fa sì che tutte le eliminazioni di DAG in un bucket Amazon S3 vengano sincronizzate con l'immagine Airflow del contenitore Fargate.

Per `requirements.txt` i file `plugins.zip` e, le modifiche avvengono solo dopo un aggiornamento dell'ambiente quando Amazon MWAA crea una nuova immagine Airflow del contenitore Fargate con i plugin personalizzati e le dipendenze Python. Se elimini la versione corrente di un `plugins.zip` file `requirements.txt` or e quindi aggiorni l'ambiente senza fornire una nuova versione per il file eliminato, l'aggiornamento avrà esito negativo e verrà visualizzato un messaggio di errore, ad esempio «Impossibile leggere la versione del file». `{version} {file}`

Eliminazione di un DAG su Amazon S3

Un file DAG (`.py`) non ha una versione e può essere eliminato direttamente dalla console Amazon S3. I passaggi seguenti descrivono come eliminare un DAG dal tuo bucket Amazon S3.

Per eliminare un DAG

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Seleziona il link del bucket S3 nel codice DAG nel riquadro S3 per aprire il bucket di archiviazione sulla console Amazon S3.
4. Scegliere la cartella `dags`.
5. Seleziona il DAG, Elimina.
6. In Eliminare oggetti? , digitare`delete`.
7. Scegliere Delete objects (Elimina oggetti).

> **Note**
>
> Apache Airflow conserva le esecuzioni DAG storiche. Una volta eseguito in Apache Airflow, un DAG rimane nell'elenco dei DAG Airflow indipendentemente dallo stato del file, finché non viene eliminato in Apache Airflow. Per eliminare un DAG in Apache Airflow, scegli il pulsante rosso «elimina» nella colonna Collegamenti.

Rimozione di un file requirements.txt o plugins.zip «attuale» da un ambiente

Al momento, non esiste un modo per rimuovere plugins.zip o requirements.txt da un ambiente dopo che sono stati aggiunti, ma stiamo lavorando al problema. Nel frattempo, una soluzione alternativa consiste nel puntare rispettivamente a un file di testo o zip vuoto.

Eliminazione di una versione «non corrente» (precedente) di requirements.txt o plugins.zip

La versione `plugins.zip` dei file `requirements.txt` e del bucket Amazon S3 è in Amazon MWAA. Se desideri eliminare completamente questi file dal tuo bucket Amazon S3, devi recuperare la versione corrente (121212) dell'oggetto (ad esempio, plugins.zip), eliminare la versione e quindi rimuovere il marker di eliminazione per le versioni del file.

Puoi anche eliminare versioni di file «non correnti» (precedenti) sulla console Amazon S3; tuttavia, dovrai comunque eliminare il marker di eliminazione utilizzando una delle seguenti opzioni.

- Per recuperare la versione dell'oggetto, consulta Recupero delle versioni degli oggetti da un bucket abilitato al controllo delle versioni nella guida di Amazon S3.
- Per eliminare la versione dell'oggetto, consulta Eliminazione delle versioni degli oggetti da un bucket abilitato al controllo delle versioni nella guida Amazon S3.
- Per rimuovere un marker di eliminazione, consulta Managing delete marker nella guida Amazon S3.

Utilizzo dei cicli di vita per eliminare le versioni «non correnti» (precedenti) ed eliminare automaticamente i marker

Puoi configurare una politica del ciclo di vita per il tuo bucket Amazon S3 per eliminare le versioni «non correnti» (precedenti) dei file plugins.zip e requirements.txt nel tuo bucket Amazon S3 dopo un certo numero di giorni o per rimuovere l'indicatore di eliminazione di un oggetto scaduto.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Nella sezione Codice DAG in Amazon S3, scegli il tuo bucket Amazon S3.
4. Scegli Crea regola del ciclo di vita.

Esempio di politica del ciclo di vita per eliminare le versioni «non correnti» di requirements.txt ed eliminare automaticamente i marker

L'esempio seguente mostra come creare una regola del ciclo di vita che elimini definitivamente le versioni «non correnti» di un file requirements.txt e i relativi indicatori di eliminazione dopo trenta giorni.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Nella sezione Codice DAG in Amazon S3, scegli il tuo bucket Amazon S3.
4. Scegli Crea regola del ciclo di vita.
5. Nel nome della regola del ciclo di vita, digita. `Delete previous requirements.txt versions and delete markers after thirty days`
6. In Prefix, requisiti.

7. In Azioni relative alle regole del ciclo di vita, scegli Elimina definitivamente le versioni precedenti degli oggetti ed Elimina i marker di eliminazione scaduti o i caricamenti multiparte incompleti.
8. In Numero di giorni dopo che gli oggetti diventano versioni precedenti, digitate. `30`
9. In Marcatori di eliminazione di oggetti scaduti, scegliete Elimina marker di eliminazione di oggetti scaduti, gli oggetti vengono eliminati definitivamente dopo 30 giorni.

Fasi successive

- Scopri di più sui marker di eliminazione di Amazon S3 nella sezione Gestione dei marker di eliminazione.
- Scopri di più sui cicli di vita di Amazon S3 nella sezione Oggetti in scadenza.

Rete

Questa guida descrive la configurazione di rete Amazon VPC necessaria per un ambiente Amazon MWAA.

Sections

- Informazioni sulla rete su Amazon MWAA
- Sicurezza nel tuo VPC su Amazon MWAA
- Gestione dell'accesso agli endpoint Amazon VPC specifici del servizio su Amazon MWAA
- Creazione degli endpoint del servizio VPC richiesti in un Amazon VPC con routing privato
- Gestione degli endpoint Amazon VPC personali su Amazon MWAA

Informazioni sulla rete su Amazon MWAA

Un Amazon VPC è una rete virtuale collegata al tuo AWS account. Ti offre sicurezza nel cloud e la possibilità di scalare dinamicamente fornendo un controllo granulare sull'infrastruttura virtuale e sulla segmentazione del traffico di rete. Questa pagina descrive l'infrastruttura Amazon VPC con routing pubblico o routing privato necessaria per supportare un ambiente Amazon Managed Workflows for Apache Airflow.

Indice

- Termini
- Cosa è supportato
- Panoramica dell'infrastruttura VPC
  - Routing pubblico su Internet
  - Routing privato senza accesso a Internet
- Esempi di casi d'uso per una modalità di accesso Amazon VPC e Apache Airflow
  - L'accesso a Internet è consentito: nuova rete Amazon VPC
  - L'accesso a Internet non è consentito: nuova rete Amazon VPC
  - L'accesso a Internet non è consentito: rete Amazon VPC esistente

Termini

Routing pubblico

Una rete Amazon VPC con accesso a Internet.

Routing privato

Una rete Amazon VPC senza accesso a Internet.

Cosa è supportato

La tabella seguente descrive i tipi di Amazon VPC supportati da Amazon MWAA.

Tipi di Amazon VPC	Supportato	
Un Amazon VPC di proprietà dell'account che sta tentando di creare l'ambiente.	Sì	
Un Amazon VPC condiviso in cui più AWS account creano le proprie AWS risorse.	Sì	

Panoramica dell'infrastruttura VPC

Quando crei un ambiente Amazon MWAA, Amazon MWAA crea da uno a due endpoint VPC per il tuo ambiente in base alla modalità di accesso Apache Airflow scelta per il tuo ambiente. Questi endpoint appaiono come Interfacce di rete elastiche (ENI) con IP privati nel tuo Amazon VPC. Dopo la creazione di questi endpoint, tutto il traffico destinato a questi IP viene instradato privatamente o pubblicamente ai servizi corrispondenti utilizzati dal tuo ambiente. AWS

La sezione seguente descrive l'infrastruttura Amazon VPC necessaria per instradare il traffico pubblicamente su Internet o privatamente all'interno del tuo Amazon VPC.

Routing pubblico su Internet

Questa sezione descrive l'infrastruttura Amazon VPC di un ambiente con routing pubblico. Avrai bisogno della seguente infrastruttura VPC:

- Un gruppo di sicurezza VPC. Un gruppo di sicurezza VPC funge da firewall virtuale per controllare il traffico di rete in ingresso (in entrata) e in uscita (in uscita) su un'istanza.
  - È possibile specificare fino a 5 gruppi di sicurezza.
  - Il gruppo di sicurezza deve specificare a se stesso una regola di ingresso autoreferenziale.
  - Il gruppo di sicurezza deve specificare una regola in uscita per tutto il traffico (). `0.0.0.0/0`
  - Il gruppo di sicurezza deve consentire tutto il traffico nella regola di autoreferenziazione. Ad esempio, (Consigliato) Esempio di gruppo di sicurezza autoreferenziato per tutti gli accessi .
  - Il gruppo di sicurezza può facoltativamente limitare ulteriormente il traffico specificando l'intervallo di porte per l'intervallo di porte HTTPS `443` e un intervallo di porte TCP. `5432` Ad esempio (Facoltativo) Esempio di gruppo di sicurezza che limita l'accesso in entrata alla porta 5432 e (Facoltativo) Esempio di gruppo di sicurezza che limita l'accesso in entrata alla porta 443.
- Due sottoreti pubbliche. Una sottorete pubblica è una sottorete associata a una tabella di routing con una route a un Internet Gateway.
  - Sono necessarie due sottoreti pubbliche. Ciò consente ad Amazon MWAA di creare una nuova immagine del contenitore per il tuo ambiente nell'altra zona di disponibilità, in caso di guasto di un container.
  - Le sottoreti devono trovarsi in zone di disponibilità diverse. Ad esempio, `us-east-1a`, `us-east-1b`.
  - Le sottoreti devono essere instradate verso un gateway NAT (o istanza NAT) con un indirizzo IP elastico (EIP).
  - Le sottoreti devono disporre di una tabella di routing che indirizza il traffico collegato a Internet verso un gateway Internet.
- Due sottoreti private. Una sottorete privata è una sottorete non associata a una tabella di routing con un percorso verso un gateway Internet.
  - Sono necessarie due sottoreti private. Ciò consente ad Amazon MWAA di creare una nuova immagine del contenitore per il tuo ambiente nell'altra zona di disponibilità, in caso di guasto di un container.
  - Le sottoreti devono trovarsi in zone di disponibilità diverse. Ad esempio, `us-east-1a`, `us-east-1b`.
  - Le sottoreti devono disporre di una tabella di routing verso un dispositivo NAT (gateway o istanza).
  - Le sottoreti non devono essere instradate verso un gateway Internet.

- Una lista di controllo degli accessi alla rete (ACL). Un NACL gestisce (mediante regole di autorizzazione o rifiuto) il traffico in entrata e in uscita a livello di sottorete.
  - Il NACL deve avere una regola in entrata che consenta tutto il traffico (). `0.0.0.0/0`
  - Il NACL deve avere una regola in uscita che neghi tutto il traffico (). `0.0.0.0/0`
  - Ad esempio, (Consigliato) Esempio di ACL.
- Due gateway NAT (o istanze NAT). Un dispositivo NAT inoltra il traffico dalle istanze della sottorete privata a Internet o ad altri AWS servizi, quindi reindirizza la risposta alle istanze.
  - Il dispositivo NAT deve essere collegato a una sottorete pubblica. (Un dispositivo NAT per sottorete pubblica).
  - Il dispositivo NAT deve avere un indirizzo IPv4 elastico (EIP) collegato a ciascuna sottorete pubblica.
- Un gateway Internet. Un gateway Internet collega un Amazon VPC a Internet e ad altri AWS servizi.
  - Un gateway Internet deve essere collegato ad Amazon VPC.

Routing privato senza accesso a Internet

Questa sezione descrive l'infrastruttura Amazon VPC di un ambiente con routing privato. Avrai bisogno della seguente infrastruttura VPC:

- Un gruppo di sicurezza VPC. Un gruppo di sicurezza VPC funge da firewall virtuale per controllare il traffico di rete in ingresso (in entrata) e in uscita (in uscita) su un'istanza.
  - È possibile specificare fino a 5 gruppi di sicurezza.
  - Il gruppo di sicurezza deve specificare a se stesso una regola di ingresso autoreferenziale.
  - Il gruppo di sicurezza deve specificare una regola in uscita per tutto il traffico (). `0.0.0.0/0`
  - Il gruppo di sicurezza deve consentire tutto il traffico nella regola di autoreferenziazione. Ad esempio, (Consigliato) Esempio di gruppo di sicurezza autoreferenziato per tutti gli accessi .
  - Il gruppo di sicurezza può facoltativamente limitare ulteriormente il traffico specificando l'intervallo di porte per l'intervallo di porte HTTPS `443` e un intervallo di porte TCP. `5432` Ad esempio (Facoltativo) Esempio di gruppo di sicurezza che limita l'accesso in entrata alla porta 5432 e (Facoltativo) Esempio di gruppo di sicurezza che limita l'accesso in entrata alla porta 443.
- Due sottoreti private. Una sottorete privata è una sottorete non associata a una tabella di routing con un percorso verso un gateway Internet.

  - Sono necessarie due sottoreti private. Ciò consente ad Amazon MWAA di creare una nuova immagine del contenitore per il tuo ambiente nell'altra zona di disponibilità, in caso di guasto di un container.
  - Le sottoreti devono trovarsi in zone di disponibilità diverse. Ad esempio, `us-east-1a`, `us-east-1b`.
  - Le sottoreti devono avere una tabella di routing verso gli endpoint VPC.
  - Le sottoreti non devono avere una tabella di routing verso un dispositivo NAT (gateway o istanza), né un gateway Internet.
- Una lista di controllo degli accessi alla rete (ACL). Un NACL gestisce (mediante regole di autorizzazione o rifiuto) il traffico in entrata e in uscita a livello di sottorete.
  - Il NACL deve avere una regola in entrata che consenta tutto il traffico (). `0.0.0.0/0`
  - Il NACL deve avere una regola in uscita che neghi tutto il traffico (). `0.0.0.0/0`
  - Ad esempio, (Consigliato) Esempio di ACL.
- Una tabella di percorsi locali. Una tabella di routing locale è una route predefinita per la comunicazione all'interno del VPC.
  - La tabella delle rotte locali deve essere associata alle sottoreti private.
  - La tabella di routing locale deve consentire alle istanze del tuo VPC di comunicare con la tua rete. Ad esempio, se utilizzi un endpoint per accedere AWS Client VPN all'interfaccia VPC per il tuo server Web Apache Airflow, la tabella di routing deve essere indirizzata all'endpoint VPC.
- Endpoint VPC per ogni AWS servizio utilizzato dal tuo ambiente e endpoint VPC Apache Airflow AWS nella stessa regione e Amazon VPC dell'ambiente Amazon MWAA.
  - Un endpoint VPC per ogni AWS servizio utilizzato dall'ambiente e endpoint VPC per Apache Airflow. Ad esempio, (Obbligatori) Endpoint VPC.
  - Gli endpoint VPC devono avere il DNS privato abilitato.
  - Gli endpoint VPC devono essere associati alle due sottoreti private del tuo ambiente.
  - Gli endpoint VPC devono essere associati al gruppo di sicurezza dell'ambiente.
  - La policy degli endpoint VPC per ogni endpoint deve essere configurata per consentire l'accesso ai AWS servizi utilizzati dall'ambiente. Ad esempio, (Consigliato) Esempio di policy degli endpoint VPC per consentire tutti gli accessi.
  - È necessario configurare una policy sugli endpoint VPC per Amazon S3 per consentire l'accesso ai bucket. Ad esempio, (Consigliato) Esempio di policy degli endpoint del gateway Amazon S3 per consentire l'accesso ai bucket.

Esempi di casi d'uso per una modalità di accesso Amazon VPC e Apache Airflow

Questa sezione descrive i diversi casi d'uso per l'accesso alla rete nel tuo Amazon VPC e la modalità di accesso al server Web Apache Airflow da scegliere sulla console Amazon MWAA.

L'accesso a Internet è consentito: nuova rete Amazon VPC

Se l'accesso a Internet nel tuo VPC è consentito dalla tua organizzazione e desideri che gli utenti accedano al tuo server Web Apache Airflow tramite Internet:

1. Crea una rete Amazon VPC con accesso a Internet.
2. Crea un ambiente con la modalità di accesso alla rete pubblica per il tuo server Web Apache Airflow.
3. Cosa consigliamo: ti consigliamo di utilizzare il modello di AWS CloudFormation avvio rapido che crea contemporaneamente l'infrastruttura Amazon VPC, un bucket Amazon S3 e un ambiente Amazon MWAA. Per ulteriori informazioni, vedi Tutorial di avvio rapido per Amazon Managed Workflows for Apache Airflow.

Se l'accesso a Internet nel tuo VPC è consentito dalla tua organizzazione e desideri limitare l'accesso al server Web Apache Airflow agli utenti all'interno del tuo VPC:

1. Crea una rete Amazon VPC con accesso a Internet.
2. Crea un meccanismo per accedere all'endpoint dell'interfaccia VPC per il tuo server Web Apache Airflow dal tuo computer.
3. Crea un ambiente con la modalità di accesso alla rete privata per il tuo server Web Apache Airflow.
4. Cosa consigliamo:
    a. Ti consigliamo di utilizzare la console Amazon MWAA in o Opzione uno: creazione della rete VPC sulla console Amazon MWAA il AWS CloudFormation modello in. Opzione due: creazione di una rete Amazon VPCconaccesso a Internet
    b. Ti consigliamo di configurare l'accesso utilizzando un AWS Client VPN al tuo server Web Apache Airflow in. Tutorial: Configurazione dell'accesso alla rete privata utilizzando unAWS Client VPN

L'accesso a Internet non è consentito: nuova rete Amazon VPC

Se l'accesso a Internet nel tuo VPC non è consentito dalla tua organizzazione:

1. Crea una rete Amazon VPC senza accesso a Internet.
2. Crea un meccanismo per accedere all'endpoint dell'interfaccia VPC per il tuo server Web Apache Airflow dal tuo computer.
3. Crea endpoint VPC per ogni AWS servizio utilizzato dal tuo ambiente.
4. Crea un ambiente con la modalità di accesso alla rete privata per il tuo server Web Apache Airflow.
5. Cosa consigliamo:

    a. Consigliamo di utilizzare il AWS CloudFormation modello per creare un Amazon VPC senza accesso a Internet e gli endpoint VPC per ogni servizio AWS utilizzato da Amazon MWAA in. Opzione tre: creazione di una rete Amazon VPCsenzaaccesso a Internet

    b. Ti consigliamo di configurare l'accesso utilizzando un al tuo server Web AWS Client VPN Apache Airflow in. Tutorial: Configurazione dell'accesso alla rete privata utilizzando unAWS Client VPN

L'accesso a Internet non è consentito: rete Amazon VPC esistente

Se l'accesso a Internet nel tuo VPC non è consentito dalla tua organizzazione e disponi già della rete Amazon VPC richiesta senza accesso a Internet:

1. Crea endpoint VPC per ogni AWS servizio utilizzato dal tuo ambiente.
2. Crea endpoint VPC per Apache Airflow.
3. Crea un meccanismo per accedere all'endpoint dell'interfaccia VPC per il tuo server Web Apache Airflow dal tuo computer.
4. Crea un ambiente con la modalità di accesso alla rete privata per il tuo server Web Apache Airflow.
5. Cosa consigliamo:

    a. Consigliamo di creare e collegare gli endpoint VPC necessari per AWS ogni servizio utilizzato da Amazon MWAA e gli endpoint VPC necessari per Apache Airflow in. Creazione degli endpoint del servizio VPC richiesti in un Amazon VPC con routing privato

b. Ti consigliamo di configurare l'accesso utilizzando un al tuo server Web Apache Airflow in. AWS Client VPN Tutorial: Configurazione dell'accesso alla rete privata utilizzando unAWS Client VPN

Sicurezza nel tuo VPC su Amazon MWAA

Questa pagina descrive i componenti Amazon VPC utilizzati per proteggere il tuo ambiente Amazon Managed Workflows for Apache Airflow e le configurazioni necessarie per questi componenti.

Indice

- Termini
- Panoramica sulla sicurezza
- Liste di controllo accessi di rete (ACL)
  - (Consigliato) Esempio di ACL
- Gruppi di sicurezza VPC
  - (Consigliato) Esempio di gruppo di sicurezza autoreferenziato per tutti gli accessi
  - (Facoltativo) Esempio di gruppo di sicurezza che limita l'accesso in entrata alla porta 5432
  - (Facoltativo) Esempio di gruppo di sicurezza che limita l'accesso in entrata alla porta 443
- Policy degli endpoint VPC (solo routing privato)
  - (Consigliato) Esempio di policy degli endpoint VPC per consentire tutti gli accessi
  - (Consigliato) Esempio di policy degli endpoint del gateway Amazon S3 per consentire l'accesso ai bucket

Termini

Routing pubblico

Una rete Amazon VPC con accesso a Internet.

Routing privato

Una rete Amazon VPC senza accesso a Internet.

Panoramica sulla sicurezza

I gruppi di sicurezza e gli elenchi di controllo degli accessi (ACL) forniscono modi per controllare il traffico di rete attraverso le sottoreti e le istanze del tuo Amazon VPC utilizzando regole da te specificate.

- Il traffico di rete da e verso una sottorete può essere controllato da elenchi di controllo degli accessi (ACL). È necessario un solo ACL e lo stesso ACL può essere utilizzato in più ambienti.
- Il traffico di rete da e verso un'istanza può essere controllato da un gruppo di sicurezza Amazon VPC. Puoi utilizzare da uno a cinque gruppi di sicurezza per ambiente.
- Il traffico di rete da e verso un'istanza può essere controllato anche dalle policy degli endpoint VPC. Se l'accesso a Internet all'interno di Amazon VPC non è consentito dalla tua organizzazione e utilizzi una rete Amazon VPC con routing privato, è necessaria una policy sugli endpoint VPC per gli endpoint VPC e gli endpoint AWS VPC Apache Airflow.

Liste di controllo accessi di rete (ACL)

Una lista di controllo degli accessi alla rete (ACL) può gestire (mediante regole di autorizzazione o negazione) il traffico in entrata e in uscita a livello di sottorete. Un ACL è stateless, il che significa che le regole in entrata e in uscita devono essere specificate separatamente ed esplicitamente. Viene utilizzato per specificare i tipi di traffico di rete consentiti in entrata o in uscita dalle istanze in una rete VPC.

Ogni Amazon VPC dispone di un ACL predefinito che consente tutto il traffico in entrata e in uscita. Puoi modificare le regole ACL predefinite o creare un ACL personalizzato e collegarlo alle tue sottoreti. Una sottorete può avere un solo ACL collegato alla volta, ma un ACL può essere collegato a più sottoreti.

(Consigliato) Esempio di ACL

L'esempio seguente mostra le regole ACL in entrata e in uscita che possono essere utilizzate per un Amazon VPC per un Amazon VPC con routing pubblico o routing privato.

Numero della regola	Type	Protocollo	Intervallo porte	Crea	Consenti/ Nega
100	Tutto il traffico IPv4	Tutti	Tutti	0.0.0.0/0	Consenso
*	Tutto il traffico IPv4	Tutti	Tutti	0.0.0.0/0	Rifiuta

Gruppi di sicurezza VPC

Un gruppo di sicurezza VPC funge da firewall virtuale che controlla il traffico di rete a livello di istanza. Un gruppo di sicurezza è dotato di stato, il che significa che quando è consentita una connessione in entrata, è consentito rispondere. Viene utilizzato per specificare i tipi di traffico di rete consentiti dalle istanze in una rete VPC.

Ogni Amazon VPC ha un gruppo di sicurezza predefinito. Per impostazione predefinita, non ha regole in entrata. Ha una regola in uscita che consente tutto il traffico in uscita. Puoi modificare le regole predefinite del gruppo di sicurezza o creare un gruppo di sicurezza personalizzato e collegarlo al tuo Amazon VPC. Su Amazon MWAA, devi configurare le regole in entrata e in uscita per indirizzare il traffico verso i tuoi gateway NAT.

(Consigliato) Esempio di gruppo di sicurezza autoreferenziato per tutti gli accessi

L'esempio seguente mostra le regole del gruppo di sicurezza in entrata che consentono tutto il traffico per un Amazon VPC per un Amazon VPC con routing pubblico o routing privato. Il gruppo di sicurezza in questo esempio è una regola autoreferenziale a se stessa.

Type	Protocollo	Tipo di fonte	Origine		
Tutto il traffico	Tutti	Tutti	sg-0909e8e81919/-group my-mwaa-vpc-security		

L'esempio seguente mostra le regole del gruppo di sicurezza in uscita.

Type	Protocollo	Tipo di fonte	Origine		
Tutto il traffico	Tutti	Tutti	0.0.0.0/0		

(Facoltativo) Esempio di gruppo di sicurezza che limita l'accesso in entrata alla porta 5432

L'esempio seguente mostra le regole del gruppo di sicurezza in entrata che consentono tutto il traffico HTTPS sulla porta 5432 per il database di metadati PostgreSQL di Amazon Aurora (di proprietà di Amazon MWAA) per il tuo ambiente.

> **Note**
>
> Se scegli di limitare il traffico utilizzando questa regola, dovrai aggiungere un'altra regola per consentire il traffico TCP sulla porta 443.

Type	Protocollo	Intervallo porte	Tipo di origine	Origine	
TCP personalizzato	TCP	5432	Personalizza	sg-0909e8e81919/-group my-mwaa-vpc-security	

(Facoltativo) Esempio di gruppo di sicurezza che limita l'accesso in entrata alla porta 443

L'esempio seguente mostra le regole del gruppo di sicurezza in entrata che consentono tutto il traffico TCP sulla porta 443 per il server Web Apache Airflow.

Type	Protocollo	Intervallo porte	Tipo di origine	Origine	
HTTPS	TCP	443	Personalizza	my-mwaa-vpc-securitysg-0909e8e81919/-group	

Policy degli endpoint VPC (solo routing privato)

Una policy VPC endpoint (AWS PrivateLink) controlla l'accesso ai AWS servizi dalla tua sottorete privata. Una policy per gli endpoint VPC è una policy delle risorse IAM da collegare al gateway VPC o all'endpoint di interfaccia. Questa sezione descrive le autorizzazioni necessarie per le policy degli endpoint VPC per ogni endpoint VPC.

Ti consigliamo di utilizzare una policy per gli endpoint dell'interfaccia VPC per ciascuno degli endpoint VPC che hai creato che consenta l'accesso completo a tutti i AWS servizi e di utilizzare il tuo ruolo di esecuzione esclusivamente per le autorizzazioni. AWS

(Consigliato) Esempio di policy degli endpoint VPC per consentire tutti gli accessi

L'esempio seguente mostra una policy per gli endpoint dell'interfaccia VPC per un Amazon VPC con routing privato.

```
{
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

(Consigliato) Esempio di policy degli endpoint del gateway Amazon S3 per consentire l'accesso ai bucket

L'esempio seguente mostra una policy per gli endpoint del gateway VPC che fornisce l'accesso ai bucket Amazon S3 necessari per le operazioni di Amazon ECR per un Amazon VPC con routing privato. Questo è necessario per recuperare l'immagine Amazon ECR, oltre al bucket in cui sono archiviati i DAG e i file di supporto.

```
{
  "Statement": [
    {
      "Sid": "Access-to-specific-bucket-only",
      "Principal": "*",
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": ["arn:aws:s3:::prod-region-starport-layer-bucket/*"]
    }
  ]
}
```

Gestione dell'accesso agli endpoint Amazon VPC specifici del servizio su Amazon MWAA

Un endpoint VPC (AWS PrivateLink) ti consente di connettere privatamente il tuo VPC ai servizi ospitati su AWS senza richiedere un gateway Internet, un dispositivo NAT, una VPN o proxy firewall. Questi endpoint sono dispositivi virtuali scalabili orizzontalmente e ad alta disponibilità che consentono la comunicazione tra le istanze del tuo VPC e i servizi. AWS Questa pagina descrive gli endpoint VPC creati da Amazon MWAA e come accedere all'endpoint VPC per il tuo server Web Apache Airflow se hai scelto la modalità di accesso alla rete privata su Amazon Managed Workflows for Apache Airflow.

Indice

- Prezzi
- Panoramica degli endpoint VPC
  - Modalità di accesso alla rete pubblica

  - Modalità di accesso alla rete privata
- Autorizzazione all'uso di altri servizi AWS
- Visualizzazione degli endpoint VPC
  - Visualizzazione degli endpoint VPC sulla console Amazon VPC
  - Identificazione degli indirizzi IP privati del server Web Apache Airflow e del relativo endpoint VPC
- Accesso all'endpoint VPC per il server Web Apache Airflow (accesso alla rete privata)
  - Utilizzando un AWS Client VPN
  - Utilizzo di un host Linux Bastion
  - Utilizzo di un Load Balancer (avanzato)

Prezzi

- AWS PrivateLink Prezzi

Panoramica degli endpoint VPC

Quando crei un ambiente Amazon MWAA, Amazon MWAA crea da uno a due endpoint VPC per il tuo ambiente. Questi endpoint appaiono come Interfacce di rete elastiche (ENI) con IP privati nel tuo Amazon VPC. Dopo la creazione di questi endpoint, tutto il traffico destinato a questi IP viene instradato privatamente o pubblicamente ai servizi corrispondenti utilizzati dal tuo ambiente. AWS

Modalità di accesso alla rete pubblica

Se si sceglie la modalità di accesso alla rete pubblica per il server Web Apache Airflow, il traffico di rete viene instradato pubblicamente su Internet.

- Amazon MWAA crea un endpoint di interfaccia VPC per il tuo database di metadati Amazon Aurora PostgreSQL. L'endpoint viene creato nelle zone di disponibilità mappate alle sottoreti private ed è indipendente dagli altri account. AWS
- Amazon MWAA associa quindi un indirizzo IP dalle sottoreti private agli endpoint dell'interfaccia. Questo è progettato per supportare la best practice di associare un singolo IP da ogni zona di disponibilità di Amazon VPC.

Modalità di accesso alla rete privata

Se hai scelto la modalità di accesso alla rete privata per il tuo server Web Apache Airflow, il traffico di rete viene instradato privatamente all'interno del tuo Amazon VPC.

- Amazon MWAA crea un endpoint di interfaccia VPC per il server Web Apache Airflow e un endpoint di interfaccia per il database di metadati PostgreSQL di Amazon Aurora. Gli endpoint vengono creati nelle zone di disponibilità mappate alle sottoreti private e sono indipendenti dagli altri account. AWS
- Amazon MWAA associa quindi un indirizzo IP dalle sottoreti private agli endpoint dell'interfaccia. Questo è progettato per supportare la best practice di associare un singolo IP da ogni zona di disponibilità di Amazon VPC.

Autorizzazione all'uso di altri servizi AWS

Gli endpoint dell'interfaccia utilizzano il ruolo di esecuzione per l'ambiente in AWS Identity and Access Management (IAM) per gestire le autorizzazioni alle AWS risorse utilizzate dall'ambiente. Man mano che vengono abilitati più AWS servizi per un ambiente, ogni servizio richiederà di configurare l'autorizzazione utilizzando il ruolo di esecuzione dell'ambiente. Per aggiungere autorizzazioni, consultaRuolo di esecuzione di Amazon MWAA.

Se hai scelto la modalità di accesso alla rete privata per il tuo server Web Apache Airflow, devi anche consentire l'autorizzazione nella politica degli endpoint VPC per ogni endpoint. Per ulteriori informazioni, consulta the section called “Policy degli endpoint VPC (solo routing privato)”.

Visualizzazione degli endpoint VPC

Questa sezione descrive come visualizzare gli endpoint VPC creati da Amazon MWAA e come identificare gli indirizzi IP privati per il tuo endpoint VPC Apache Airflow.

Visualizzazione degli endpoint VPC sulla console Amazon VPC

La sezione seguente mostra i passaggi per visualizzare gli endpoint VPC creati da Amazon MWAA e tutti gli endpoint VPC che potresti aver creato se utilizzi il routing privato per Amazon VPC.

Per visualizzare gli endpoint VPC

1. Apri la pagina Endpoints sulla console Amazon VPC.

2. Usa il selettore AWS della regione per selezionare la tua regione.
3. Dovresti vedere gli endpoint dell'interfaccia VPC creati da Amazon MWAA e tutti gli endpoint VPC che potresti aver creato se utilizzi il routing privato nel tuo Amazon VPC.

Per ulteriori informazioni sugli endpoint del servizio VPC necessari per un Amazon VPC con routing privato, consulta. Creazione degli endpoint del servizio VPC richiesti in un Amazon VPC con routing privato

Identificazione degli indirizzi IP privati del server Web Apache Airflow e del relativo endpoint VPC

I passaggi seguenti descrivono come recuperare il nome host del server Web Apache Airflow e il relativo endpoint di interfaccia VPC e i relativi indirizzi IP privati.

1. Utilizzate il seguente comando AWS Command Line Interface (AWS CLI) per recuperare il nome host del vostro server Web Apache Airflow.

   ```
   aws mwaa get-environment --name YOUR_ENVIRONMENT_NAME --query
    'Environment.WebserverUrl'
   ```

   Dovreste vedere qualcosa di simile alla seguente risposta:

   ```
   "99aa99aa-55aa-44a1-a91f-f4552cf4e2f5-vpce.c10.us-west-2.airflow.amazonaws.com"
   ```

2. Esegui un comando dig sul nome host restituito nella risposta del comando precedente. Per esempio:

   ```
   dig CNAME +short 99aa99aa-55aa-44a1-a91f-f4552cf4e2f5-vpce.c10.us-
   west-2.airflow.amazonaws.com
   ```

   Dovresti vedere qualcosa di simile alla seguente risposta:

   ```
   vpce-0699aa333a0a0a0-bf90xjtr.vpce-svc-00bb7c2ca2213bc37.us-
   west-2.vpce.amazonaws.com.
   ```

3. Usa il seguente comando AWS Command Line Interface (AWS CLI) per recuperare il nome DNS dell'endpoint VPC restituito nella risposta del comando precedente. Per esempio:

```
aws ec2 describe-vpc-endpoints | grep vpce-0699aa333a0a0a0-bf90xjtr.vpce-
svc-00bb7c2ca2213bc37.us-west-2.vpce.amazonaws.com.
```

Dovresti vedere qualcosa di simile alla seguente risposta:

```
"DnsName": "vpce-066777a0a0a0-bf90xjtr.vpce-svc-00bb7c2ca2213bc37.us-
west-2.vpce.amazonaws.com",
```

4. Esegui un comando nslookup o dig sul nome host di Apache Airflow e sul nome DNS dell'endpoint VPC per recuperare gli indirizzi IP. Per esempio:

```
dig +short YOUR_AIRFLOW_HOST_NAME YOUR_AIRFLOW_VPC_ENDPOINT_DNS
```

Dovresti vedere qualcosa di simile alla seguente risposta:

```
192.0.5.1
  192.0.6.1
```

Accesso all'endpoint VPC per il server Web Apache Airflow (accesso alla rete privata)

Se hai scelto la modalità di accesso alla rete privata per il tuo server Web Apache Airflow, dovrai creare un meccanismo per accedere all'endpoint dell'interfaccia VPC per il tuo server Web Apache Airflow. È necessario utilizzare lo stesso Amazon VPC, lo stesso gruppo di sicurezza VPC e le stesse sottoreti private dell'ambiente Amazon MWAA per queste risorse.

Utilizzando un AWS Client VPN

AWS Client VPN è un servizio VPN gestito basato su client che consente di accedere in modo sicuro alle AWS risorse e alle risorse della rete locale. Fornisce una connessione TLS sicura da qualsiasi posizione utilizzando il client OpenVPN.

Ti consigliamo di seguire il tutorial di Amazon MWAA per configurare un Client VPN:. Tutorial: Configurazione dell'accesso alla rete privata utilizzando unAWS Client VPN

Utilizzo di un host Linux Bastion

Un bastion host è un server il cui scopo è fornire l'accesso a una rete privata da una rete esterna, ad esempio tramite Internet dal computer. Le istanze Linux si trovano in una sottorete pubblica e sono configurate con un gruppo di sicurezza che consente l'accesso SSH dal gruppo di sicurezza collegato all'istanza Amazon EC2 sottostante che esegue l'host bastion.

Ti consigliamo di seguire il tutorial di Amazon MWAA per configurare un host Linux Bastion:. Tutorial: configurazione dell'accesso alla rete privata utilizzando un host Linux Bastion

Utilizzo di un Load Balancer (avanzato)

La sezione seguente mostra le configurazioni da applicare a un Application Load Balancer.

1. Gruppi target. Dovrai utilizzare gruppi target che puntano agli indirizzi IP privati del tuo server Web Apache Airflow e del relativo endpoint di interfaccia VPC. Ti consigliamo di specificare entrambi gli indirizzi IP privati come destinazioni registrate, poiché utilizzarne solo uno può ridurre la disponibilità. Per ulteriori informazioni su come identificare gli indirizzi IP privati, vederethe section called “Identificazione degli indirizzi IP privati del server Web Apache Airflow e del relativo endpoint VPC”.
2. Codici di stato. Ti consigliamo di utilizzare `200` i codici di `302` stato nelle impostazioni del gruppo target. In caso contrario, le destinazioni potrebbero essere contrassegnate come non integre se l'endpoint VPC per il server Web Apache Airflow risponde con un errore. `302 Redirect`
3. Listener HTTPS. È necessario specificare la porta di destinazione per il server Web Apache Airflow. Per esempio:

Protocollo	Porta
HTTPS	443

4. Nuovo dominio ACM. Se desideri associare un certificato SSL/TLS a AWS Certificate Manager, dovrai creare un nuovo dominio per il listener HTTPS per il tuo sistema di bilanciamento del carico.
5. Regione del certificato ACM. Se desideri associare un certificato SSL/TLS a AWS Certificate Manager, dovrai caricarlo nella stessa AWS regione del tuo ambiente. Per esempio:

- Example regione in cui caricare il certificato

```
aws acm import-certificate --certificate fileb://Certificate.pem --certificate-
chain fileb://CertificateChain.pem --private-key fileb://PrivateKey.pem --
region us-west-2
```

Creazione degli endpoint del servizio VPC richiesti in un Amazon VPC con routing privato

Una rete Amazon VPC esistente senza accesso a Internet richiede endpoint di servizio VPC aggiuntivi (AWS PrivateLink) per utilizzare Apache Airflow su Amazon Managed Workflows for Apache Airflow. Questa pagina descrive gli endpoint VPC necessari per i AWS servizi utilizzati da Amazon MWAA, gli endpoint VPC richiesti per Apache Airflow e come creare e collegare gli endpoint VPC a un Amazon VPC esistente con routing privato.

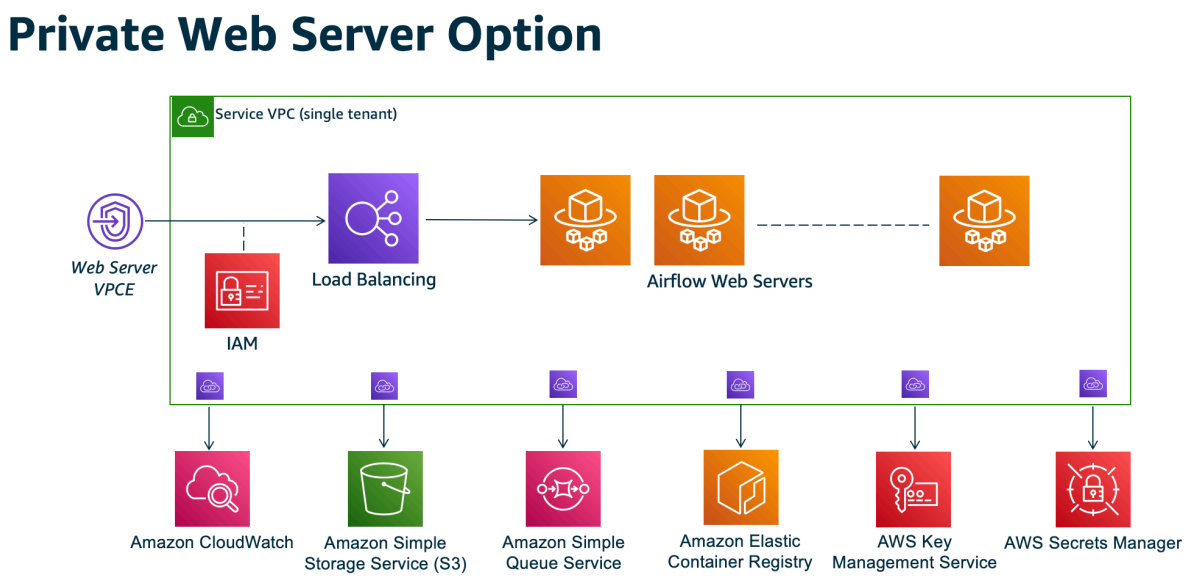
Indice

- Prezzi
- Rete privata e routing privato
- (Obbligatori) Endpoint VPC
- Collegamento degli endpoint VPC richiesti
  - Endpoint VPC necessari per i servizi AWS
  - Endpoint VPC richiesti per Apache Airflow
- (Facoltativo) Abilita gli indirizzi IP privati per l'endpoint dell'interfaccia VPC Amazon S3
  - Usare Route 53
  - VPC con DNS personalizzato

Prezzi

- AWS PrivateLink Prezzi

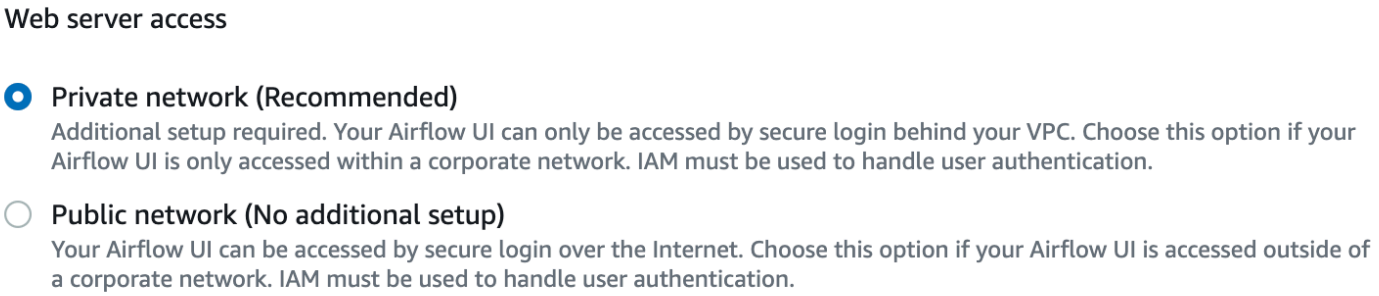
Rete privata e routing privato



La modalità di accesso alla rete privata limita l'accesso all'interfaccia utente di Apache Airflow agli utenti del tuo Amazon VPC a cui è stato concesso l'accesso alla policy IAM per il tuo ambiente.

Quando crei un ambiente con accesso privato al server web, devi impacchettare tutte le tue dipendenze in un archivio Python wheel `.whl` (), quindi fare riferimento a `.whl` nel tuo. `requirements.txt` Per istruzioni su come impacchettare e installare le dipendenze usando wheel, vedi Gestire le dipendenze usando Python wheel.

L'immagine seguente mostra dove trovare l'opzione Rete privata sulla console Amazon MWAA.



- Routing privato. Un Amazon VPC senza accesso a Internet limita il traffico di rete all'interno del VPC. Questa pagina presuppone che Amazon VPC non disponga di accesso a Internet e richieda endpoint VPC per AWS ogni servizio utilizzato dal tuo ambiente e endpoint VPC per Apache Airflow nella stessa regione e AWS Amazon VPC dell'ambiente Amazon MWAA.

(Obbligatori) Endpoint VPC

La sezione seguente mostra gli endpoint VPC necessari per un Amazon VPC senza accesso a Internet. Elenca gli endpoint VPC per ogni AWS servizio utilizzato da Amazon MWAA, inclusi gli endpoint VPC necessari per Apache Airflow.

```
com.amazonaws.YOUR_REGION.s3
com.amazonaws.YOUR_REGION.monitoring
com.amazonaws.YOUR_REGION.ecr.dkr
com.amazonaws.YOUR_REGION.ecr.api
com.amazonaws.YOUR_REGION.logs
com.amazonaws.YOUR_REGION.sqs
com.amazonaws.YOUR_REGION.kms
com.amazonaws.YOUR_REGION.airflow.api
com.amazonaws.YOUR_REGION.airflow.env
com.amazonaws.YOUR_REGION.airflow.ops
```

Collegamento degli endpoint VPC richiesti

Questa sezione descrive i passaggi per collegare gli endpoint VPC richiesti per un Amazon VPC con routing privato.

Endpoint VPC necessari per i servizi AWS

La sezione seguente mostra i passaggi per collegare gli endpoint VPC per i AWS servizi utilizzati da un ambiente a un Amazon VPC esistente.

Per collegare gli endpoint VPC alle sottoreti private

1. Apri la pagina Endpoints sulla console Amazon VPC.
2. Usa il selettore AWS della regione per selezionare la tua regione.
3. Crea l'endpoint per Amazon S3:
    a. Scegliere Create Endpoint (Crea endpoint).
    b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.s3`**, quindi premi Invio sulla tastiera.
    c. Ti consigliamo di scegliere l'endpoint di servizio elencato per il tipo di Gateway.

        Ad esempio, **`com.amazonaws.us-west-2.s3 amazon Gateway`**

d. Scegli Amazon VPC in VPC del tuo ambiente.

e. Assicurati che le tue due sottoreti private in zone di disponibilità diverse siano selezionate e che il DNS privato sia abilitato selezionando Abilita nome DNS.

f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.

g. Scegli Accesso completo in Policy.

h. Seleziona Crea endpoint.

4. Crea il primo endpoint per Amazon ECR:

a. Scegliere Create Endpoint (Crea endpoint).

b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.ecr.dkr`**, quindi premi Invio sulla tastiera.

c. Seleziona l'endpoint del servizio.

d. Scegli Amazon VPC in VPC del tuo ambiente.

e. Assicurati che le due sottoreti private in zone di disponibilità diverse siano selezionate e che l'opzione Abilita nome DNS sia abilitata.

f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.

g. Scegli Accesso completo in Policy.

h. Seleziona Crea endpoint.

5. Crea il secondo endpoint per Amazon ECR:

a. Scegliere Create Endpoint (Crea endpoint).

b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.ecr.api`**, quindi premi Invio sulla tastiera.

c. Seleziona l'endpoint del servizio.

d. Scegli Amazon VPC in VPC del tuo ambiente.

e. Assicurati che le due sottoreti private in zone di disponibilità diverse siano selezionate e che l'opzione Abilita nome DNS sia abilitata.

f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.

g. Scegli Accesso completo in Policy.

h. Seleziona Crea endpoint.

6. Crea l'endpoint per CloudWatch Logs:

a. Scegliere Create Endpoint (Crea endpoint).

b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.logs`**, quindi premi Invio sulla tastiera.
c. Seleziona l'endpoint del servizio.
d. Scegli Amazon VPC in VPC del tuo ambiente.
e. Assicurati che le due sottoreti private in zone di disponibilità diverse siano selezionate e che l'opzione Abilita nome DNS sia abilitata.
f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.
g. Scegli Accesso completo in Policy.
h. Seleziona Crea endpoint.

7. Crea l'endpoint per il CloudWatch monitoraggio:

a. Scegliere Create Endpoint (Crea endpoint).
b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.monitoring`**, quindi premi Invio sulla tastiera.
c. Seleziona l'endpoint del servizio.
d. Scegli Amazon VPC in VPC del tuo ambiente.
e. Assicurati che le due sottoreti private in zone di disponibilità diverse siano selezionate e che l'opzione Abilita nome DNS sia abilitata.
f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.
g. Scegli Accesso completo in Policy.
h. Seleziona Crea endpoint.

8. Crea l'endpoint per Amazon SQS:

a. Scegliere Create Endpoint (Crea endpoint).
b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.sqs`**, quindi premi Invio sulla tastiera.
c. Seleziona l'endpoint del servizio.
d. Scegli Amazon VPC in VPC del tuo ambiente.
e. Assicurati che le due sottoreti private in zone di disponibilità diverse siano selezionate e che l'opzione Abilita nome DNS sia abilitata.
f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.
g. Scegli Accesso completo in Policy.
h. Seleziona Crea endpoint.

9. Crea l'endpoint per AWS KMS:

    a. Scegliere Create Endpoint (Crea endpoint).

    b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.kms`**, quindi premi Invio sulla tastiera.

    c. Seleziona l'endpoint del servizio.

    d. Scegli Amazon VPC in VPC del tuo ambiente.

    e. Assicurati che le due sottoreti private in zone di disponibilità diverse siano selezionate e che l'opzione Abilita nome DNS sia abilitata.

    f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.

    g. Scegli Accesso completo in Policy.

    h. Seleziona Crea endpoint.

Endpoint VPC richiesti per Apache Airflow

La sezione seguente mostra i passaggi per collegare gli endpoint VPC per Apache Airflow a un Amazon VPC esistente.

Per collegare gli endpoint VPC alle sottoreti private

1. Apri la pagina Endpoints sulla console Amazon VPC.
2. Usa il selettore AWS della regione per selezionare la tua regione.
3. Crea l'endpoint per l'API Apache Airflow:

    a. Scegliere Create Endpoint (Crea endpoint).

    b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.airflow.api`**, quindi premi Invio sulla tastiera.

    c. Seleziona l'endpoint del servizio.

    d. Scegli Amazon VPC in VPC del tuo ambiente.

    e. Assicurati che le due sottoreti private in zone di disponibilità diverse siano selezionate e che l'opzione Abilita nome DNS sia abilitata.

    f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.

    g. Scegli Accesso completo in Policy.

    h. Seleziona Crea endpoint.

4. Crea il primo endpoint per l'ambiente Apache Airflow:
    a. Scegliere Create Endpoint (Crea endpoint).
    b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.airflow.env`**, quindi premi Invio sulla tastiera.
    c. Seleziona l'endpoint del servizio.
    d. Scegli Amazon VPC in VPC del tuo ambiente.
    e. Assicurati che le due sottoreti private in zone di disponibilità diverse siano selezionate e che l'opzione Abilita nome DNS sia abilitata.
    f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.
    g. Scegli Accesso completo in Policy.
    h. Seleziona Crea endpoint.
5. Crea il secondo endpoint per le operazioni di Apache Airflow:
    a. Scegliere Create Endpoint (Crea endpoint).
    b. Nel campo di testo Filtra per attributi o cerca per parola chiave, digita: **`.airflow.ops`**, quindi premi Invio sulla tastiera.
    c. Seleziona l'endpoint del servizio.
    d. Scegli Amazon VPC in VPC del tuo ambiente.
    e. Assicurati che le due sottoreti private in zone di disponibilità diverse siano selezionate e che l'opzione Abilita nome DNS sia abilitata.
    f. Scegli i gruppi di sicurezza Amazon VPC del tuo ambiente.
    g. Scegli Accesso completo in Policy.
    h. Seleziona Crea endpoint.

(Facoltativo) Abilita gli indirizzi IP privati per l'endpoint dell'interfaccia VPC Amazon S3

Gli endpoint dell'interfaccia Amazon S3 non supportano il DNS privato. Le richieste degli endpoint S3 vengono comunque risolte in un indirizzo IP pubblico. Per risolvere l'indirizzo S3 in un indirizzo IP privato, devi aggiungere una zona ospitata privata in Route 53 per l'endpoint regionale S3.

Usare Route 53

Questa sezione descrive i passaggi per abilitare gli indirizzi IP privati per un endpoint dell'interfaccia S3 utilizzando Route 53.

1. Crea una zona ospitata privata per il tuo endpoint di interfaccia VPC Amazon S3 (ad esempio, s3.eu-west-1.amazonaws.com) e associala al tuo Amazon VPC.
2. Crea un ALIAS Un record per il tuo endpoint di interfaccia VPC Amazon S3 (ad esempio, s3.eu-west-1.amazonaws.com) che si risolve nel nome DNS dell'endpoint dell'interfaccia VPC.
3. Crea un ALIAS Un record wildcard per l'endpoint dell'interfaccia Amazon S3 (ad esempio*. s3.eu-west-1.amazonaws.com) che si risolve nel nome DNS dell'endpoint dell'interfaccia VPC.

VPC con DNS personalizzato

Se il tuo Amazon VPC utilizza un routing DNS personalizzato, devi apportare le modifiche nel tuo resolver DNS (non Route 53, in genere un'istanza EC2 che esegue un server DNS) creando un record CNAME. Per esempio:

```
Name: s3.us-west-2.amazonaws.com
Type: CNAME
Value:  *.vpce-0f67d23e37648915c-e2q2e2j3.s3.us-west-2.vpce.amazonaws.com
```

Gestione degli endpoint Amazon VPC personali su Amazon MWAA

Amazon MWAA utilizza gli endpoint Amazon VPC per l'integrazione con vari AWS servizi necessari per configurare un ambiente Apache Airflow. La gestione dei propri endpoint ha due casi d'uso principali:

1. Significa che puoi creare ambienti Apache Airflow in un Amazon VPC condiviso quando usi un AWS Organizationsper gestire più AWS account e condividere risorse.
2. Ti consente di utilizzare politiche di accesso più restrittive restringendo le autorizzazioni alle risorse specifiche che utilizzano i tuoi endpoint.

Se scegli di gestire i tuoi endpoint VPC, sei responsabile della creazione dei tuoi endpoint per l'ambiente RDS per il database PostgreSQL e per l'ambiente web server.

Per ulteriori informazioni su come Amazon MWAA implementa Apache Airflow nel cloud, consulta il diagramma dell'architettura di Amazon MWAA.

Creazione di un ambiente in un Amazon VPC condiviso

Se gestisci più AWS account che condividono risorse, puoi utilizzare gli endpoint VPC gestiti dai clienti con Amazon MWAA per condividere le risorse dell'ambiente con un altro account della tua organizzazione. AWS Organizations

Quando configuri l'accesso VPC condiviso, l'account che possiede l'Amazon VPC principale (proprietario) condivide le due sottoreti private richieste da Amazon MWAA con altri account (partecipanti) che appartengono alla stessa organizzazione. Gli account dei partecipanti che condividono tali sottoreti possono visualizzare, creare, modificare ed eliminare ambienti nell'Amazon VPC condiviso.

Supponiamo di avere un account`0wner`, che funge da `Root` account nell'organizzazione e possiede le risorse Amazon VPC, e un account partecipante`Participant`, membro della stessa organizzazione. Quando `Participant` crea un nuovo Amazon MWAA in Amazon VPC con cui lo condivide, `0wner` Amazon MWAA crea prima le risorse VPC del servizio, quindi entra in uno stato per un massimo di 72 ore. `PENDING`

Dopo che lo stato dell'ambiente cambia da `CREATING` a`PENDING`, un principale che agisce per conto di crea gli endpoint richiesti. `0wner` A tale scopo, Amazon MWAA elenca l'endpoint del database e del server Web nella console Amazon MWAA. Puoi anche richiamare l'azione `GetEnvironment`API per ottenere gli endpoint del servizio.

> **Note**
>
> Se l'Amazon VPC che usi per condividere risorse è un Amazon VPC privato, devi comunque completare i passaggi descritti in. the section called “Gestione dell'accesso agli endpoint VPC” L'argomento tratta la configurazione di un set diverso di endpoint Amazon VPC relativi ad altri AWS servizi con cui AWS si integra, come Amazon ECR, Amazon ECS e Amazon SQS. Questi servizi sono essenziali per il funzionamento e la gestione dell'ambiente Apache Airflow nel cloud.

Prerequisiti

Prima di creare un ambiente Amazon MWAA in un VPC condiviso, sono necessarie le seguenti risorse:

- Un AWS account, `Owner` da utilizzare come account proprietario di Amazon VPC.
- Un'unità AWS Organizationsorganizzativa, `MyOrganization` creata come root.
- Un secondo AWS account`Participant`, destinato `MyOrganization` a servire l'account del partecipante che crea il nuovo ambiente.

Inoltre, ti consigliamo di acquisire familiarità con le responsabilità e le autorizzazioni dei proprietari e dei partecipanti quando condividono risorse in Amazon VPC.

Crea Amazon VPC

Innanzitutto, crea un nuovo Amazon VPC che gli account del proprietario e del partecipante condivideranno:

1. Accedi alla console utilizzando`Owner`, quindi, apri la AWS CloudFormation console. Usa il seguente modello per creare uno stack. Questo stack fornisce una serie di risorse di rete, tra cui un Amazon VPC e le sottoreti che i due account condivideranno in questo scenario.

```
AWSTemplateFormatVersion: "2010-09-09"
Description: >-
  This template deploys a VPC, with a pair of public and private subnets spread
  across two Availability Zones. It deploys an internet gateway, with a default
  route on the public subnets. It deploys a pair of NAT gateways (one in each
  AZ), and default routes for them in the private subnets.
Parameters:
  EnvironmentName:
    Description: An environment name that is prefixed to resource names
    Type: String
    Default: mwaa-
  VpcCIDR:
    Description: Please enter the IP range (CIDR notation) for this VPC
    Type: String
    Default: 10.192.0.0/16
  PublicSubnet1CIDR:
    Description: >-
      Please enter the IP range (CIDR notation) for the public subnet in the
      first Availability Zone
```

```
    Type: String
    Default: 10.192.10.0/24
  PublicSubnet2CIDR:
    Description: >-
      Please enter the IP range (CIDR notation) for the public subnet in the
      second Availability Zone
    Type: String
    Default: 10.192.11.0/24
  PrivateSubnet1CIDR:
    Description: >-
      Please enter the IP range (CIDR notation) for the private subnet in the
      first Availability Zone
    Type: String
    Default: 10.192.20.0/24
  PrivateSubnet2CIDR:
    Description: >-
      Please enter the IP range (CIDR notation) for the private subnet in the
      second Availability Zone
    Type: String
    Default: 10.192.21.0/24
Resources:
  VPC:
    Type: 'AWS::EC2::VPC'
    Properties:
      CidrBlock: !Ref VpcCIDR
      EnableDnsSupport: true
      EnableDnsHostnames: true
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName
  InternetGateway:
    Type: 'AWS::EC2::InternetGateway'
    Properties:
      Tags:
        - Key: Name
          Value: !Ref EnvironmentName
  InternetGatewayAttachment:
    Type: 'AWS::EC2::VPCGatewayAttachment'
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC
  PublicSubnet1:
    Type: 'AWS::EC2::Subnet'
    Properties:
```

```
      VpcId: !Ref VPC
      AvailabilityZone: !Select
        - 0
        - !GetAZs ''
      CidrBlock: !Ref PublicSubnet1CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub '${EnvironmentName} Public Subnet (AZ1)'
  PublicSubnet2:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select
        - 1
        - !GetAZs ''
      CidrBlock: !Ref PublicSubnet2CIDR
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub '${EnvironmentName} Public Subnet (AZ2)'
  PrivateSubnet1:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select
        - 0
        - !GetAZs ''
      CidrBlock: !Ref PrivateSubnet1CIDR
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
          Value: !Sub '${EnvironmentName} Private Subnet (AZ1)'
  PrivateSubnet2:
    Type: 'AWS::EC2::Subnet'
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select
        - 1
        - !GetAZs ''
      CidrBlock: !Ref PrivateSubnet2CIDR
      MapPublicIpOnLaunch: false
      Tags:
        - Key: Name
```

```
        Value: !Sub '${EnvironmentName} Private Subnet (AZ2)'
NatGateway1EIP:
  Type: 'AWS::EC2::EIP'
  DependsOn: InternetGatewayAttachment
  Properties:
    Domain: vpc
NatGateway2EIP:
  Type: 'AWS::EC2::EIP'
  DependsOn: InternetGatewayAttachment
  Properties:
    Domain: vpc
NatGateway1:
  Type: 'AWS::EC2::NatGateway'
  Properties:
    AllocationId: !GetAtt NatGateway1EIP.AllocationId
    SubnetId: !Ref PublicSubnet1
NatGateway2:
  Type: 'AWS::EC2::NatGateway'
  Properties:
    AllocationId: !GetAtt NatGateway2EIP.AllocationId
    SubnetId: !Ref PublicSubnet2
PublicRouteTable:
  Type: 'AWS::EC2::RouteTable'
  Properties:
    VpcId: !Ref VPC
    Tags:
      - Key: Name
        Value: !Sub '${EnvironmentName} Public Routes'
DefaultPublicRoute:
  Type: 'AWS::EC2::Route'
  DependsOn: InternetGatewayAttachment
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway
PublicSubnet1RouteTableAssociation:
  Type: 'AWS::EC2::SubnetRouteTableAssociation'
  Properties:
    RouteTableId: !Ref PublicRouteTable
    SubnetId: !Ref PublicSubnet1
PublicSubnet2RouteTableAssociation:
  Type: 'AWS::EC2::SubnetRouteTableAssociation'
  Properties:
    RouteTableId: !Ref PublicRouteTable
```

```
      SubnetId: !Ref PublicSubnet2
  PrivateRouteTable1:
    Type: 'AWS::EC2::RouteTable'
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${EnvironmentName} Private Routes (AZ1)'
  DefaultPrivateRoute1:
    Type: 'AWS::EC2::Route'
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway1
  PrivateSubnet1RouteTableAssociation:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      SubnetId: !Ref PrivateSubnet1
  PrivateRouteTable2:
    Type: 'AWS::EC2::RouteTable'
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${EnvironmentName} Private Routes (AZ2)'
  DefaultPrivateRoute2:
    Type: 'AWS::EC2::Route'
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway2
  PrivateSubnet2RouteTableAssociation:
    Type: 'AWS::EC2::SubnetRouteTableAssociation'
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      SubnetId: !Ref PrivateSubnet2
  SecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupName: mwaa-security-group
      GroupDescription: Security group with a self-referencing inbound rule.
      VpcId: !Ref VPC
  SecurityGroupIngress:
```

```
    Type: 'AWS::EC2::SecurityGroupIngress'
    Properties:
      GroupId: !Ref SecurityGroup
      IpProtocol: '-1'
      SourceSecurityGroupId: !Ref SecurityGroup
Outputs:
  VPC:
    Description: A reference to the created VPC
    Value: !Ref VPC
  PublicSubnets:
    Description: A list of the public subnets
    Value: !Join
      - ','
      - - !Ref PublicSubnet1
        - !Ref PublicSubnet2
  PrivateSubnets:
    Description: A list of the private subnets
    Value: !Join
      - ','
      - - !Ref PrivateSubnet1
        - !Ref PrivateSubnet2
  PublicSubnet1:
    Description: A reference to the public subnet in the 1st Availability Zone
    Value: !Ref PublicSubnet1
  PublicSubnet2:
    Description: A reference to the public subnet in the 2nd Availability Zone
    Value: !Ref PublicSubnet2
  PrivateSubnet1:
    Description: A reference to the private subnet in the 1st Availability Zone
    Value: !Ref PrivateSubnet1
  PrivateSubnet2:
    Description: A reference to the private subnet in the 2nd Availability Zone
    Value: !Ref PrivateSubnet2
  SecurityGroupIngress:
    Description: Security group with self-referencing inbound rule
    Value: !Ref SecurityGroupIngress
```

2. Dopo aver effettuato il provisioning delle nuove risorse Amazon VPC, accedi alla AWS Resource Access Manager console, quindi scegli Crea condivisione di risorse.
3. Scegli le sottoreti che hai creato nel primo passaggio dall'elenco delle sottoreti disponibili con cui puoi condividere. `Participant`

Creazione dell'ambiente

Completa i seguenti passaggi per creare un ambiente Amazon MWAA con endpoint Amazon VPC gestiti dal cliente.

1. Accedi utilizzando `Participant` e apri la console Amazon MWAA. Completa la prima fase: specifica i dettagli per specificare un bucket Amazon S3, una cartella DAG e le dipendenze per il tuo nuovo ambiente. Per ulteriori informazioni, consulta la sezione Guida introduttiva.
2. Nella pagina Configura impostazioni avanzate, in Rete, scegli le sottoreti dall'Amazon VPC condiviso.
3. In Gestione degli endpoint, scegli CLIENTE dall'elenco a discesa.
4. Mantieni l'impostazione predefinita per le opzioni rimanenti sulla pagina, quindi scegli Crea ambiente nella pagina Rivedi e crea.

L'ambiente inizia in uno `CREATING` stato, quindi diventa`PENDING`. Quando l'ambiente lo è`PENDING`, annota il nome del servizio endpoint del database e il nome del servizio endpoint del server Web (se hai configurato un server Web privato) utilizzando la console.

Quando crei un nuovo ambiente utilizzando la console Amazon MWAA. Amazon MWAA crea un nuovo gruppo di sicurezza con le regole in entrata e in uscita richieste. Annotare l'ID del gruppo di sicurezza.

Nella prossima sezione, `0wner` utilizzerà gli endpoint del servizio e l'ID del gruppo di sicurezza per creare nuovi endpoint Amazon VPC nell'Amazon VPC condiviso.

Crea gli endpoint Amazon VPC

Completa i seguenti passaggi per creare gli endpoint Amazon VPC richiesti per il tuo ambiente.

1. Accedi a AWS Management Console Using`0wner`, quindi apri https://console.aws.amazon.com/vpc/.
2. Scegli Gruppi di sicurezza dal pannello di navigazione a sinistra, quindi crea un nuovo gruppo di sicurezza nell'Amazon VPC condiviso utilizzando le seguenti regole in entrata e in uscita:

	Type	Protocollo	Tipo di origine	Origine
In entrata	Tutto il traffico	Tutti	Tutti	Il tuo gruppo di sicurezza ambientale
In uscita	Tutto il traffico	Tutti	Tutti	`0.0.0.0/0`

> **Warning**
>
> L'`0wner`account deve configurare un gruppo di sicurezza nell'`0wner`account per consentire il traffico dal nuovo ambiente all'Amazon VPC condiviso. Puoi farlo creando un nuovo gruppo di sicurezza in `0wner` o modificandone uno esistente.

3. Scegli Endpoints, quindi crea nuovi endpoint per il database dell'ambiente e il server Web (se in modalità privata) utilizzando i nomi dei servizi endpoint dei passaggi precedenti. Scegli l'Amazon VPC condiviso, le sottoreti che hai usato per l'ambiente e il gruppo di sicurezza dell'ambiente.

In caso di successo, l'ambiente cambierà da zero `PENDING` a`CREATING`, poi finalmente a. `AVAILABLE` Quando lo è`AVAILABLE`, puoi accedere alla console Apache Airflow.

Risoluzione dei problemi con Amazon VPC condiviso

Utilizza il seguente riferimento per risolvere i problemi riscontrati durante la creazione di ambienti in un Amazon VPC condiviso.

Ambiente in **`CREATE_FAILED`** stato successivo **`PENDING`**

- Verifica che `0wner` stia condividendo le sottoreti con `Participant` using. AWS Resource Access Manager
- Verifica che gli endpoint Amazon VPC per il database e il server Web siano creati nelle stesse sottoreti associate all'ambiente.
- Verifica che il gruppo di sicurezza utilizzato con i tuoi endpoint consenta il traffico proveniente dai gruppi di sicurezza utilizzati per l'ambiente. L'`0wner`account crea regole che fanno riferimento al gruppo di sicurezza `Participant` come*`account-number`*/*`security-group-id`*:.

Type	Protocollo	Tipo di origine	Origine
Tutto il traffico	Tutti	Tutti	*123456789012 /sg-0909e8e81919*

Per ulteriori informazioni, vedere Responsabilità e autorizzazioni per proprietari e partecipanti

Ambiente bloccato **`PENDING`**

Verifica lo stato di ogni endpoint VPC per assicurarti che lo sia. `Available` Se configuri un ambiente con un server web privato, devi anche creare un endpoint per il server web. Se l'ambiente è bloccato`PENDING`, ciò potrebbe indicare che manca l'endpoint del server Web privato.

Errore ricevuto **`The Vpc Endpoint Service '`*`vpce-service-name`*`' does not exist`**

Se visualizzi il seguente errore, verifica che l'account che crea gli endpoint sia nell'0`wner`account proprietario del VPC condiviso:

```
ClientError: An error occurred (InvalidServiceName) when calling the
 CreateVpcEndpoint operation:

The Vpc Endpoint Service 'vpce-service-name' does not exist
```

Tutorial per Amazon Managed Workflows per Apache Airflow

Questa guida include step-by-step tutorial sull'uso e la configurazione di un ambiente Amazon Managed Workflows for Apache Airflow.

Argomenti

- Tutorial: Configurazione dell'accesso alla rete privata utilizzando unAWS Client VPN
- Tutorial: configurazione dell'accesso alla rete privata utilizzando un host Linux Bastion
- Tutorial: limitazione dell'accesso di un utente Amazon MWAA a un sottoinsieme di DAG
- Tutorial: automatizza la gestione degli endpoint del tuo ambiente su Amazon MWAA

Tutorial: Configurazione dell'accesso alla rete privata utilizzando unAWS Client VPN

Questo tutorial illustra i passaggi per creare un tunnel VPN dal tuo computer al server Web Apache Airflow per il tuo ambiente Amazon Managed Workflow for Apache Airflow. Per connetterti a Internet tramite un tunnel VPN, devi prima creare unAWS Client VPN endpoint. Una volta configurato, un endpoint Client VPN funge da server VPN che consente una connessione sicura dal tuo computer alle risorse del tuo VPC. Ti connetterai quindi al Client VPN dal tuo computer utilizzando AWS Client VPNfor Desktop.

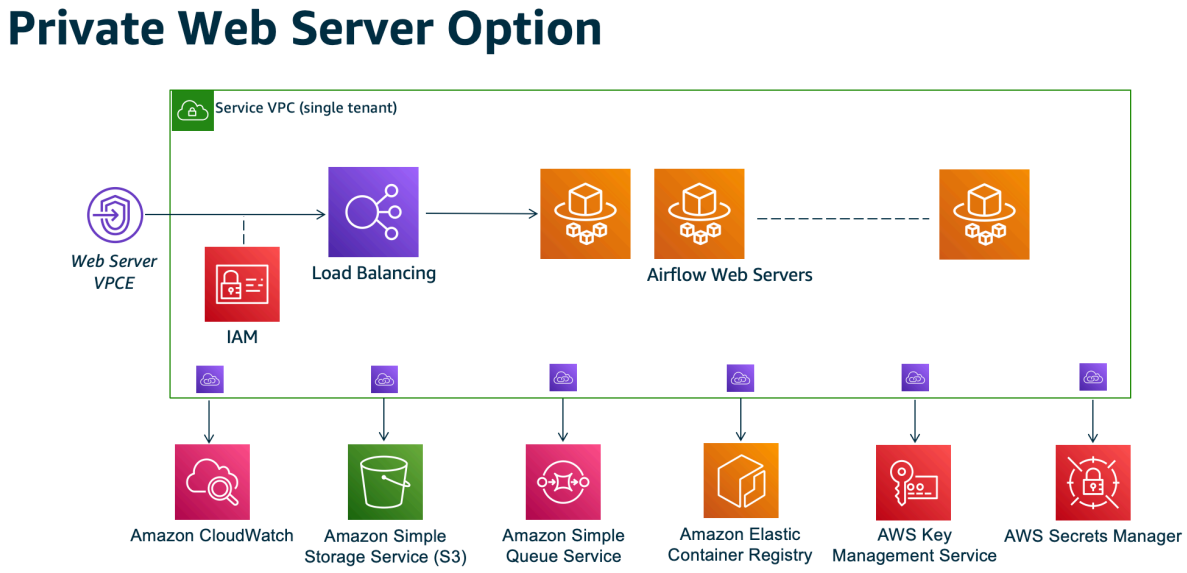
Sezioni

- Rete privata privata privata
- Casi d'uso
- Prima di iniziare
- Obiettivi
- (Facoltativo) Fase uno: identificazione del VPC, delle regole CIDR e delle protezioni VPC
- Fase 2: Creazione dei certificati server e client e client e caricare i certificati server e client
- Fase tre: salvare ilAWS CloudFormation modello localmente
- Fase quattro: creare loAWS CloudFormation stack Client VPN
- Fase cinque: associa le sottoreti alla tua Client VPN
- Fase sei: aggiungi una regola di accesso di autorizzazione alla tua Client VPN

- Fase 7: Scaricare il file di configurazione dell'endpoint Client VPN VPN Client VPN VPN Client VPN VPN
- Fase otto: Connect alAWS Client VPN
- Fasi successive

Rete privata privata privata

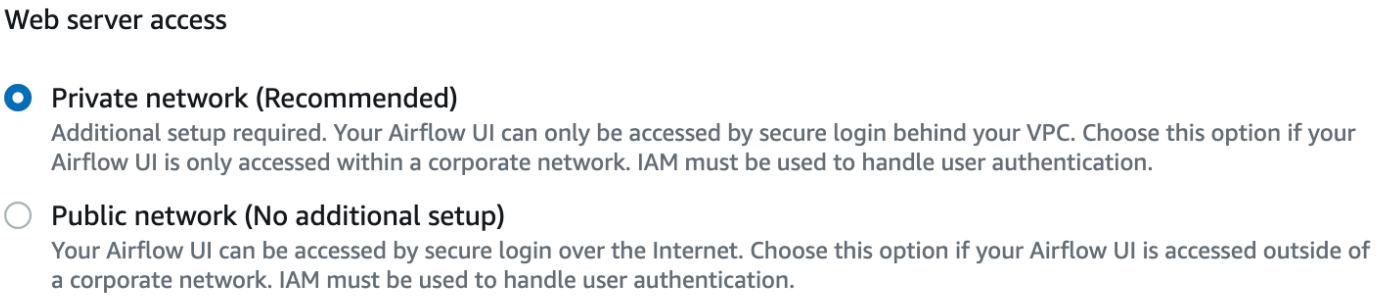
Questo tutorial presuppone che tu abbia scelto la modalità di accesso alla rete privata per il tuo server Web Apache Airflow.



La modalità di accesso alla rete privata limita l'accesso all'interfaccia utente di Apache Airflow agli utenti all'interno del tuo Amazon VPC a cui è stato concesso l'accesso alla policy IAM per il tuo ambiente.

Quando crei un ambiente con accesso privato al server web, devi impacchettare tutte le tue dipendenze in un archivio Python wheel (`.whl`), quindi fare riferimento a quelle`.whl` nel tuo`requirements.txt`. Per istruzioni sulla creazione di pacchetti e sull'installazione delle dipendenze utilizzando wheel, vedere Gestire le dipendenze utilizzando la ruota Python.

L'immagine seguente mostra dove trovare l'opzione Rete privata sulla console Amazon MWAA.



Casi d'uso

Puoi usare questo tutorial prima o dopo aver creato un ambiente Amazon MWAA. Devi utilizzare gli stessi Amazon VPC, gli stessi gruppi di sicurezza VPC e le stesse sottoreti private del tuo ambiente. Se utilizzi questo tutorial dopo aver creato un ambiente Amazon MWAA, una volta completati i passaggi, puoi tornare alla console Amazon MWAA e modificare la modalità di accesso al server Web Apache Airflow in Rete privata.

Prima di iniziare

1. Verifica le autorizzazioni degli utenti. Assicurati che il tuo account inAWS Identity and Access Management (IAM) disponga di autorizzazioni sufficienti per creare e gestire le risorse VPC.
2. Usa il tuo Amazon MWAA VPC. Questo tutorial presuppone che tu stia associando Client VPN a un VPC esistente. Amazon VPC deve trovarsi nella stessaAWS regione di un ambiente Amazon MWAA e avere due sottoreti private. Se non hai creato un Amazon VPC, utilizza ilAWS CloudFormation modello in[Opzione tre: creazione di una rete Amazon VPCsenzaaccesso a Internet](#).

Obiettivi

In questo tutorial, verranno eseguite le operazioni seguenti:

1. Crea unAWS Client VPN endpoint utilizzando unAWS CloudFormation modello per un Amazon VPC esistente.
2. Generare i certificati e le chiavi server e client e caricare il certificato e la chiave serverAWS Certificate Manager nella stessaAWS regione in cui si trova un ambiente Amazon MWAA.
3. Scarica e modifica un file di configurazione dell'endpoint Client VPN per Client VPN e utilizza il file per creare un profilo VPN per connetterti utilizzando Client VPN for Desktop.

(Facoltativo) Fase uno: identificazione del VPC, delle regole CIDR e delle protezioni VPC

La sezione seguente descrive come trovare gli ID per il tuo Amazon VPC, il gruppo di sicurezza VPC e un modo per identificare le regole CIDR necessarie per creare il tuo Client VPN nei passaggi successivi.

Identifica le tue regole CIDR

La sezione seguente mostra come identificare le regole CIDR, necessarie per creare il tuo Client VPN.

Per identificare il CIDR per il tuo Client VPN

1. Apri la pagina I tuoi Amazon VPC nella console Amazon VPC nella console Amazon VPC.
2. Utilizza il selettore di regione nella barra di navigazione per scegliere la stessaAWS regione di un ambiente Amazon MWAA.
3. Scegli Amazon VPC.
4. Supponendo che i CIDR per le tue sottoreti private siano:

   - Sottorete privata 1:10.192.10.0/24
   - Sottorete privata 2:10.192.11.0/24

   Se il CIDR per Amazon VPC è 10.192.0.0/16, il CIDR IPv4 del client che dovresti specificare per la tua Client VPN sarà 10.192.0.0/22.
5. Salva questo valore CIDR e il valore del tuo ID VPC per i passaggi successivi.

Identifica il tuo VPC e i tuoi gruppi di sicurezza

La sezione seguente mostra come trovare l'ID del tuo Amazon VPC e dei gruppi di sicurezza necessari per creare la tua Client VPN.

> **Note**
>
> Potresti utilizzare più di un gruppo di sicurezza. Dovrai specificare tutti i gruppi di sicurezza del tuo VPC nei passaggi successivi.

Per identificare il/i gruppo/i di sicurezza

1. Aprire la pagina Security Gruppi di sicurezza nella console Amazon VPC.
2. Utilizzare il selettore delle regioni nella barra di navigazione per scegliere laAWS regione.
3. Cerca Amazon VPC nell'ID VPC e identifica i gruppi di sicurezza associati al VPC.
4. Salva l'ID dei tuoi gruppi di sicurezza e del cloud privato VPC per i passaggi successivi.

Fase 2: Creazione dei certificati server e client e client e caricare i certificati server e client

Un endpoint Client VPN supporta solo chiavi RSA a 1024 bit e 2048 bit. La seguente sezione mostra come utilizzare OpenVPN easy-rsa per generare i certificati e le chiavi server e client e caricare i certificati in ACM utilizzando ilAWS Command Line Interface (AWS CLI).

Per creare i certificati client

1. Segui questi rapidi passaggi per creare e caricare i certificati su ACM tramite Autenticazione e autorizzazione del client: Autenticazione reciproca.AWS CLI
2. In questi passaggi, devi specificare la stessaAWS regione di un ambiente Amazon MWAA nelAWS CLI comando quando carichi i certificati server e client. Di seguito sono riportati alcuni esempi di come specificare la regione in questi comandi:

   a. Example regione per il certificato server server per server server

   ```
   aws acm import-certificate --certificate fileb://server.crt --private-key
    fileb://server.key --certificate-chain fileb://ca.crt --region us-west-2
   ```

   b. Example regione per il certificato client per il certificato client

   ```
   aws acm import-certificate --certificate fileb://client1.domain.tld.crt
    --private-key fileb://client1.domain.tld.key --certificate-chain fileb://
   ca.crt --region us-west-2
   ```

   c. Dopo questi passaggi, salva il valore restituito nellaAWS CLI risposta per gli ARN del certificato del server e del certificato client. Specificherai questi ARN nel tuoAWS CloudFormation modello per creare il Client VPN.
3. In questi passaggi, un certificato client e una chiave privata vengono salvati sul computer. Di seguito è riportato un esempio di dove trovare queste credenziali:

a. Example in macOS OS OS OS OS

   Su macOS i contenuti vengono salvati in`/Users/`*`youruser`*`/custom_folder`. Se elenchi tutti i contenuti (`ls -a`) di questa directory, dovresti vedere qualcosa di simile al seguente:

```
.
..
ca.crt
client1.domain.tld.crt
client1.domain.tld.key
server.crt
server.key
```

b. Dopo questi passaggi, salva il contenuto o annota la posizione del certificato client e la chiave privata`client1.domain.tld.key`.`client1.domain.tld.crt` Aggiungerai questi valori al file di configurazione per il tuo Client VPN.

Fase tre: salvare ilAWS CloudFormation modello localmente

La sezione seguente contiene ilAWS CloudFormation modello per creare il Client VPN. Devi specificare gli stessi Amazon VPC, gli stessi gruppi di sicurezza VPC e le stesse sottoreti private del tuo ambiente Amazon MWAA.

- Copiare il contenuto del seguente modello e salvare localmente come`mwaa_vpn_client.yaml`. Puoi anche scaricare il modello.

  Sostituisci i seguenti valori:

  - **`YOUR_CLIENT_ROOT_CERTIFICATE_ARN`**— L'ARN del tuo certificato client1.domain.tld in`ClientRootCertificateChainArn`.
  - **`YOUR_SERVER_CERTIFICATE_ARN`**— L'ARN per il certificato del server in`ServerCertificateArn`.
  - La regola CIDR IPv4 del client in`ClientCidrBlock`. Viene fornita una regola CIDR di`10.192.0.0/22`.
  - Il tuo ID Amazon VPC è inserito`VpcId`. Viene fornito un VPC di`vpc-010101010101`.
  - I tuoi ID del gruppo di sicurezza VPC in`SecurityGroupIds`. Viene fornito un gruppo`sg-0101010101` di sicurezza di.

```
AWSTemplateFormatVersion: 2010-09-09
Description: This template deploys a VPN Client Endpoint.
Resources:
  ClientVpnEndpoint:
    Type: 'AWS::EC2::ClientVpnEndpoint'
    Properties:
      AuthenticationOptions:
        - Type: "certificate-authentication"
          MutualAuthentication:
            ClientRootCertificateChainArn: "YOUR_CLIENT_ROOT_CERTIFICATE_ARN"
      ClientCidrBlock: 10.192.0.0/22
      ClientConnectOptions:
        Enabled: false
      ConnectionLogOptions:
        Enabled: false
      Description: "MWAA Client VPN"
      DnsServers: []
      SecurityGroupIds:
        - sg-0101010101
      SelfServicePortal: ''
      ServerCertificateArn: "YOUR_SERVER_CERTIFICATE_ARN"
      SplitTunnel: true
      TagSpecifications:
        - ResourceType: "client-vpn-endpoint"
          Tags:
          - Key: Name
            Value: MWAA-Client-VPN
      TransportProtocol: udp
      VpcId: vpc-010101010101
      VpnPort: 443
```

Note

Se utilizzi più di un gruppo di sicurezza per il tuo ambiente, puoi specificare più gruppi di sicurezza nel seguente formato:

```
SecurityGroupIds:
      - sg-0112233445566778b
```

```
    - sg-0223344556677889f
```

Fase quattro: creare loAWS CloudFormation stack Client VPN

Per creare il AWS Client VPN

1. Aprire la console AWS CloudFormation.
2. Scegli il modello è pronto, carica un file modello.
3. Scegli Scegli file e seleziona il tuo`mwaa_vpn_client.yaml` file.
4.
5. Scegli Avanti, Avanti.
6. Seleziona il riconoscimento, quindi scegli Crea pila.

Fase cinque: associa le sottoreti alla tua Client VPN

Per associare sottoreti private aAWS Client VPN

1. Apri la console Amazon VPC.
2. Scegli la pagina Client VPN Endpoints.
3. Seleziona il tuo Client VPN, quindi scegli la scheda Associazioni, Associa.
4. Scegli quanto segue nell'elenco a discesa:
   - Il tuo Amazon VPC in VPC.
   - Una delle tue sottoreti private in Scegli una sottorete da associare.
5. Selezionare Associate (Associa).

> **Note**
>
> Per associare il VPC e la sottorete alla Client VPN client sono necessari alcuni minuti.

Fase sei: aggiungi una regola di accesso di autorizzazione alla tua Client VPN

Devi aggiungere una regola di accesso di autorizzazione utilizzando la regola CIDR per il tuo VPC alla tua Client VPN. Se desideri autorizzare utenti o gruppi specifici dal tuo Active Directory Group o Identity Provider (IdP) basato su SAML, consulta le regole di autorizzazione nella guida Client VPN.

Per aggiungere il CIDR alAWS Client VPN

1. Apri la console Amazon VPC.
2. Scegli la pagina Client VPN Endpoints.
3. Seleziona il tuo Client VPN, quindi scegli la scheda Autorizzazione, Autorizza l'ingresso.
4. Specificare le impostazioni seguenti:
   - La regola CIDR di Amazon VPC nella rete di destinazione da abilitare. Ad esempio:

     ```
     10.192.0.0/16
     ```

   - Scegli Consenti l'accesso a tutti gli utenti in Concedi accesso a.
   - Inserire un nome descrittivo in Descrizione.
5. Scegli Aggiungi regola di autorizzazione.

> **Note**
>
> A seconda dei componenti di rete del tuo Amazon VPC, potresti aver bisogno anche di questa regola di autorizzazione per l'ingresso alla tua lista di controllo degli accessi alla rete (NACL).

Fase 7: Scaricare il file di configurazione dell'endpoint Client VPN VPN Client VPN VPN Client VPN VPN

Per scaricare il file di configurazione

1. Segui questi rapidi passaggi per scaricare il file di configurazione Client VPN all'indirizzo Scarica il file di configurazione dell'endpoint Client VPN.

2. In questi passaggi, ti viene chiesto di aggiungere una stringa al nome DNS dell'endpoint Client VPN. Ecco un esempio:

   - Example nome DNS dell'endpoint

     Se il nome DNS dell'endpoint Client VPN ha il seguente aspetto:

     ```
     remote cvpn-endpoint-0909091212aaee1.prod.clientvpn.us-west-1.amazonaws.com 443
     ```

     Puoi aggiungere una stringa per identificare il tuo endpoint Client VPN in questo modo:

     ```
     remote mwaavpn.cvpn-endpoint-0909091212aaee1.prod.clientvpn.us-
     west-1.amazonaws.com 443
     ```

3. In questi passaggi ti viene chiesto di aggiungere il contenuto del certificato client tra un nuovo set di`<cert></cert>` tag e il contenuto della chiave privata tra un nuovo set di`<key></key>` tag. Ecco un esempio:

   a. Apri un prompt dei comandi e modifica le directory nella posizione del certificato client e della chiave privata.

   b. Example Client macOS 1.domain.tld.crt

      Per mostrare il contenuto del`client1.domain.tld.crt` file su macOS, puoi usare`cat client1.domain.tld.crt`.

      Copia il valore dal terminale e incollalo in`downloaded-client-config.ovpn` questo modo:

      ```
      ZZZ1111dddaBBB
      -----END CERTIFICATE-----
      </ca>
      <cert>
      -----BEGIN CERTIFICATE-----
      YOUR client1.domain.tld.crt
      -----END CERTIFICATE-----
      </cert>
      ```

   c. Example client1.domain.tld.key per macOS

      Per mostrare il contenuto di`client1.domain.tld.key`, puoi usare`cat client1.domain.tld.key`.

Copia il valore dal terminale e incollalo in`downloaded-client-config.ovpn` questo modo:

```
ZZZ1111dddaBBB
-----END CERTIFICATE-----
</ca>
<cert>
-----BEGIN CERTIFICATE-----
YOUR client1.domain.tld.crt
-----END CERTIFICATE-----
</cert>
<key>
-----BEGIN CERTIFICATE-----
YOUR client1.domain.tld.key
-----END CERTIFICATE-----
</key>
```

Fase otto: Connect alAWS Client VPN

Il client perAWS Client VPN il client è fornito gratuitamente. Puoi connettere direttamente il tuo computerAWS Client VPN per un'esperienza VPN end-to-end.

Per connettersi alla Client VPN Client VPN Client VPN

1. Scarica e installa la versioneAWS Client VPN per desktop.
2. Aprire AWS Client VPN.
3. Scegli File, Profili gestiti nel menu del client VPN.
4. Scegli Aggiungi profilo, quindi scegli il`downloaded-client-config.ovpn`.
5. Immettete un nome descrittivo in Nome visualizzato.
6. Scegli Aggiungi profilo, Fine.
7. Scegli Connect (Connetti).

Dopo esserti connesso alla Client VPN, dovrai disconnetterti dalle altre VPN per visualizzare le risorse del tuo Amazon VPC.

> **Note**
>
> Potrebbe essere necessario chiudere il client e ricominciare prima di poter connettersi.

Fasi successive

- Informazioni sulla creazione di un ambiente Amazon MWAA inInizia a usare Amazon Managed Workflows per Apache Airflow. È necessario creare un ambiente nella stessaAWS regione della Client VPN e utilizzare lo stesso VPC, sottoreti private e gruppo di sicurezza della Client VPN.

Tutorial: configurazione dell'accesso alla rete privata utilizzando un host Linux Bastion

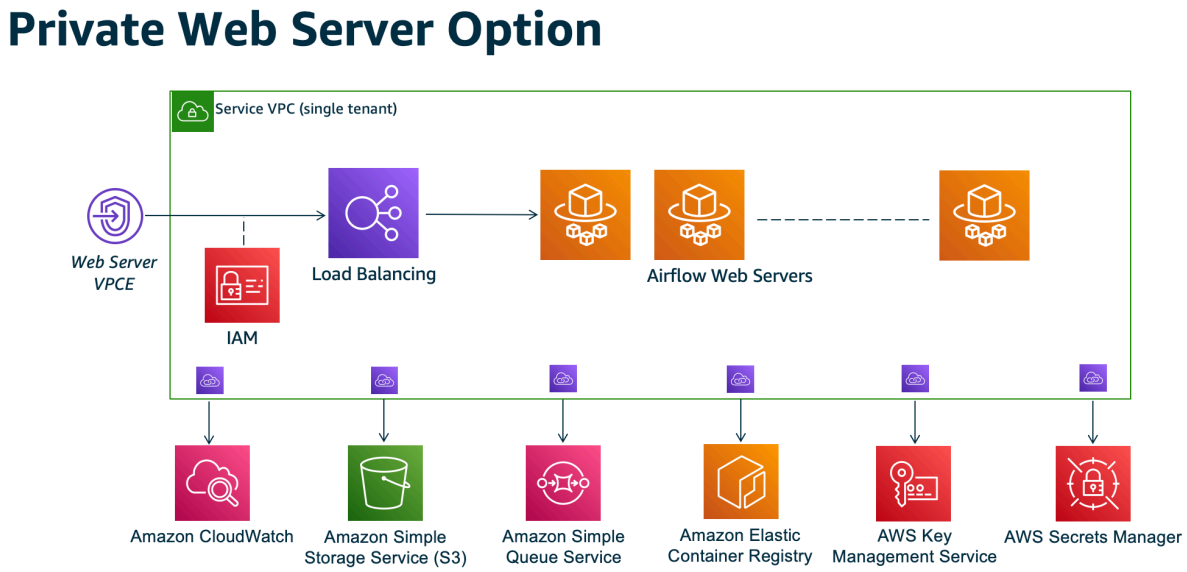
Questo tutorial illustra i passaggi per creare un tunnel SSH dal computer al server Web Apache Airflow per l'ambiente Amazon Managed Workflows for Apache Airflow. Si presuppone che tu abbia già creato un ambiente Amazon MWAA. Una volta configurato, un Linux Bastion Host funge da jump server che consente una connessione sicura dal computer alle risorse del VPC. Utilizzerai quindi un componente aggiuntivo di gestione proxy SOCKS per controllare le impostazioni del proxy nel browser per accedere all'interfaccia utente di Apache Airflow.

Sections

- Rete privata
- Casi d'uso
- Prima di iniziare
- Obiettivi
- Fase uno: creare l'istanza bastion
- Fase due: creare il tunnel ssh
- Fase tre: configura il bastion security group come regola in entrata
- Fase quattro: Copia l'URL di Apache Airflow
- Fase cinque: configurare le impostazioni del proxy
- Fase sei: apri l'interfaccia utente di Apache Airflow
- Fasi successive

Rete privata

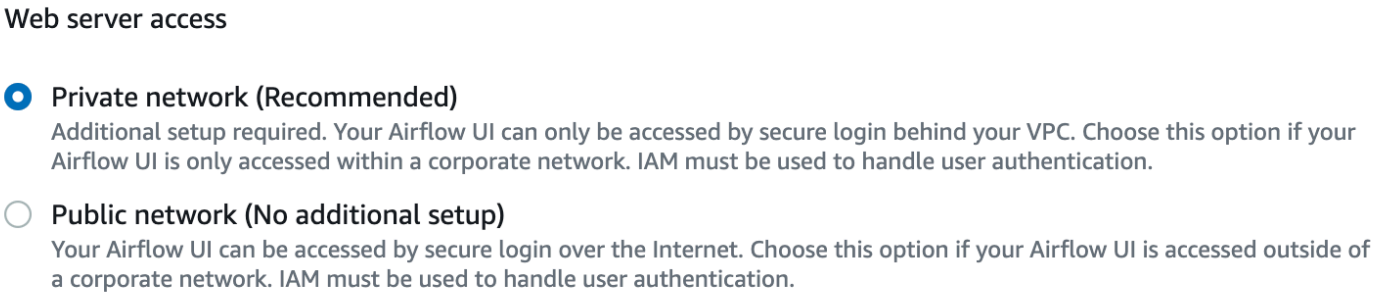
Questo tutorial presuppone che tu abbia scelto la modalità di accesso alla rete privata per il tuo server Web Apache Airflow.



La modalità di accesso alla rete privata limita l'accesso all'interfaccia utente di Apache Airflow agli utenti del tuo Amazon VPC a cui è stato concesso l'accesso alla policy IAM per il tuo ambiente.

Quando crei un ambiente con accesso privato al server web, devi impacchettare tutte le tue dipendenze in un archivio Python wheel `.whl` (), quindi fare riferimento a `.whl` nel tuo. `requirements.txt` Per istruzioni su come impacchettare e installare le dipendenze usando wheel, vedi Gestire le dipendenze usando Python wheel.

L'immagine seguente mostra dove trovare l'opzione Rete privata sulla console Amazon MWAA.



Casi d'uso

Puoi utilizzare questo tutorial dopo aver creato un ambiente Amazon MWAA. È necessario utilizzare lo stesso Amazon VPC, gli stessi gruppi di sicurezza VPC e le stesse sottoreti pubbliche del tuo ambiente.

Prima di iniziare

1. Verifica le autorizzazioni degli utenti. Assicurati che il tuo account in AWS Identity and Access Management (IAM) disponga di autorizzazioni sufficienti per creare e gestire risorse VPC.
2. Usa il tuo Amazon MWAA VPC. Questo tutorial presuppone che tu stia associando l'host bastion a un VPC esistente. Amazon VPC deve trovarsi nella stessa regione dell'ambiente Amazon MWAA e disporre di due sottoreti private, come definito in. Crea la rete VPC
3. Crea una chiave SSH. È necessario creare una chiave SSH Amazon EC2 (.pem) nella stessa regione dell'ambiente Amazon MWAA per connettersi ai server virtuali. Se non disponi di una chiave SSH, consulta Creare o importare una coppia di chiavi nella Guida per l'utente di Amazon EC2.

Obiettivi

In questo tutorial, verranno eseguite le operazioni seguenti:

1. Crea un'istanza Linux Bastion Host utilizzando un AWS CloudFormation modello per un VPC esistente.
2. Autorizza il traffico in entrata verso il gruppo di sicurezza dell'istanza bastion utilizzando una regola di ingresso sulla porta. 22
3. Autorizza il traffico in entrata dal gruppo di sicurezza di un ambiente Amazon MWAA al gruppo di sicurezza dell'istanza bastion.
4. Crea un tunnel SSH verso l'istanza bastion.
5. Installa e configura il FoxyProxy componente aggiuntivo per il browser Firefox per visualizzare l'interfaccia utente di Apache Airflow.

Fase uno: creare l'istanza bastion

La sezione seguente descrive i passaggi per creare l'istanza linux bastion utilizzando un AWS CloudFormation modello per un VPC esistente sulla AWS CloudFormation console.

Per creare Linux Bastion Host

1. Apri la pagina Deploy Quick Start sulla AWS CloudFormation console.
2. Usa il selettore di regione nella barra di navigazione per scegliere la stessa AWS regione del tuo ambiente Amazon MWAA.
3. Seleziona Successivo.
4. Digita un nome nel campo di testo del nome dello stack, ad esempio. `mwaa-linux-bastion`
5. Nel riquadro Parametri, Configurazione di rete, scegli le seguenti opzioni:
   a. Scegli l'ID VPC del tuo ambiente Amazon MWAA.
   b. Scegli l'ID Public Subnet 1 del tuo ambiente Amazon MWAA.
   c. Scegli l'ID Public Subnet 2 del tuo ambiente Amazon MWAA.
   d. Inserisci l'intervallo di indirizzi più ristretto possibile (ad esempio, un intervallo CIDR interno) in Allowed bastion external access CIDR.

   > **Note**
   >
   > Il modo più semplice per identificare un intervallo consiste nell'utilizzare lo stesso intervallo CIDR delle sottoreti pubbliche. Ad esempio, le sottoreti pubbliche nel AWS CloudFormation modello sulla pagina sono e. Crea la rete VPC `10.192.10.0/24` `10.192.11.0/24`

6. Nel riquadro di configurazione di Amazon EC2, scegli quanto segue:
   a. Scegli la tua chiave SSH nell'elenco a discesa in Nome della coppia di chiavi.
   b. Inserisci un nome in Bastion Host Name.
   c. Scegli true per l'inoltro TCP.

   > **Warning**
   >
   > L'inoltro TCP deve essere impostato su true in questo passaggio. Altrimenti, non sarai in grado di creare un tunnel SSH nel passaggio successivo.

7. Scegli Avanti, Avanti.
8. Seleziona la conferma, quindi scegli Crea pila.

Per saperne di più sull'architettura del tuo Linux Bastion Host, consulta Linux Bastion Hosts on the Cloud: Architecture. AWS

Fase due: creare il tunnel ssh

I passaggi seguenti descrivono come creare il tunnel ssh per il tuo bastione Linux. Un tunnel SSH riceve la richiesta dal tuo indirizzo IP locale al bastione linux, motivo per cui l'inoltro TCP per il bastione Linux era impostato come nei passaggi precedenti. `true`

macOS/Linux

Per creare un tunnel tramite riga di comando

1. Apri la pagina Istanze sulla console Amazon EC2.
2. Scegli un'istanza.
3. Copia l'indirizzo nel DNS IPv4 pubblico. Ad esempio, `ec2-4-82-142-1.compute-1.amazonaws.com`.
4. Nel prompt dei comandi, accedi alla directory in cui è archiviata la tua chiave SSH.
5. Esegui il comando seguente per connetterti all'istanza bastion usando ssh. Sostituisci il valore di esempio con il nome della tua chiave SSH in. `mykeypair.pem`

```
ssh -i mykeypair.pem -N -D 8157 ec2-user@YOUR_PUBLIC_IPV4_DNS
```

Windows (PuTTY)

Per creare un tunnel usando PuTTY

1. Apri la pagina Istanze sulla console Amazon EC2.
2. Scegli un'istanza.
3. Copia l'indirizzo nel DNS IPv4 pubblico. Ad esempio, `ec2-4-82-142-1.compute-1.amazonaws.com`.
4. Apri PuTTY, seleziona Sessione.
5. Inserisci il nome host in Nome host come ec2-user@ ***YOUR_PUBLIC_IPV4_DNS*** e la porta come. 22
6. Espandi la scheda SSH, seleziona Auth. Nel file con chiave privata per l'autenticazione, scegli il tuo file «ppk» locale.

7. In SSH, scegli la scheda Tunnel, quindi seleziona le opzioni Dynamic e Auto.
8. In Porta di origine, aggiungi la `8157` porta (o qualsiasi altra porta inutilizzata), quindi lascia vuota la porta di destinazione. Scegli Aggiungi.
9. Scegli la scheda Sessione e inserisci un nome per la sessione. Ad esempio `SSH Tunnel`.
10. Scegli Salva, Apri.

> **Note**
>
> Potrebbe essere necessario inserire una passphrase per la chiave pubblica.

> **Note**
>
> Se ricevi un `Permission denied (publickey)` errore, ti consigliamo di utilizzare lo strumento AWSSupport-TroubleshootSSH e di scegliere Esegui questa automazione (console) per risolvere i problemi di configurazione SSH.

Fase tre: configura il bastion security group come regola in entrata

L'accesso ai server e l'accesso regolare a Internet dai server sono consentiti con uno speciale gruppo di sicurezza di manutenzione collegato a tali server. I passaggi seguenti descrivono come configurare il bastion security group come fonte di traffico in entrata verso il gruppo di sicurezza VPC di un ambiente.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Nel riquadro Rete, scegli il gruppo di sicurezza VPC.
4. Sceglere Edit inbound rules (Modifica regole in entrata).
5. Scegli Aggiungi regola.
6. Scegli l'ID del tuo gruppo di sicurezza VPC nell'elenco a discesa Source.
7. Lascia vuote le opzioni rimanenti o imposta i valori predefiniti.
8. Scegliere Salva regole.

Fase quattro: Copia l'URL di Apache Airflow

I passaggi seguenti descrivono come aprire la console Amazon MWAA e copiare l'URL nell'interfaccia utente di Apache Airflow.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Copia l'URL nell'interfaccia utente Airflow per i passaggi successivi.

Fase cinque: configurare le impostazioni del proxy

Se si utilizza un tunnel SSH con inoltro dinamico delle porte, è necessario utilizzare un add-on per la gestione dei proxy SOCKS per controllare le impostazioni proxy nel browser. Ad esempio, puoi utilizzare la `--proxy-server` funzionalità di Chromium per avviare una sessione del browser o utilizzare l' FoxyProxy estensione nel browser Mozilla. FireFox

Opzione uno: configura un tunnel SSH utilizzando il port forwarding locale

Se non desideri utilizzare un proxy SOCKS, puoi configurare un tunnel SSH utilizzando il port forwarding locale. Il seguente comando di esempio accede all'interfaccia Web di Amazon ResourceManagerEC2 inoltrando il traffico sulla porta locale 8157.

1. Aprire il prompt dei comandi in una nuova finestra.
2. Digita il comando seguente per aprire un tunnel SSH.

```
ssh -i mykeypair.pem -N -L 8157:YOUR_VPC_ENDPOINT_ID-
vpce.YOUR_REGION.airflow.amazonaws.com:443
 ubuntu@YOUR_PUBLIC_IPV4_DNS.YOUR_REGION.compute.amazonaws.com
```

   `-L`Indica l'uso del port forwarding locale che consente di specificare una porta locale utilizzata per inoltrare i dati alla porta remota identificata sul server web locale del nodo.

3. Digita `http://localhost:8157/` nel tuo browser.

> **Note**
>
> Potrebbe essere necessario utilizzare`https://localhost:8157/`.

Opzione due: proxy tramite riga di comando

La maggior parte dei browser Web consente di configurare i proxy tramite una riga di comando o un parametro di configurazione. Ad esempio, con Chromium puoi avviare il browser con il seguente comando:

```
chromium --proxy-server="socks5://localhost:8157"
```

Questo avvia una sessione del browser che utilizza il tunnel ssh creato nei passaggi precedenti per inoltrare le sue richieste. Puoi aprire l'URL dell'ambiente Amazon MWAA privato (con https://) nel modo seguente:

```
https://YOUR_VPC_ENDPOINT_ID-vpce.YOUR_REGION.airflow.amazonaws.com/home.
```

Opzione tre: utilizzo di proxy per Mozilla Firefox FoxyProxy

L'esempio seguente mostra una configurazione FoxyProxy Standard (versione 7.5.1) per Mozilla Firefox. FoxyProxy fornisce una serie di strumenti di gestione dei proxy. Consente di utilizzare un server proxy per gli URL che corrispondono ai modelli corrispondenti ai domini utilizzati dall'interfaccia utente di Apache Airflow.

1. In Firefox, apri la pagina dell'estensione Standard. FoxyProxy
2. Scegli Aggiungi a Firefox.
3. Scegli Aggiungi.
4. Scegli l' FoxyProxy icona nella barra degli strumenti del browser, scegli Opzioni.
5. Copia il codice seguente e salvalo localmente con nome. `mwaa-proxy.json` Sostituisci il valore di esempio in ***YOUR_HOST_NAME*** con l'URL di Apache Airflow.

```
{
  "e0b7kh1606694837384": {
    "type": 3,
    "color": "#66cc66",
    "title": "airflow",
    "active": true,
    "address": "localhost",
    "port": 8157,
    "proxyDNS": false,
    "username": "",
```

```
    "password": "",
    "whitePatterns": [
      {
        "title": "airflow-ui",
        "pattern": "YOUR_HOST_NAME",
        "type": 1,
        "protocols": 1,
        "active": true
      }
    ],
    "blackPatterns": [],
    "pacURL": "",
    "index": -1
  },
  "k20d21508277536715": {
    "active": true,
    "title": "Default",
    "notes": "These are the settings that are used when no patterns match a URL.",
    "color": "#0055E5",
    "type": 5,
    "whitePatterns": [
      {
        "title": "all URLs",
        "active": true,
        "pattern": "*",
        "type": 1,
        "protocols": 1
      }
    ],
    "blackPatterns": [],
    "index": 9007199254740991
  },
  "logging": {
    "active": true,
    "maxSize": 500
  },
  "mode": "patterns",
  "browserVersion": "82.0.3",
  "foxyProxyVersion": "7.5.1",
  "foxyProxyEdition": "standard"
}
```

6. Nel riquadro Importa impostazioni da FoxyProxy 6.0+, scegli Importa impostazioni e seleziona il file. `mwaa-proxy.json`

7. Scegli OK.

Fase sei: apri l'interfaccia utente di Apache Airflow

I passaggi seguenti descrivono come aprire l'interfaccia utente di Apache Airflow.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli Open Airflow UI.

Fasi successive

- Scopri come eseguire i comandi CLI Airflow su un tunnel SSH verso un host bastion in. Riferimento ai comandi CLI di Apache Airflow
- Scopri come caricare il codice DAG nel tuo bucket Amazon S3 in. Aggiunta o aggiornamento di DAG

Tutorial: limitazione dell'accesso di un utente Amazon MWAA a un sottoinsieme di DAG

Amazon MWAA gestisce l'accesso al tuo ambiente mappando i tuoi principali IAM su uno o più ruoli predefiniti di Apache Airflow. Il seguente tutorial mostra come limitare i singoli utenti di Amazon MWAA a visualizzare e interagire solo con un DAG specifico o un set di DAG.

> **Note**
>
> I passaggi di questo tutorial possono essere completati utilizzando l'accesso federato, purché si possano assumere i ruoli IAM.

Argomenti

- Prerequisiti
- Fase uno: fornisci l'accesso al server Web Amazon MWAA al tuo principale IAM con il ruolo Public Apache Airflow predefinito.
- Fase due: creare un nuovo ruolo personalizzato di Apache Airflow
- Fase tre: assegna il ruolo che hai creato al tuo utente Amazon MWAA

- Passaggi successivi
- Risorse correlate

Prerequisiti

Per completare i passaggi di questo tutorial, avrai bisogno di quanto segue:

- Un ambiente Amazon MWAA con più DAG
- Un principale IAM, `Admin` con AdministratorAccessautorizzazioni, e un utente IAM, `MWAAUser` come principale per il quale è possibile limitare l'accesso al DAG. Per ulteriori informazioni sui ruoli di amministratore, consulta la funzione di amministratore nella Guida per l'utente IAM

> **Note**
>
> Non allegare politiche di autorizzazione direttamente ai tuoi utenti IAM. Ti consigliamo di configurare ruoli IAM che gli utenti possono assumere per ottenere l'accesso temporaneo alle tue risorse Amazon MWAA.

- AWS Command Line Interface versione 2 installata.

Fase uno: fornisci l'accesso al server Web Amazon MWAA al tuo principale IAM con il ruolo **`Public`** Apache Airflow predefinito.

Per concedere l'autorizzazione utilizzando il AWS Management Console

1. Accedi al tuo AWS account con un `Admin` ruolo e apri la console IAM.
2. Nel riquadro di navigazione a sinistra, scegli Utenti, quindi scegli il tuo utente Amazon MWAA IAM dalla tabella degli utenti.
3. Nella pagina dei dettagli dell'utente, in Riepilogo, scegli la scheda Autorizzazioni, quindi scegli Politiche di autorizzazione per espandere la scheda e scegli Aggiungi autorizzazioni.
4. Nella sezione Concedi autorizzazioni, scegli Allega direttamente le politiche esistenti, quindi scegli Crea politica per creare e allegare la tua politica di autorizzazioni personalizzata.
5. Nella pagina Crea politica, scegli JSON, quindi copia e incolla la seguente politica di autorizzazioni JSON nell'editor delle politiche. Questa politica concede l'accesso al server Web all'utente con il ruolo Apache Airflow predefinito`Public`.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
        "Effect": "Allow",
        "Action": "airflow:CreateWebLoginToken",
        "Resource": [

 "arn:aws:airflow:YOUR_REGION:YOUR_ACCOUNT_ID:role/YOUR_ENVIRONMENT_NAME/Public"
            ]
        }
    ]
 }
```

Fase due: creare un nuovo ruolo personalizzato di Apache Airflow

Per creare un nuovo ruolo utilizzando l'interfaccia utente di Apache Airflow

1. Utilizzando il tuo ruolo di amministratore IAM, apri la console Amazon MWAA e avvia l'interfaccia utente Apache Airflow del tuo ambiente.
2. Dal pannello di navigazione in alto, passa il mouse su Sicurezza per aprire l'elenco a discesa, quindi scegli Elenca ruoli per visualizzare i ruoli predefiniti di Apache Airflow.
3. Dall'elenco dei ruoli, seleziona Utente, quindi nella parte superiore della pagina scegli Azioni per aprire il menu a discesa. Scegli Copia ruolo e conferma Ok

   > **Note**
   >
   > Copia i ruoli Ops o Viewer per concedere rispettivamente un accesso maggiore o minore.

4. Individua il nuovo ruolo che hai creato nella tabella e scegli Modifica record.
5. Nella pagina Modifica ruolo, procedi come segue:

   - Per Nome, digita un nuovo nome per il ruolo nel campo di testo. Ad esempio, **`Restricted`**.
   - Per l'elenco delle autorizzazioni, rimuovi `can read on DAGs` e `can edit on DAGs` quindi aggiungi le autorizzazioni di lettura e scrittura per il set di DAG a cui desideri fornire l'accesso. Ad esempio, per un DAG, aggiungi e. `example_dag.py` **`can read on DAG:`*`example_dag`* `can edit on DAG:`*`example_dag`**

Selezionare Salva. Ora dovresti avere un nuovo ruolo che limiti l'accesso a un sottoinsieme di DAG disponibili nel tuo ambiente Amazon MWAA. Ora puoi assegnare questo ruolo a qualsiasi utente Apache Airflow esistente.

Fase tre: assegna il ruolo che hai creato al tuo utente Amazon MWAA

Per assegnare il nuovo ruolo

1. Utilizzando le credenziali di accesso per`MWAAUser`, esegui il seguente comando CLI per recuperare l'URL del server Web del tuo ambiente.

   ```
   $ aws mwaa get-environment --name YOUR_ENVIRONMENT_NAME | jq
    '.Environment.WebserverUrl'
   ```

   In caso di successo, verrà visualizzato il seguente risultato:

   ```
   "ab1b2345-678a-90a1-a2aa-34a567a8a901.c13.us-west-2.airflow.amazonaws.com"
   ```

2. Dopo `MWAAUser` aver effettuato l'accesso AWS Management Console, apri una nuova finestra del browser e accedi al seguente URL. `Webserver-URL`Sostituiscilo con i tuoi dati.

   ```
   https://<Webserver-URL>/home
   ```

   In caso di successo, verrà visualizzata una pagina di `Forbidden` errore perché non `MWAAUser` è stata ancora concessa l'autorizzazione per accedere all'interfaccia utente di Apache Airflow.

3. Dopo `Admin` aver effettuato l'accesso AWS Management Console, apri nuovamente la console Amazon MWAA e avvia l'interfaccia utente Apache Airflow del tuo ambiente.
4. Dalla dashboard dell'interfaccia utente, espandi il menu a discesa Sicurezza e questa volta scegli Elenca utenti.
5. Nella tabella degli utenti, trova il nuovo utente Apache Airflow e scegli Modifica record. Il nome dell'utente corrisponderà al tuo nome utente IAM nel seguente schema:. `user/mwaa-user`
6. Nella pagina Modifica utente, nella sezione Ruolo, aggiungi il nuovo ruolo personalizzato che hai creato, quindi scegli Salva.

> **Note**
>
> Il campo Cognome è obbligatorio, ma uno spazio soddisfa il requisito.

Il `Public` responsabile IAM concede l'`MWAAUser`autorizzazione per accedere all'interfaccia utente di Apache Airflow, mentre il nuovo ruolo fornisce le autorizzazioni aggiuntive necessarie per visualizzare i propri DAG.

> **Important**
>
> Tutti i 5 ruoli predefiniti (ad esempio`Admin`) non autorizzati da IAM che vengono aggiunti utilizzando l'interfaccia utente di Apache Airflow verranno rimossi al successivo accesso dell'utente.

Passaggi successivi

- Per ulteriori informazioni sulla gestione dell'accesso al tuo ambiente Amazon MWAA e per visualizzare esempi di policy JSON IAM che puoi utilizzare per gli utenti del tuo ambiente, consulta the section called “Accesso a un ambiente Amazon MWAA”

Risorse correlate

- Controllo degli accessi (documentazione di Apache Airflow): scopri di più sui ruoli predefiniti di Apache Airflow sul sito Web della documentazione di Apache Airflow.

Tutorial: automatizza la gestione degli endpoint del tuo ambiente su Amazon MWAA

Se gestisci più AWS account che condividono risorse, Amazon MWAA ti consente di creare e gestire i tuoi endpoint Amazon VPC. AWS Organizations Ciò significa che puoi utilizzare politiche di sicurezza più rigorose che consentono l'accesso solo alle risorse richieste dal tuo ambiente.

Quando crei un ambiente in un Amazon VPC condiviso, l'account proprietario dell'Amazon VPC principale (proprietario) condivide le due sottoreti private richieste da Amazon MWAA con altri account (partecipanti) che appartengono alla stessa organizzazione. Gli account dei partecipanti che condividono tali sottoreti possono quindi visualizzare, creare, modificare ed eliminare ambienti nel VPC condiviso.

Quando crei un ambiente in un Amazon VPC condiviso o altrimenti soggetto a restrizioni, Amazon MWAA crea prima le risorse VPC del servizio, quindi entra `PENDING`in uno stato per un massimo di 72 ore.

Quando lo stato dell'ambiente cambia da `CREATING` a`PENDING`, Amazon MWAA invia una EventBridge notifica Amazon del cambiamento di stato. Ciò consente all'account proprietario di creare gli endpoint richiesti per conto dei partecipanti in base alle informazioni sul servizio endpoint dalla console o dall'API Amazon MWAA o in modo programmatico. Di seguito, creiamo nuovi endpoint Amazon VPC utilizzando una funzione Lambda e una EventBridge regola che ascolta le notifiche di modifica dello stato di Amazon MWAA.

Qui creiamo i nuovi endpoint nello stesso Amazon VPC dell'ambiente. Per configurare un Amazon VPC condiviso, crea la EventBridge regola e la funzione Lambda nell'account del proprietario e l'ambiente Amazon MWAA nell'account del partecipante.

Argomenti

- Prerequisiti
- Crea Amazon VPC
- Creazione della funzione Lambda
- Crea la regola EventBridge
- Crea l'ambiente Amazon MWAA

Prerequisiti

Per completare i passaggi di questo tutorial, avrai bisogno di quanto segue:

- ...

Crea Amazon VPC

Utilizza il AWS CloudFormation modello e il AWS CLI comando seguenti per creare un nuovo Amazon VPC. Il modello configura le risorse Amazon VPC e modifica la policy degli endpoint per limitare l'accesso a una coda specifica.

1. Scarica il AWS CloudFormation modello, quindi decomprimi il file. `.yml`
2. In una nuova finestra del prompt dei comandi, accedi alla cartella in cui hai salvato il modello, quindi usala `create-stack`per creare lo stack. Il `--template-body` flag specifica il percorso del modello.

   ```
   $ aws cloudformation create-stack --stack-name stack-name --template-body file://
   cfn-vpc-private-network.yml
   ```

Nella prossima sezione, creerai la funzione Lambda.

Creazione della funzione Lambda

Usa il seguente codice Python e la policy IAM JSON per creare una nuova funzione Lambda e un nuovo ruolo di esecuzione. Questa funzione crea endpoint Amazon VPC per un server Web Apache Airflow privato e una coda Amazon SQS. Amazon MWAA utilizza Amazon SQS per mettere in coda le attività con Celery tra più lavoratori durante la scalabilità dell'ambiente.

1. Scarica il codice della funzione Python.
2. Scarica la politica di autorizzazione IAM, quindi decomprimi il file.
3. Apri un prompt dei comandi, quindi vai alla cartella in cui hai salvato la politica di autorizzazione JSON. Usa il `create-role`comando IAM per creare il nuovo ruolo.

   ```
   $ aws iam create-role --role-name function-role \
   --assume-role-policy-document file://lambda-mwaa-vpce-policy.json
   ```

   Nota il ruolo ARN della AWS CLI risposta. Nel passaggio successivo, specifichiamo questo nuovo ruolo come ruolo di esecuzione della funzione utilizzando il relativo ARN.
4. Vai alla cartella in cui hai salvato il codice della funzione, quindi usa il `create-function`comando per creare una nuova funzione.

   ```
   $ aws lambda create-function --function-name mwaa-vpce-lambda \
   ```

```
--zip-file file://mwaa-lambda-shared-vpc.zip --runtime python3.8 --role
 arn:aws:iam::123456789012:role/function-role --handler lambda_handler
```

Nota la funzione ARN dalla AWS CLI risposta. Nel passaggio successivo specifichiamo l'ARN per configurare la funzione come destinazione per una nuova EventBridge regola.

Nella sezione successiva, creerai la EventBridge regola che richiama questa funzione quando l'ambiente entra in uno stato. `PENDING`

Crea la regola EventBridge

Effettua le seguenti operazioni per creare una nuova regola che ascolti le notifiche di Amazon MWAA e utilizzi la tua nuova funzione Lambda.

1. Usa il EventBridge `put-rule` comando per creare una nuova regola. EventBridge

```
$ aws events put-rule --name "mwaa-lambda-rule" \
--event-pattern "{\"source\":[\"aws.airflow\"],\"detail-type\":[\"MWAA Environment
 Status Change\"]}"
```

Il pattern di eventi rileva le notifiche che Amazon MWAA invia ogni volta che lo stato di un ambiente cambia.

```
{
  "source": ["aws.airflow"],
  "detail-type": ["MWAA Environment Status Change"]
}
```

2. Utilizzate il `put-targets` comando per aggiungere la funzione Lambda come destinazione per la nuova regola.

```
$ aws events put-targets --rule "mwaa-lambda-rule" \
--targets "Id"="1","Arn"="arn:aws::lambda:region:123456789012:function:mwaa-vpce-
lambda"
```

Sei pronto per creare un nuovo ambiente Amazon MWAA con endpoint Amazon VPC gestiti dal cliente.

Crea l'ambiente Amazon MWAA

Usa la console Amazon MWAA per creare un nuovo ambiente con endpoint Amazon VPC gestiti dal cliente.

1. Apri la console Amazon MWAA e scegli Crea un ambiente.
2. Per Nome inserisci un nome univoco.
3. Per la versione Airflow scegli la versione più recente.
4. Scegli un bucket Amazon S3 e una cartella DAG, ad esempio da **`dags/`** utilizzare con l'ambiente, quindi scegli Avanti.
5. Nella pagina Configura impostazioni avanzate, procedi come segue:
   a. Per il Virtual Private Cloud, scegli l'Amazon VPC che hai creato nel passaggio precedente.
   b. Per l'accesso al server Web, scegli Rete pubblica (accessibile da Internet).
   c. Per i gruppi di sicurezza, scegli il gruppo di sicurezza con cui hai creatoAWS CloudFormation. Poiché i gruppi di sicurezza per gli AWS PrivateLink endpoint del passaggio precedente sono autoreferenziali, è necessario scegliere lo stesso gruppo di sicurezza per l'ambiente.
   d. Per la gestione degli endpoint, scegli Endpoint gestiti dal cliente.
6. Mantieni le impostazioni predefinite rimanenti, quindi scegli Avanti.
7. Controlla le tue selezioni, quindi scegli Crea ambiente.

> **Tip**
>
> Per ulteriori informazioni sulla configurazione di un nuovo ambiente, consulta Getting started with Amazon MWAA.

Quando l'ambiente lo è`PENDING`, Amazon MWAA invia una notifica che corrisponde allo schema di eventi impostato per la regola. La regola richiama la funzione Lambda. La funzione analizza l'evento di notifica e ottiene le informazioni necessarie sull'endpoint per il server Web e la coda Amazon SQS. Quindi crea gli endpoint nel tuo Amazon VPC.

Quando gli endpoint sono disponibili, Amazon MWAA riprende a creare il tuo ambiente. Quando è pronto, lo stato dell'ambiente cambia `AVAILABLE` e puoi accedere al server Web Apache Airflow utilizzando la console Amazon MWAA.

Esempi di codice per Amazon Managed Workflows for Apache Airflow

Questa guida contiene esempi di codice, inclusi DAG e plug-in personalizzati, che puoi utilizzare in un ambiente Amazon Managed Workflows for Apache Airflow. Per altri esempi di utilizzo di Apache Airflow con i AWS servizi, consulta la directory nel repository Apache Airflow. `dags` GitHub

Esempi

- Utilizzo di un DAG per importare variabili nella CLI
- Creazione di una connessione SSH utilizzando SSHOperator
- Utilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow Snowflake
- Utilizzo di un DAG per scrivere metriche personalizzate inCloudWatch
- Pulizia del database Aurora PostgreSQL in un ambiente Amazon MWAA
- Esportazione di metadati ambientali in file CSV su Amazon S3
- Utilizzo di una chiave segreta inAWS Secrets Managerper una variabile Apache Airflow
- Utilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow
- Creazione di un plugin personalizzato con Oracle
- Creazione di un plugin personalizzato che genera variabili di ambiente di runtime
- Modifica del fuso orario di un DAG su Amazon MWAA
- Rinfrescare un CodeArtifact gettone
- Creazione di un plugin personalizzato con Apache Hive e Hadoop
- Creazione di un plugin personalizzato per Apache AirflowPythonVirtualenvOperator
- Invocare DAG con una funzione Lambda
- Richiamo di DAG in diversi ambienti Amazon MWAA
- Utilizzo di Amazon MWAA con Amazon RDS per Microsoft SQL Server
- Utilizzo di Amazon MWAA con Amazon EMR
- Utilizzo di Amazon MWAA con Amazon EKS
- Connessione ad Amazon ECS tramiteECSOperator

- Usare dbt con Amazon MWAA
- AWS blog e tutorial

Utilizzo di un DAG per importare variabili nella CLI

Il seguente codice di esempio importa le variabili utilizzando l'interfaccia a riga di comando su Amazon Managed Workflows per Apache Airflow.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Dipendenze
- Esempio di codice
- Fasi successive

Versione

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

- Non sono necessarie autorizzazioni aggiuntive per utilizzare l'esempio di codice in questa pagina.

Autorizzazioni

Il tuoAWSl'account deve accedere al`AmazonMWAAAirflowCliAccess`politica. Per ulteriori informazioni, consulta Politica CLI di Apache Airflow: AmazonMWAA Access AirflowCli.

Dipendenze

- Per utilizzare questo esempio di codice con Apache Airflow v2, non sono necessarie dipendenze aggiuntive. Il codice utilizzaInstallazione base di Apache Airflow v2sul tuo ambiente.

Esempio di codice

Il seguente codice di esempio richiede tre input: il nome del tuo ambiente Amazon MWAA (in`mwaa_env`), ilAWSRegione del tuo ambiente (in`aws_region`) e il file locale che contiene le variabili da importare (in`var_file`).

```
import boto3
import json
import requests
import base64
import getopt
import sys

argv = sys.argv[1:]
mwaa_env=''
aws_region=''
var_file=''

try:
    opts, args = getopt.getopt(argv, 'e:v:r:', ['environment', 'variable-
file','region'])
    #if len(opts) == 0 and len(opts) > 3:
    if len(opts) != 3:
        print ('Usage: -e MWAA environment -v variable file location and filename -r
 aws region')
    else:
        for opt, arg in opts:
            if opt in ("-e"):
                mwaa_env=arg
            elif opt in ("-r"):
                aws_region=arg
            elif opt in ("-v"):
                var_file=arg

        boto3.setup_default_session(region_name="{}".format(aws_region))
        mwaa_env_name = "{}".format(mwaa_env)

        client = boto3.client('mwaa')
        mwaa_cli_token = client.create_cli_token(
            Name=mwaa_env_name
        )

        with open ("{}".format(var_file), "r") as myfile:
```

```
            fileconf = myfile.read().replace('\n', '')

        json_dictionary = json.loads(fileconf)
        for key in json_dictionary:
            print(key, " ", json_dictionary[key])
            val = (key + " " + json_dictionary[key])
            mwaa_auth_token = 'Bearer ' + mwaa_cli_token['CliToken']
            mwaa_webserver_hostname = 'https://{0}/aws_mwaa/
cli'.format(mwaa_cli_token['WebServerHostname'])
            raw_data = "variables set {0}".format(val)
            mwaa_response = requests.post(
                mwaa_webserver_hostname,
                headers={
                    'Authorization': mwaa_auth_token,
                    'Content-Type': 'text/plain'
                    },
                data=raw_data
                )
            mwaa_std_err_message = base64.b64decode(mwaa_response.json()
['stderr']).decode('utf8')
            mwaa_std_out_message = base64.b64decode(mwaa_response.json()
['stdout']).decode('utf8')
            print(mwaa_response.status_code)
            print(mwaa_std_err_message)
            print(mwaa_std_out_message)

except:
    print('Use this script with the following options: -e MWAA environment -v variable
 file location and filename -r aws region')
    print("Unexpected error:", sys.exc_info()[0])
    sys.exit(2)
```

Fasi successive

- Scopri come caricare il codice DAG in questo esempio sudagscartella nel tuo bucket Amazon S3Aggiunta o aggiornamento di DAG.

Creazione di una connessione SSH utilizzando **`SSHOperator`**

L'esempio seguente descrive come utilizzare il `SSHOperator` in un grafo aciclico diretto (DAG) per connetterti a un'istanza Amazon EC2 remota dal tuo ambiente Amazon Managed Workflows

for Apache Airflow. Puoi usare un approccio simile per connetterti a qualsiasi istanza remota con accesso SSH.

Nell'esempio seguente, carichi una chiave segreta SSH (`.pem`) `dags` nella directory del tuo ambiente su Amazon S3. Quindi, installi le dipendenze necessarie utilizzando `requirements.txt` e creando una nuova connessione Apache Airflow nell'interfaccia utente. Infine, scrivi un DAG che crea una connessione SSH all'istanza remota.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Requisiti
- Copia la tua chiave segreta su Amazon S3
- Crea una nuova connessione Apache Airflow
- Esempio di codice

Versione

- Puoi usare l'esempio di codice in questa pagina con Apache Airflow v2 e versioni successive in Python 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- Un ambiente Amazon MWAA.
- Una chiave segreta SSH. L'esempio di codice presuppone che tu abbia un'istanza Amazon EC2 e `.pem` una nella stessa regione del tuo ambiente Amazon MWAA. Se non disponi di una chiave, consulta Creare o importare una coppia di chiavi nella Guida per l'utente di Amazon EC2.

Autorizzazioni

- Non sono necessarie autorizzazioni aggiuntive per utilizzare l'esempio di codice in questa pagina.

Requisiti

Aggiungere il seguente parametro `requirements.txt` per installare il `apache-airflow-providers-ssh` pacchetto sul server Web. Una volta aggiornato l'ambiente e dopo che Amazon MWAA avrà installato correttamente la dipendenza, nell'interfaccia utente verrà visualizzato un nuovo tipo di connessione SSH.

```
-c https://raw.githubusercontent.com/apache/airflow/constraints-Airflow-version/
constraints-Python-version.txt
apache-airflow-providers-ssh
```

> **Note**
>
> `-c`definisce l'URL dei vincoli in. `requirements.txt` Ciò garantisce che Amazon MWAA installi la versione del pacchetto corretta per il tuo ambiente.

Copia la tua chiave segreta su Amazon S3

Usa il seguente AWS Command Line Interface comando per copiare la `.pem` chiave `dags` nella directory del tuo ambiente in Amazon S3.

```
$ aws s3 cp your-secret-key.pem s3://your-bucket/dags/
```

Amazon MWAA copia il contenuto`dags`, inclusa la `.pem` chiave, nella `/usr/local/airflow/dags/` directory locale. In questo modo, Apache Airflow può accedere alla chiave.

Crea una nuova connessione Apache Airflow

Per creare una nuova connessione SSH utilizzando l'interfaccia utente di Apache Airflow

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Dall'elenco degli ambienti, scegli Open Airflow UI per il tuo ambiente.
3. Nella pagina dell'interfaccia utente di Apache Airflow, scegli Amministratore dalla barra di navigazione in alto per espandere l'elenco a discesa, quindi scegli Connessioni.
4. Nella pagina Elenco connessioni, scegli + o il pulsante Aggiungi un nuovo record per aggiungere una nuova connessione.

5. Nella pagina Aggiungi connessione, aggiungi le seguenti informazioni:

   a. Per ID di connessione, immettere**ssh_new**.

   b. Per Tipo di connessione, scegli SSH dall'elenco a discesa.

   > **Note**
   >
   > Se il tipo di connessione SSH non è disponibile nell'elenco, Amazon MWAA non ha installato il pacchetto richiesto. `apache-airflow-providers-ssh` Aggiorna il `requirements.txt` file per includere questo pacchetto, quindi riprova.

   c. Per Host, inserisci l'indirizzo IP dell'istanza Amazon EC2 a cui desideri connetterti. Ad esempio, **12.345.67.89**.

   d. Per Nome utente, inserisci **ec2-user** se ti stai connettendo a un'istanza Amazon EC2. Il tuo nome utente potrebbe essere diverso, a seconda del tipo di istanza remota a cui desideri connettere Apache Airflow.

   e. Per Extra, inserisci la seguente coppia chiave-valore in formato JSON:

   ```
   { "key_file": "/usr/local/airflow/dags/your-secret-key.pem" }
   ```

   Questa coppia chiave-valore indica ad Apache Airflow di cercare la chiave segreta nella directory locale. `/dags`

Esempio di codice

Il seguente DAG utilizza il `SSHOperator` per connettersi all'istanza Amazon EC2 di destinazione, quindi esegue `hostname` il comando Linux per stampare il nome dell'istanza. È possibile modificare il DAG per eseguire qualsiasi comando o script sull'istanza remota.

1. Apri un terminale e vai alla directory in cui è memorizzato il codice DAG. Per esempio:

   ```
   cd dags
   ```

2. Copiate il contenuto del seguente esempio di codice e salvatelo localmente con nome. `ssh.py`

   ```
   from airflow.decorators import dag
   from datetime import datetime
   from airflow.providers.ssh.operators.ssh import SSHOperator
   ```

```
@dag(
    dag_id="ssh_operator_example",
    schedule_interval=None,
    start_date=datetime(2022, 1, 1),
    catchup=False,
    )
def ssh_dag():
    task_1=SSHOperator(
        task_id="ssh_task",
        ssh_conn_id='ssh_new',
        command='hostname',
    )

my_ssh_dag = ssh_dag()
```

3. Esegui il AWS CLI comando seguente per copiare il DAG nel bucket del tuo ambiente, quindi attiva il DAG utilizzando l'interfaccia utente di Apache Airflow.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. In caso di successo, nei log delle attività del DAG verrà visualizzato un risultato simile al seguente: `ssh_task` `ssh_operator_example`

```
[2022-01-01, 12:00:00 UTC] {{base.py:79}} INFO - Using connection to: id: ssh_new.
 Host: 12.345.67.89, Port: None,
Schema: , Login: ec2-user, Password: None, extra: {'key_file': '/usr/local/airflow/
dags/your-secret-key.pem'}
[2022-01-01, 12:00:00 UTC] {{ssh.py:264}} WARNING - Remote Identification Change is
 not verified. This won't protect against Man-In-The-Middle attacks
[2022-01-01, 12:00:00 UTC] {{ssh.py:270}} WARNING - No Host Key Verification. This
 won't protect against Man-In-The-Middle attacks
[2022-01-01, 12:00:00 UTC] {{transport.py:1819}} INFO - Connected (version 2.0,
 client OpenSSH_7.4)
[2022-01-01, 12:00:00 UTC] {{transport.py:1819}} INFO - Authentication (publickey)
 successful!
[2022-01-01, 12:00:00 UTC] {{ssh.py:139}} INFO - Running command: hostname
[2022-01-01, 12:00:00 UTC]{{ssh.py:171}} INFO - ip-123-45-67-89.us-
west-2.compute.internal
[2022-01-01, 12:00:00 UTC] {{taskinstance.py:1280}} INFO - Marking task as SUCCESS.
 dag_id=ssh_operator_example, task_id=ssh_task, execution_date=20220712T200914,
 start_date=20220712T200915, end_date=20220712T200916
```

Utilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow Snowflake

I seguenti esempi di chiamateAWS Secrets Managerper ottenere una chiave segreta per una connessione Apache Airflow Snowflake su Amazon Managed Workflows per Apache Airflow. Si presuppone che tu abbia completato i passaggi inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Requisiti
- Esempio di codice
- Fasi successive

Versione

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- Il backend di Secrets Manager come opzione di configurazione di Apache Airflow, come mostrato inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.
- Una stringa di connessione Apache Airflow in Secrets Manager come mostrata inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.

Autorizzazioni

- Autorizzazioni di Secrets Manager come mostrato inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.

Requisiti

Per utilizzare il codice di esempio in questa pagina, aggiungi le seguenti dipendenze al tuo`requirements.txt`. Per ulteriori informazioni, consulta Installazione delle dipendenze in Python.

```
apache-airflow-providers-snowflake==1.3.0
```

Esempio di codice

I passaggi seguenti descrivono come creare il codice DAG che richiama Secrets Manager per ottenere il segreto.

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il codice DAG. Ad esempio:

   ```
   cd dags
   ```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`snowflake_connection.py`.

   ```
   """
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

   Permission is hereby granted, free of charge, to any person obtaining a copy of
   this software and associated documentation files (the "Software"), to deal in
   the Software without restriction, including without limitation the rights to
   use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
   the Software, and to permit persons to whom the Software is furnished to do so.

   THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
   IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
   FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
   COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
   IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
   CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
   """
   ```

```
from airflow import DAG
from airflow.providers.snowflake.operators.snowflake import SnowflakeOperator
from airflow.utils.dates import days_ago

snowflake_query = [
    """use warehouse "MY_WAREHOUSE";""",
    """select * from "SNOWFLAKE_SAMPLE_DATA"."WEATHER"."WEATHER_14_TOTAL" limit
 100;""",
]

with DAG(dag_id='snowflake_test', schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
    snowflake_select = SnowflakeOperator(
        task_id="snowflake_select",
        sql=snowflake_query,
        snowflake_conn_id="snowflake_conn",
    )
```

Fasi successive

- Scopri come caricare il codice DAG in questo esempio su`dags`cartella nel tuo bucket Amazon S3 inAggiunta o aggiornamento di DAG.

Utilizzo di un DAG per scrivere metriche personalizzate inCloudWatch

È possibile utilizzare il seguente esempio di codice per scrivere un grafo aciclico diretto (DAG) che esegue un`PythonOperator`per recuperare le metriche a livello di sistema operativo per un ambiente Amazon MWAA. Il DAG pubblica quindi i dati come metriche personalizzate su AmazonCloudWatch.

Le metriche personalizzate a livello di sistema operativo forniscono una visibilità aggiuntiva su come i dipendenti dell'ambiente utilizzano risorse come la memoria virtuale e la CPU. È possibile utilizzare queste informazioni per selezionareclasse di ambientequella che meglio si adatta al tuo carico di lavoro.

Argomenti

- Versione

- Prerequisiti
- Autorizzazioni
- Dipendenze
- Esempio di codice

Versione

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare l'esempio di codice in questa pagina, è necessario quanto segue:

- UnAmbiente Amazon MWAA.

Autorizzazioni

- Non sono necessarie autorizzazioni aggiuntive per utilizzare l'esempio di codice in questa pagina.

Dipendenze

- Non sono necessarie dipendenze aggiuntive per utilizzare l'esempio di codice in questa pagina.

Esempio di codice

1. Nel prompt dei comandi, accedi alla cartella in cui è archiviato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`dag-custom-metrics.py`. Sostituisci`MWAA-ENV-NAME`con il nome del tuo ambiente.

```
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from airflow.utils.dates import days_ago
```

```
from datetime import datetime
import os,json,boto3,psutil,socket

def publish_metric(client,name,value,cat,unit='None'):
    environment_name = os.getenv("MWAA_ENV_NAME")
    value_number=float(value)
    hostname = socket.gethostname()
    ip_address = socket.gethostbyname(hostname)
    print('writing value',value_number,'to metric',name)
    response = client.put_metric_data(
        Namespace='MWAA-Custom',
        MetricData=[
            {
                'MetricName': name,
                'Dimensions': [
                    {
                        'Name': 'Environment',
                        'Value': environment_name
                    },
                    {
                        'Name': 'Category',
                        'Value': cat
                    },
                    {
                        'Name': 'Host',
                        'Value': ip_address
                    },
                ],
                'Timestamp': datetime.now(),
                'Value': value_number,
                'Unit': unit
            },
        ]
    )
    print(response)
    return response

def python_fn(**kwargs):
    client = boto3.client('cloudwatch')

    cpu_stats = psutil.cpu_stats()
    print('cpu_stats', cpu_stats)

    virtual = psutil.virtual_memory()
```

```
    cpu_times_percent = psutil.cpu_times_percent(interval=0)

    publish_metric(client=client, name='virtual_memory_total',
 cat='virtual_memory', value=virtual.total, unit='Bytes')
    publish_metric(client=client, name='virtual_memory_available',
 cat='virtual_memory', value=virtual.available, unit='Bytes')
    publish_metric(client=client, name='virtual_memory_used', cat='virtual_memory',
 value=virtual.used, unit='Bytes')
    publish_metric(client=client, name='virtual_memory_free', cat='virtual_memory',
 value=virtual.free, unit='Bytes')
    publish_metric(client=client, name='virtual_memory_active',
 cat='virtual_memory', value=virtual.active, unit='Bytes')
    publish_metric(client=client, name='virtual_memory_inactive',
 cat='virtual_memory', value=virtual.inactive, unit='Bytes')
    publish_metric(client=client, name='virtual_memory_percent',
 cat='virtual_memory', value=virtual.percent, unit='Percent')

    publish_metric(client=client, name='cpu_times_percent_user',
 cat='cpu_times_percent', value=cpu_times_percent.user, unit='Percent')
    publish_metric(client=client, name='cpu_times_percent_system',
 cat='cpu_times_percent', value=cpu_times_percent.system, unit='Percent')
    publish_metric(client=client, name='cpu_times_percent_idle',
 cat='cpu_times_percent', value=cpu_times_percent.idle, unit='Percent')

    return "OK"


with DAG(dag_id=os.path.basename(__file__).replace(".py", ""),
 schedule_interval='*/5 * * * *', catchup=False, start_date=days_ago(1)) as dag:
    t = PythonOperator(task_id="memory_test", python_callable=python_fn,
 provide_context=True)
```

3. Esegui quanto segueAWS CLIcomando per copiare il DAG nel bucket dell'ambiente, quindi attivare il DAG utilizzando l'interfaccia utente di Apache Airflow.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. Se il DAG viene eseguito correttamente, dovresti vedere qualcosa di simile a quanto segue nei log di Apache Airflow:

```
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO -
 cpu_stats scpustats(ctx_switches=3253992384, interrupts=1964237163,
 soft_interrupts=492328209, syscalls=0)
```

```
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - writing value
 16024199168.0 to metric virtual_memory_total
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - {'ResponseMetadata':
 {'RequestId': 'fad289ac-aa51-46a9-8b18-24e4e4063f4d', 'HTTPStatusCode': 200,
 'HTTPHeaders': {'x-amzn-requestid': 'fad289ac-aa51-46a9-8b18-24e4e4063f4d',
 'content-type': 'text/xml', 'content-length': '212', 'date': 'Tue, 16 Aug 2022
 17:54:45 GMT'}, 'RetryAttempts': 0}}
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - writing value
 14356287488.0 to metric virtual_memory_available
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - {'ResponseMetadata':
 {'RequestId': '6ef60085-07ab-4865-8abf-dc94f90cab46', 'HTTPStatusCode': 200,
 'HTTPHeaders': {'x-amzn-requestid': '6ef60085-07ab-4865-8abf-dc94f90cab46',
 'content-type': 'text/xml', 'content-length': '212', 'date': 'Tue, 16 Aug 2022
 17:54:45 GMT'}, 'RetryAttempts': 0}}
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - writing value
 1342296064.0 to metric virtual_memory_used
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - {'ResponseMetadata':
 {'RequestId': 'd5331438-5d3c-4df2-bc42-52dcf8d60a00', 'HTTPStatusCode': 200,
 'HTTPHeaders': {'x-amzn-requestid': 'd5331438-5d3c-4df2-bc42-52dcf8d60a00',
 'content-type': 'text/xml', 'content-length': '212', 'date': 'Tue, 16 Aug 2022
 17:54:45 GMT'}, 'RetryAttempts': 0}}
...
[2022-08-16, 10:54:46 UTC] {{python.py:152}} INFO - Done. Returned value was: OK
[2022-08-16, 10:54:46 UTC] {{taskinstance.py:1280}} INFO - Marking task as SUCCESS.
 dag_id=dag-custom-metrics, task_id=memory_test, execution_date=20220816T175444,
 start_date=20220816T175445, end_date=20220816T175446
[2022-08-16, 10:54:46 UTC] {{local_task_job.py:154}} INFO - Task exited with return
 code 0
```

Pulizia del database Aurora PostgreSQL in un ambiente Amazon MWAA

Amazon Managed Workflows for Apache Airflow utilizza un database Aurora PostgreSQL come database di metadati Apache Airflow, in cui vengono eseguite le operazioni DAG e vengono archiviate le istanze delle attività. Il seguente codice di esempio cancella periodicamente le voci dal database Aurora PostgreSQL dedicato per il tuo ambiente Amazon MWAA.

Argomenti

- Versione
- Prerequisiti

- Dipendenze
- Esempio di codice

Versione

- Puoi usare l'esempio di codice in questa pagina con Apache Airflow v2 e versioni successive in Python 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- Un ambiente Amazon MWAA.

Dipendenze

- Per utilizzare questo esempio di codice con Apache Airflow v2, non sono richieste dipendenze aggiuntive. Il codice utilizza l'installazione di base di Apache Airflow v2 nell'ambiente in uso.

Esempio di codice

Il seguente DAG pulisce il database dei metadati per le tabelle specificate in. `TABLES_TO_CLEAN` L'esempio elimina i dati dalle tabelle specificate degli ultimi sette giorni. Per modificare la data di eliminazione delle voci, `MAX_AGE_IN_DAYS` impostate un valore diverso.

Apache Airflow v2

```
from airflow import settings
from airflow.utils.dates import days_ago
from airflow.models import DagTag, DagModel, DagRun, ImportError, Log, SlaMiss,
 RenderedTaskInstanceFields, TaskInstance, TaskReschedule, XCom
from airflow.decorators import dag, task
from airflow.utils.dates import days_ago
from time import sleep

from airflow.version import version
major_version, minor_version = int(version.split('.')[0]), int(version.split('.')
[1])
```

```
if major_version >= 2 and minor_version >= 6:
    from airflow.jobs.job import Job
else:
    # The BaseJob class was renamed as of Apache Airflow v2.6
    from airflow.jobs.base_job import BaseJob as Job

# Delete entries for the past seven days. Adjust MAX_AGE_IN_DAYS to set how far back
 this DAG cleans the database.
MAX_AGE_IN_DAYS = 7
MIN_AGE_IN_DAYS = 0
DECREMENT = -7

# This is a list of (table, time) tuples.
# table = the table to clean in the metadata database
# time  = the column in the table associated to the timestamp of an entry
#         or None if not applicable.
TABLES_TO_CLEAN = [[Job, Job.latest_heartbeat],
    [TaskInstance, TaskInstance.execution_date],
    [TaskReschedule, TaskReschedule.execution_date],
    [DagTag, None],
    [DagModel, DagModel.last_parsed_time],
    [DagRun, DagRun.execution_date],
    [ImportError, ImportError.timestamp],
    [Log, Log.dttm],
    [SlaMiss, SlaMiss.execution_date],
    [RenderedTaskInstanceFields, RenderedTaskInstanceFields.execution_date],
    [XCom, XCom.execution_date],
]

@task()
def cleanup_db_fn(x):
    session = settings.Session()

    if x[1]:
        for oldest_days_ago in range(MAX_AGE_IN_DAYS, MIN_AGE_IN_DAYS, DECREMENT):
            earliest_days_ago = max(oldest_days_ago + DECREMENT, MIN_AGE_IN_DAYS)
            print(f"deleting {str(x[0])} entries between {earliest_days_ago} and
 {oldest_days_ago} days old...")
            earliest_date = days_ago(earliest_days_ago)
            oldest_date = days_ago(oldest_days_ago)
            query = session.query(x[0]).filter(x[1] >= oldest_date).filter(x[1] <=
 earliest_date)
            query.delete(synchronize_session= False)
            session.commit()
```

```
            sleep(5)
    else:
        # No time column specified for the table. Delete all entries
        print("deleting", str(x[0]), "...")
        query = session.query(x[0])
        query.delete(synchronize_session= False)
        session.commit()

    session.close()


@dag(
    dag_id="cleanup_db",
    schedule_interval="@weekly",
    start_date=days_ago(7),
    catchup=False,
    is_paused_upon_creation=False
)

def clean_db_dag_fn():
    t_last=None
    for x in TABLES_TO_CLEAN:
        t=cleanup_db_fn(x)
        if t_last:
            t_last >> t
        t_last = t

clean_db_dag = clean_db_dag_fn()
```

Esportazione di metadati ambientali in file CSV su Amazon S3

Il seguente esempio di codice mostra come creare un grafo aciclico diretto (DAG) che interroga il database per una serie di informazioni sull'esecuzione del DAG e scrive i dati in file archiviati `.csv` su Amazon S3.

Potresti voler esportare informazioni dal database Aurora PostgreSQL del tuo ambiente per ispezionare i dati localmente, archiviarli nello storage di oggetti o combinarli con strumenti come l'operatore Amazon S3 su Amazon Redshift e la pulizia del database, per spostare i metadati Amazon MWAA fuori dall'ambiente, ma conservarli per analisi future.

Puoi interrogare il database per qualsiasi oggetto elencato nei modelli Apache Airflow. Questo esempio di codice utilizza tre modelli,`DagRun`, e `TaskFailTaskInstance`, che forniscono informazioni pertinenti alle esecuzioni DAG.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Requisiti
- Esempio di codice

Versione

- Puoi usare l'esempio di codice in questa pagina con Apache Airflow v2 e versioni successive in Python 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- Un ambiente Amazon MWAA.
- Un nuovo bucket Amazon S3 in cui esportare le informazioni sui metadati.

Autorizzazioni

Amazon MWAA necessita dell'autorizzazione per l'`s3:PutObject`azione di Amazon S3 per scrivere le informazioni sui metadati richieste su Amazon S3. Aggiungi la seguente dichiarazione politica al ruolo di esecuzione del tuo ambiente.

```
{
  "Effect": "Allow",
  "Action": "s3:PutObject*",
  "Resource": "arn:aws:s3:::your-new-export-bucket"
}
```

Questa politica limita solo l'accesso in scrittura a *`your-new-export-bucket`*.

Requisiti

- Per utilizzare questo esempio di codice con Apache Airflow v2, non sono richieste dipendenze aggiuntive. Il codice utilizza l'installazione di base di Apache Airflow v2 nell'ambiente in uso.

Esempio di codice

I passaggi seguenti descrivono come creare un DAG che interroga Aurora PostgreSQL e scrive il risultato nel nuovo bucket Amazon S3.

1. Nel tuo terminale, accedi alla directory in cui è memorizzato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`metadata_to_csv.py`. È possibile modificare il valore assegnato per `MAX_AGE_IN_DAYS` controllare l'età dei record più vecchi interrogati dal DAG dal database dei metadati.

```
from airflow.decorators import dag, task
from airflow import settings
import os
import boto3
from airflow.utils.dates import days_ago
from airflow.models import DagRun, TaskFail, TaskInstance
import csv, re
from io import StringIO

DAG_ID = os.path.basename(__file__).replace(".py", "")

MAX_AGE_IN_DAYS = 30
S3_BUCKET = '<your-export-bucket>'
S3_KEY = 'files/export/{0}.csv'

# You can add other objects to export from the metadatabase,
OBJECTS_TO_EXPORT = [
    [DagRun,DagRun.execution_date],
    [TaskFail,TaskFail.execution_date],
    [TaskInstance, TaskInstance.execution_date],
]
```

```
@task()
def export_db_task(**kwargs):
    session = settings.Session()
    print("session: ",str(session))

    oldest_date = days_ago(MAX_AGE_IN_DAYS)
    print("oldest_date: ",oldest_date)

    s3 = boto3.client('s3')

    for x in OBJECTS_TO_EXPORT:
        query = session.query(x[0]).filter(x[1] >= days_ago(MAX_AGE_IN_DAYS))
        print("type",type(query))
        allrows=query.all()
        name=re.sub("[<>']", "", str(x[0]))
        print(name,": ",str(allrows))

        if len(allrows) > 0:
            outfileStr=""
            f = StringIO(outfileStr)
            w = csv.DictWriter(f, vars(allrows[0]).keys())
            w.writeheader()
            for y in allrows:
                w.writerow(vars(y))
            outkey = S3_KEY.format(name[6:])
            s3.put_object(Bucket=S3_BUCKET, Key=outkey, Body=f.getvalue())

@dag(
    dag_id=DAG_ID,
    schedule_interval=None,
    start_date=days_ago(1),
    )
def export_db():
    t = export_db_task()

metadb_to_s3_test = export_db()
```

3. Esegui il AWS CLI comando seguente per copiare il DAG nel bucket del tuo ambiente, quindi attiva il DAG utilizzando l'interfaccia utente di Apache Airflow.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. In caso di successo, nei log delle attività dell'operazione verrà generato un risultato simile al seguente: `export_db`

```
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - type <class
 'sqlalchemy.orm.query.Query'>
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - class
 airflow.models.dagrun.DagRun : [your-tasks]
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - type <class
 'sqlalchemy.orm.query.Query'>
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - class
 airflow.models.taskfail.TaskFail :  [your-tasks]
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - type <class
 'sqlalchemy.orm.query.Query'>
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - class
 airflow.models.taskinstance.TaskInstance :  [your-tasks]
[2022-01-01, 12:00:00 PDT] {{python.py:152}} INFO - Done. Returned value was: OK
[2022-01-01, 12:00:00 PDT] {{taskinstance.py:1280}} INFO - Marking task as
 SUCCESS. dag_id=metadb_to_s3, task_id=export_db, execution_date=20220101T000000,
 start_date=20220101T000000, end_date=20220101T000000
[2022-01-01, 12:00:00 PDT] {{local_task_job.py:154}} INFO - Task exited with return
 code 0
[2022-01-01, 12:00:00 PDT] {{local_task_job.py:264}} INFO - 0 downstream tasks
 scheduled from follow-on schedule check
```

Ora puoi accedere e scaricare i `.csv` file esportati nel tuo nuovo bucket Amazon S3. `/files/export/`

Utilizzo di una chiave segreta inAWS Secrets Managerper una variabile Apache Airflow

I seguenti esempi di chiamateAWS Secrets Managerper ottenere una chiave segreta per una variabile Apache Airflow su Amazon Managed Workflows per Apache Airflow. Si presuppone che tu abbia completato i passaggi inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni

- Requisiti
- Esempio di codice
- Fasi successive

Versione

- Il codice di esempio in questa pagina può essere utilizzato conApache Airflow v1nelPython 3.7.
- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- Il backend di Secrets Manager come opzione di configurazione di Apache Airflow, come mostrato inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.
- Una stringa variabile Apache Airflow in Secrets Manager, come mostrato inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.

Autorizzazioni

- Autorizzazioni di Secrets Manager come mostrato inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.

Requisiti

- Per utilizzare questo esempio di codice con Apache Airflow v1, non sono necessarie dipendenze aggiuntive. Il codice utilizzaInstallazione base di Apache Airflow v1sul tuo ambiente.
- Per utilizzare questo esempio di codice con Apache Airflow v2, non sono necessarie dipendenze aggiuntive. Il codice utilizzaInstallazione base di Apache Airflow v2sul tuo ambiente.

Esempio di codice

I passaggi seguenti descrivono come creare il codice DAG che richiama Secrets Manager per ottenere il segreto.

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`secrets-manager-var.py`.

```
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from airflow.models import Variable
from airflow.utils.dates import days_ago
from datetime import timedelta
import os
DAG_ID = os.path.basename(__file__).replace(".py", "")
DEFAULT_ARGS = {
    'owner': 'airflow',
    'depends_on_past': False,
    'email': ['airflow@example.com'],
    'email_on_failure': False,
    'email_on_retry': False,
}
def get_variable_fn(**kwargs):
    my_variable_name = Variable.get("test-variable", default_var="undefined")
    print("my_variable_name: ", my_variable_name)
    return my_variable_name
with DAG(
    dag_id=DAG_ID,
    default_args=DEFAULT_ARGS,
    dagrun_timeout=timedelta(hours=2),
    start_date=days_ago(1),
    schedule_interval='@once',
    tags=['variable']
) as dag:
    get_variable = PythonOperator(
        task_id="get_variable",
        python_callable=get_variable_fn,
        provide_context=True
```

```
)
```

Fasi successive

- Scopri come caricare il codice DAG in questo esempio `sudags`cartella nel tuo bucket Amazon S3Aggiunta o aggiornamento di DAG.

Utilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow

I seguenti esempi di chiamateAWS Secrets Managerper ottenere una chiave segreta per una connessione Apache Airflow su Amazon Managed Workflows per Apache Airflow. Si presuppone che tu abbia completato i passaggi inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Requisiti
- Esempio di codice
- Fasi successive

Versione

- Il codice di esempio in questa pagina può essere utilizzato conApache Airflow v1nelPython 3.7.

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- Il backend di Secrets Manager come opzione di configurazione di Apache Airflow, come mostrato inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.
- Una stringa di connessione Apache Airflow in Secrets Manager come mostrata inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.

Autorizzazioni

- Autorizzazioni di Secrets Manager come mostrato inConfigurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager.

Requisiti

- Per utilizzare questo esempio di codice con Apache Airflow v1, non sono necessarie dipendenze aggiuntive. Il codice utilizzaInstallazione base di Apache Airflow v1sul tuo ambiente.

- Per utilizzare questo esempio di codice con Apache Airflow v2, non sono necessarie dipendenze aggiuntive. Il codice utilizzaInstallazione base di Apache Airflow v2sul tuo ambiente.

Esempio di codice

I passaggi seguenti descrivono come creare il codice DAG che richiama Secrets Manager per ottenere il segreto.

Apache Airflow v2

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`secrets-manager.py`.

```
"""
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of
```

```
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""
from airflow import DAG, settings, secrets
from airflow.operators.python import PythonOperator
from airflow.utils.dates import days_ago
from airflow.providers.amazon.aws.hooks.base_aws import AwsBaseHook

from datetime import timedelta
import os

### The steps to create this secret key can be found at: https://
docs.aws.amazon.com/mwaa/latest/userguide/connections-secrets-manager.html
sm_secretId_name = 'airflow/connections/myconn'

default_args = {
    'owner': 'airflow',
    'start_date': days_ago(1),
    'depends_on_past': False
}


### Gets the secret myconn from Secrets Manager
def read_from_aws_sm_fn(**kwargs):
    ### set up Secrets Manager
    hook = AwsBaseHook(client_type='secretsmanager')
    client = hook.get_client_type('secretsmanager')
    response = client.get_secret_value(SecretId=sm_secretId_name)
    myConnSecretString = response["SecretString"]

    return myConnSecretString

### 'os.path.basename(__file__).replace(".py", "")' uses the file name secrets-
manager.py for a DAG ID of secrets-manager
with DAG(
```

```
        dag_id=os.path.basename(__file__).replace(".py", ""),
        default_args=default_args,
        dagrun_timeout=timedelta(hours=2),
        start_date=days_ago(1),
        schedule_interval=None
) as dag:
    write_all_to_aws_sm = PythonOperator(
        task_id="read_from_aws_sm",
        python_callable=read_from_aws_sm_fn,
        provide_context=True
    )
```

Apache Airflow v1

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`secrets-manager.py`.

```
from airflow import DAG, settings, secrets
from airflow.operators.python_operator import PythonOperator
from airflow.utils.dates import days_ago
from airflow.contrib.hooks.aws_hook import AwsHook

from datetime import timedelta
import os

### The steps to create this secret key can be found at: https://
docs.aws.amazon.com/mwaa/latest/userguide/connections-secrets-manager.html
sm_secretId_name = 'airflow/connections/myconn'

default_args = {
    'owner': 'airflow',
    'start_date': days_ago(1),
    'depends_on_past': False
}


### Gets the secret myconn from Secrets Manager
def read_from_aws_sm_fn(**kwargs):
```

```
    ### set up Secrets Manager
    hook = AwsHook()
    client = hook.get_client_type('secretsmanager')
    response = client.get_secret_value(SecretId=sm_secretId_name)
    myConnSecretString = response["SecretString"]

    return myConnSecretString

### 'os.path.basename(__file__).replace(".py", "")' uses the file name secrets-
manager.py for a DAG ID of secrets-manager
with DAG(
        dag_id=os.path.basename(__file__).replace(".py", ""),
        default_args=default_args,
        dagrun_timeout=timedelta(hours=2),
        start_date=days_ago(1),
        schedule_interval=None
) as dag:
    write_all_to_aws_sm = PythonOperator(
        task_id="read_from_aws_sm",
        python_callable=read_from_aws_sm_fn,
        provide_context=True
    )
```

Fasi successive

- Scopri come caricare il codice DAG in questo esempio sudagscartella nel tuo bucket Amazon S3 inAggiunta o aggiornamento di DAG.

Creazione di un plugin personalizzato con Oracle

L'esempio seguente illustra i passaggi per creare un plug-in personalizzato utilizzando Oracle per Amazon MWAA e può essere combinato con altri plugin e binari personalizzati nel file plugins.zip.

Indice

- Versione
- Prerequisiti
- Autorizzazioni
- Requisiti

- Esempio di codice
- Crea il plugin personalizzato
  - Scarica le dipendenze
  - Plugin personalizzato
  - Plugins.zip
- Opzioni di configurazione del flusso d'aria
- Fasi successive

Versione

- Il codice di esempio in questa pagina può essere utilizzato conApache Airflow v1nelPython 3.7.

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- UnAmbiente Amazon MWAA.
- Lavoratoreregistrazione abilitata a qualsiasi livello di registro,`CRITICAL`o superiore, per il tuo ambiente. Per ulteriori informazioni sui tipi di log di Amazon MWAA e su come gestire i gruppi di log, consultathe section called “Visualizzazione dei registri Airflow”

Autorizzazioni

- Non sono necessarie autorizzazioni aggiuntive per utilizzare l'esempio di codice in questa pagina.

Requisiti

Per utilizzare il codice di esempio in questa pagina, aggiungi le seguenti dipendenze al tuo`requirements.txt`. Per ulteriori informazioni, consulta Installazione delle dipendenze in Python.

Apache Airflow v2

```
-c https://raw.githubusercontent.com/apache/airflow/constraints-2.0.2/
constraints-3.7.txt
cx_Oracle
apache-airflow-providers-oracle
```

Apache Airflow v1

```
cx_Oracle==8.1.0
apache-airflow[oracle]==1.10.12
```

Esempio di codice

I passaggi seguenti descrivono come creare il codice DAG che testerà il plug-in personalizzato.

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`oracle.py`.

```
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from airflow.utils.dates import days_ago
import os
import cx_Oracle

DAG_ID = os.path.basename(__file__).replace(".py", "")

def testHook(**kwargs):
    cx_Oracle.init_oracle_client()
    version = cx_Oracle.clientversion()
    print("cx_Oracle.clientversion",version)
    return version

with DAG(dag_id=DAG_ID, schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
    hook_test = PythonOperator(
        task_id="hook_test",
        python_callable=testHook,
```

```
        provide_context=True
    )
```

Crea il plugin personalizzato

Questa sezione descrive come scaricare le dipendenze, creare il plugin personalizzato e il file plugins.zip.

Scarica le dipendenze

Amazon MWAA estrarrà il contenuto di plugins.zip in`/usr/local/airflow/plugins`su ogni pianificatore e container di lavoro Amazon MWAA. Viene utilizzato per aggiungere binari al tuo ambiente. I passaggi seguenti descrivono come assemblare i file necessari per il plug-in personalizzato.

Estrarre l'immagine del contenitore Amazon Linux

1. Nel prompt dei comandi, estrai l'immagine del contenitore Amazon Linux ed esegui il contenitore localmente. Ad esempio:

   ```
   docker pull amazonlinux
   docker run -it amazonlinux:latest /bin/bash
   ```

   Il prompt dei comandi dovrebbe richiamare una riga di comando bash. Ad esempio:

   ```
   bash-4.2#
   ```

2. Installa la struttura I/O asincrona nativa per Linux (libaio).

   ```
   yum -y install libaio
   ```

3. Tieni aperta questa finestra per i passaggi successivi. Copieremo i seguenti file localmente:`lib64/libaio.so.1`,`lib64/libaio.so.1.0.0`,`lib64/libaio.so.1.0.1`.

Scarica la cartella del cliente

1. Installa il pacchetto di decompressione localmente. Ad esempio:

```
sudo yum install unzip
```

2. Crea una directory `oracle_plugin`. Ad esempio:

```
mkdir oracle_plugin
cd oracle_plugin
```

3. Usa il seguente comando curl per scaricare il[instantclient-basic-linux.x64-18.5.0.0.0dbru.zip](#)da[Download di Oracle Instant Client per Linux x86-64 (64 bit)](#).

```
curl https://download.oracle.com/otn_software/linux/instantclient/185000/
instantclient-basic-linux.x64-18.5.0.0.0dbru.zip > client.zip
```

4. Decomprimi il file `client.zip`. Ad esempio:

```
unzip *.zip
```

Estrarre file da Docker

1. In un nuovo prompt dei comandi, visualizza e annota l'ID del contenitore Docker. Ad esempio:

```
docker container ls
```

Il prompt dei comandi dovrebbe restituire tutti i contenitori e i relativi ID. Ad esempio:

```
debc16fd6970
```

2. Nel tuo`oracle_plugin`cartella, estrai il`lib64/libaio.so.1`,`lib64/libaio.so.1.0.0`,`lib64/libaio.so.1.0.1`file in locale`instantclient_18_5`cartella. Ad esempio:

```
docker cp debc16fd6970:/lib64/libaio.so.1 instantclient_18_5/
docker cp debc16fd6970:/lib64/libaio.so.1.0.0 instantclient_18_5/
docker cp debc16fd6970:/lib64/libaio.so.1.0.1 instantclient_18_5/
```

Plugin personalizzato

Apache Airflow eseguirà il contenuto dei file Python nella cartella dei plugin all'avvio. Viene utilizzato per impostare e modificare le variabili di ambiente. I passaggi seguenti descrivono il codice di esempio per il plug-in personalizzato.

- Copia il contenuto del seguente esempio di codice e salvalo localmente come`env_var_plugin_oracle.py`.

```
from airflow.plugins_manager import AirflowPlugin
import os

os.environ["LD_LIBRARY_PATH"]='/usr/local/airflow/plugins/instantclient_18_5'
os.environ["DPI_DEBUG_LEVEL"]="64"

class EnvVarPlugin(AirflowPlugin):
    name = 'env_var_plugin'
```

Plugins.zip

I passaggi seguenti mostrano come crea re`plugins.zip`. I contenuti di questo esempio possono essere combinati con gli altri plugin e binari in un unico`plugins.zip`fascicolo.

Comprimi il contenuto della directory del plugin

1. Nel prompt dei comandi, accedi a`oracle_plugin`rubrica. Ad esempio:

```
cd oracle_plugin
```

2. Comprimi il`instantclient_18_5`cartella in plugins.zip. Ad esempio:

```
zip -r ../plugins.zip ./
```

3. Dovresti vedere quanto segue nel prompt dei comandi:

```
oracle_plugin$ ls
client.zip  instantclient_18_5
```

4. Rimuovi il`client.zip`fascicolo. Ad esempio:

```
rm client.zip
```

Comprimi il file env_var_plugin_oracle.py

1. Aggiungi laenv_var_plugin_oracle.pyfile nella cartella principale del file plugins.zip. Ad esempio:

```
zip plugins.zip env_var_plugin_oracle.py
```

2. Il tuo plugins.zip dovrebbe ora includere quanto segue:

```
env_var_plugin_oracle.py
instantclient_18_5/
```

Opzioni di configurazione del flusso d'aria

Se utilizzi Apache Airflow v2, aggiungicore.lazy_load_plugins : Falsecome opzione di configurazione di Apache Airflow. Per saperne di più, consultaUtilizzo delle opzioni di configurazione per caricare i plugin in 2.

Fasi successive

- Scopri come caricare ilrequirements.txtinserisci questo esempio nel tuo bucket Amazon S3Installazione delle dipendenze in Python.
- Scopri come caricare il codice DAG in questo esempio sudagscartella nel tuo bucket Amazon S3 inAggiunta o aggiornamento di DAG.
- Scopri di più su come caricare ilplugins.zipinserisci questo esempio nel tuo bucket Amazon S3Installazione di plugin personalizzati.

Creazione di un plugin personalizzato che genera variabili di ambiente di runtime

L'esempio seguente illustra i passaggi per creare un plug-in personalizzato che generi variabili di ambiente in fase di esecuzione in un ambiente Amazon Managed Workflows for Apache Airflow.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Requisiti
- Plug-in personalizzato
- Plugins.zip
- Opzioni di configurazione del flusso d'aria
- Fasi successive

Versione

- Il codice di esempio in questa pagina può essere utilizzato con Apache Airflow v1 in Python 3.7.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, è necessario quanto segue:

- Un ambiente Amazon MWAA.

Autorizzazioni

- Non sono necessarie autorizzazioni aggiuntive per utilizzare l'esempio di codice in questa pagina.

Requisiti

- Per utilizzare questo esempio di codice con Apache Airflow v1, non sono necessarie dipendenze aggiuntive. Il codice utilizza l'installazione di base di Apache Airflow v1 nell'ambiente in uso.

Plug-in personalizzato

Apache Airflow eseguirà il contenuto dei file Python nella cartella dei plugin all'avvio. Viene utilizzato per impostare e modificare le variabili di ambiente. Nei passaggi seguenti viene illustrato il codice di esempio per il plugin personalizzato.

1. Nella directory in cui sono archiviati i plugin. Ad esempio:

```
cd plugins
```

2. Copiare il contenuto del seguente esempio di codice e salvare localmente come`env_var_plugin.py` nella directory precedente.

```
from airflow.plugins_manager import AirflowPlugin
import os

os.environ["PATH"] = os.getenv("PATH") + ":/usr/local/airflow/.local/lib/python3.7/
site-packages"
os.environ["JAVA_HOME"]="/usr/lib/jvm/java-1.8.0-
openjdk-1.8.0.272.b10-1.amzn2.0.1.x86_64"

class EnvVarPlugin(AirflowPlugin):
     name = 'env_var_plugin'
```

Plugins.zip

Nei passaggi seguenti viene illustrato come crea`plugins.zip`. Il contenuto di questo esempio può essere combinato con altri plugin e binari in un unico`plugins.zip` file.

1. Nella directory del prompt dei comandi, passare alla`hive_plugin` directory del passaggio precedente. Ad esempio:

```
cd plugins
```

2. Comprimi i contenuti all'interno della`plugins` cartella.

```
zip -r ../plugins.zip ./
```

Opzioni di configurazione del flusso d'aria

Se utilizzi Apache Airflow v2, aggiungilo`core.lazy_load_plugins : False` come opzione di configurazione di Apache Airflow. Per ulteriori informazioni, consulta Utilizzo delle opzioni di configurazione per caricare i plugin in 2.

Fasi successive

- Scopri come caricare il`requirements.txt` file di questo esempio nel tuo bucket in Amazon S3Installazione delle dipendenze in Python.
- Scopri come caricare il codice DAG di questo esempio`dags` nella cartella del tuo bucket in Amazon S3Aggiunta o aggiornamento di DAG.
- Scopri di più su come caricare il`plugins.zip` file di questo esempio nel tuo bucket in Amazon S3Installazione di plugin personalizzati.

Modifica del fuso orario di un DAG su Amazon MWAA

Per impostazione predefinita, Apache Airflow pianifica il grafo aciclico diretto (DAG) in UTC+0. I passaggi seguenti mostrano come modificare il fuso orario con cui Amazon MWAA esegue i tuoi DAGPendolo. Facoltativamente, questo argomento dimostra come creare un plug-in personalizzato per modificare il fuso orario dei log di Apache Airflow del tuo ambiente.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Crea un plugin per modificare il fuso orario nei log di Airflow
- Creazione di una destinazione plugins.zip
- Esempio di codice
- Fasi successive

Versione

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- UnAmbiente Amazon MWAA.

Autorizzazioni

- Non sono necessarie autorizzazioni aggiuntive per utilizzare l'esempio di codice in questa pagina.

Crea un plugin per modificare il fuso orario nei log di Airflow

Apache Airflow eseguirà i file Python nel`plugins`elenco all'avvio. Con il seguente plugin, puoi sovrascrivere il fuso orario dell'esecutore, che modifica il fuso orario in cui Apache Airflow scrive i log.

1. Crea una cartella denominata`plugins`per il tuo plugin personalizzato e vai alla directory. Ad esempio:

```
$ mkdir plugins
$ cd plugins
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`dag-timezone-plugin.py`nel`plugins`cartella.

```
import time
import os

os.environ['TZ'] = 'America/Los_Angeles'
time.tzset()
```

3. Nel`plugins`directory, crea un file Python vuoto denominato`__init__.py`. Il tuo`plugins`la cartella dovrebbe essere simile alla seguente:

```
plugins/
|-- __init__.py
|-- dag-timezone-plugin.py
```

Creazione di una destinazione **`plugins.zip`**

I passaggi seguenti mostrano come crea`plugins.zip`. Il contenuto di questo esempio può essere combinato con altri plugin e binari in un unico`plugins.zip`fascicolo.

1. Nel prompt dei comandi, accedi a `plugins` cartella del passaggio precedente. Ad esempio:

```
cd plugins
```

2. Comprimi i contenuti all'interno del tuo `plugins` rubrica.

```
zip -r ../plugins.zip ./
```

3. Caricare `plugins.zip` al tuo bucket S3

```
$ aws s3 cp plugins.zip s3://your-mwaa-bucket/
```

Esempio di codice

Per modificare il fuso orario predefinito (UTC+0) in cui viene eseguito il DAG, useremo una libreria chiamata Pendolo, una libreria Python per lavorare con datetime compatibili con il fuso orario.

1. Nel prompt dei comandi, accedi alla directory in cui sono archiviati i tuoi DAG. Ad esempio:

```
$ cd dags
```

2. Copia il contenuto dell'esempio seguente e salvalo con nome `tz-aware-dag.py`.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from datetime import datetime, timedelta
# Import the Pendulum library.
import pendulum

# Instantiate Pendulum and set your timezone.
local_tz = pendulum.timezone("America/Los_Angeles")

with DAG(
    dag_id = "tz_test",
    schedule_interval="0 12 * * *",
    catchup=False,
    start_date=datetime(2022, 1, 1, tzinfo=local_tz)
) as dag:
    bash_operator_task = BashOperator(
        task_id="tz_aware_task",
        dag=dag,
```

```
        bash_command="date"
    )
```

3. Esegui quanto segueAWS CLIcomando per copiare il DAG nel bucket dell'ambiente, quindi attivare il DAG utilizzando l'interfaccia utente di Apache Airflow.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. In caso di successo, nei registri delle attività per il`tz_aware_task`nel`tz_test`GIORNO:

```
[2022-08-01, 12:00:00 PDT] {{subprocess.py:74}} INFO - Running command: ['bash', '-
c', 'date']
[2022-08-01, 12:00:00 PDT] {{subprocess.py:85}} INFO - Output:
[2022-08-01, 12:00:00 PDT] {{subprocess.py:89}} INFO - Mon Aug  1 12:00:00 PDT 2022
[2022-08-01, 12:00:00 PDT] {{subprocess.py:93}} INFO - Command exited with return
 code 0
[2022-08-01, 12:00:00 PDT] {{taskinstance.py:1280}} INFO - Marking task as
 SUCCESS. dag_id=tz_test, task_id=tz_aware_task, execution_date=20220801T190033,
 start_date=20220801T190035, end_date=20220801T190035
[2022-08-01, 12:00:00 PDT] {{local_task_job.py:154}} INFO - Task exited with return
 code 0
[2022-08-01, 12:00:00 PDT] {{local_task_job.py:264}} INFO - 0 downstream tasks
 scheduled from follow-on schedule check
```

Fasi successive

- Scopri di più su come caricare il`plugins.zip`inserisci questo esempio nel tuo bucket Amazon S3Installazione di plugin personalizzati.

Rinfrescare un CodeArtifact gettone

Se stai usando CodeArtifact per installare le dipendenze Python, Amazon MWAA richiede un token attivo. Per consentire ad Amazon MWAA di accedere a CodeArtifact repository in fase di esecuzione, puoi usare unscript di avvioe imposta il`PIP_EXTRA_INDEX_URL`con il token.

L'argomento seguente descrive come creare uno script di avvio che utilizza il`get_authorization_token` CodeArtifact Funzionamento dell'API per recuperare un nuovo token ogni volta che l'ambiente si avvia o si aggiorna.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Esempio di codice
- Fasi successive

Versione

- Puoi utilizzare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- UnAmbiente Amazon MWAA.
- UNCodeArtifact magazzinodove archiviate le dipendenze per il vostro ambiente.

Autorizzazioni

Per aggiornare il CodeArtifact token e scrittura del risultato su Amazon S3 Amazon MWAA deve disporre delle seguenti autorizzazioni nel ruolo di esecuzione.

- Il`codeartifact:GetAuthorizationToken`nazione consente ad Amazon MWAA di recuperare un nuovo token da CodeArtifact. La seguente politica concede l'autorizzazione per ogni CodeArtifact dominio che crei. Puoi limitare ulteriormente l'accesso ai tuoi domini modificando il valore della risorsa nell'istruzione e specificando solo i domini a cui desideri che l'ambiente acceda.

```
{
  "Effect": "Allow",
  "Action": "codeartifact:GetAuthorizationToken",
```

```
  "Resource": "arn:aws:codeartifact:us-west-2:*:domain/*"
}
```

- Il`sts:GetServiceBearerToken`è necessaria un'azione per chiamare il CodeArtifact [GetAuthorizationToken](#)funzionamento dell'API. Questa operazione restituisce un token che deve essere utilizzato quando si utilizza un gestore di pacchetti come`pip`con CodeArtifact. Per utilizzare un gestore di pacchetti con un CodeArtifact repository, il ruolo del ruolo di esecuzione dell'ambiente deve consentire`sts:GetServiceBearerToken`come illustrato nella seguente dichiarazione politica.

```
{
  "Sid": "AllowServiceBearerToken",
  "Effect": "Allow",
  "Action": "sts:GetServiceBearerToken",
  "Resource": "*"
}
```

Esempio di codice

I passaggi seguenti descrivono come creare uno script di avvio che aggiorni il CodeArtifact gettone.

1. Copia il contenuto del seguente esempio di codice e salvalo localmente come`code_artifact_startup_script.sh`.

```
#!/bin/sh

# Startup script for MWAA, see https://docs.aws.amazon.com/mwaa/latest/userguide/
using-startup-script.html

set -eu

# setup code artifact endpoint and token
# https://pip.pypa.io/en/stable/cli/pip_install/#cmdoption-0
# https://docs.aws.amazon.com/mwaa/latest/userguide/samples-code-artifact.html
DOMAIN="amazon"
DOMAIN_OWNER="112233445566"
REGION="us-west-2"
REPO_NAME="MyRepo"
echo "Getting token for CodeArtifact with args: --domain $DOMAIN --region $REGION
 --domain-owner $DOMAIN_OWNER"
```

```
TOKEN=$(aws codeartifact get-authorization-token --domain $DOMAIN --region $REGION
 --domain-owner $DOMAIN_OWNER | jq -r '.authorizationToken')
echo "Setting Pip env var for '--index-url' to point to CodeArtifact"
export PIP_EXTRA_INDEX_URL="https://aws:$TOKEN@$DOMAIN-
$DOMAIN_OWNER.d.codeartifact.$REGION.amazonaws.com/pypi/$REPO_NAME/simple/"
echo "CodeArtifact startup setup complete"
```

2. Vai alla cartella in cui hai salvato lo script. Usacpin una nuova finestra di richiesta per caricare lo script nel tuo bucket. Sostituisci*your-s3-bucket*con le tue informazioni.

```
$ aws s3 cp code_artifact_startup_script.sh s3://your-s3-bucket/
code_artifact_startup_script.sh
```

In caso di successo, Amazon S3 restituisce il percorso URL dell'oggetto:

```
upload: ./code_artifact_startup_script.sh to s3://your-s3-bucket/
code_artifact_startup_script.sh
```

Dopo aver caricato lo script, l'ambiente si aggiorna ed esegue lo script all'avvio.

Fasi successive

- Scopri come utilizzare gli script di avvio per personalizzare il tuo ambiente inthe section called “Utilizzo di uno script di avvio”.
- Scopri come caricare il codice DAG in questo esempio su`dags`cartella nel tuo bucket Amazon S3 inAggiunta o aggiornamento di DAG.
- Scopri di più su come caricare il`plugins.zip`archivia questo esempio nel tuo bucket Amazon S3 inInstallazione di plugin personalizzati.

Creazione di un plugin personalizzato con Apache Hive e Hadoop

Amazon MWAA estrae il contenuto di un`plugins.zip`a`/usr/local/airflow/plugins`. Questo può essere usato per aggiungere binari ai tuoi contenitori. Inoltre, Apache Airflow esegue il contenuto dei file Python in`plugins`cartella inavviare—che consente di impostare e modificare le variabili di ambiente. L'esempio seguente illustra i passaggi per creare un plug-in personalizzato utilizzando Apache Hive e Hadoop su un ambiente Amazon Managed Workflows per Apache Airflow e può essere combinato con altri plugin e binari personalizzati.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Requisiti
- Scarica le dipendenze
- Plugin personalizzato
- Plugins.zip
- Esempio di codice
- Opzioni di configurazione del flusso d'aria
- Fasi successive

Versione

- Il codice di esempio in questa pagina può essere utilizzato conApache Airflow v1nelPython 3.7.
- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- UnAmbiente Amazon MWAA.

Autorizzazioni

- Non sono necessarie autorizzazioni aggiuntive per utilizzare l'esempio di codice in questa pagina.

Requisiti

Per utilizzare il codice di esempio in questa pagina, aggiungi le seguenti dipendenze al tuo`requirements.txt`. Per ulteriori informazioni, consulta Installazione delle dipendenze in Python.

Apache Airflow v2

```
-c https://raw.githubusercontent.com/apache/airflow/constraints-2.0.2/
constraints-3.7.txt
apache-airflow-providers-amazon[apache.hive]
```

Apache Airflow v1

```
apache-airflow[hive]==1.10.12
```

Scarica le dipendenze

Amazon MWAA estrarrà il contenuto di plugins.zip in`/usr/local/airflow/pluginssu` ogni pianificatore e container di lavoro Amazon MWAA. Viene utilizzato per aggiungere binari al tuo ambiente. I passaggi seguenti descrivono come assemblare i file necessari per il plug-in personalizzato.

1. Nel prompt dei comandi, vai alla directory in cui desideri creare il tuo plugin. Ad esempio:

   ```
   cd plugins
   ```

2. ScaricaHadoopda unspecchio, ad esempio:

   ```
   wget https://downloads.apache.org/hadoop/common/hadoop-3.3.0/hadoop-3.3.0.tar.gz
   ```

3. ScaricaAlvearеda unspecchio, ad esempio:

   ```
   wget https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
   ```

4. Crea una directory. Ad esempio:

   ```
   mkdir hive_plugin
   ```

5. Estrai Hadoop.

   ```
   tar -xvzf hadoop-3.3.0.tar.gz -C hive_plugin
   ```

6. Estrai Hive.

```
tar -xvzf apache-hive-3.1.2-bin.tar.gz -C hive_plugin
```

Plugin personalizzato

Apache Airflow eseguirà il contenuto dei file Python nella cartella dei plugin all'avvio. Viene utilizzato per impostare e modificare le variabili di ambiente. I passaggi seguenti descrivono il codice di esempio per il plug-in personalizzato.

1. Nel prompt dei comandi, accedi a`hive_plugin`rubrica. Ad esempio:

```
cd hive_plugin
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`hive_plugin.py`nel`hive_plugin`rubrica.

```
from airflow.plugins_manager import AirflowPlugin
import os
os.environ["JAVA_HOME"]="/usr/lib/jvm/jre"
os.environ["HADOOP_HOME"]='/usr/local/airflow/plugins/hadoop-3.3.0'
os.environ["HADOOP_CONF_DIR"]='/usr/local/airflow/plugins/hadoop-3.3.0/etc/hadoop'
os.environ["HIVE_HOME"]='/usr/local/airflow/plugins/apache-hive-3.1.2-bin'
os.environ["PATH"] = os.getenv("PATH") + ":/usr/local/airflow/plugins/
hadoop-3.3.0:/usr/local/airflow/plugins/apache-hive-3.1.2-bin/bin:/usr/local/
airflow/plugins/apache-hive-3.1.2-bin/lib"
os.environ["CLASSPATH"] = os.getenv("CLASSPATH") + ":/usr/local/airflow/plugins/
apache-hive-3.1.2-bin/lib"
class EnvVarPlugin(AirflowPlugin):
    name = 'hive_plugin'
```

3. Copia il contenuto del seguente testo e salvalo localmente come`.airflowignore`nel`hive_plugin`rubrica.

```
hadoop-3.3.0
apache-hive-3.1.2-bin
```

Plugins.zip

I passaggi seguenti mostrano come creare`plugins.zip`. Il contenuto di questo esempio può essere combinato con altri plugin e binari in un unico`plugins.zip`fascicolo.

1. Nel prompt dei comandi, accedi a`hive_plugin`cartella del passaggio precedente. Ad esempio:

```
cd hive_plugin
```

2. Comprimi i contenuti all'interno del tuo`plugins`cartella.

```
zip -r ../hive_plugin.zip ./
```

Esempio di codice

I passaggi seguenti descrivono come creare il codice DAG che testerà il plug-in personalizzato.

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`hive.py`.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

with DAG(dag_id="hive_test_dag", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
    hive_test = BashOperator(
        task_id="hive_test",
        bash_command='hive --help'
    )
```

Opzioni di configurazione del flusso d'aria

Se utilizzi Apache Airflow v2, aggiungi`core.lazy_load_plugins : False`come opzione di configurazione di Apache Airflow. Per saperne di più, consultaUtilizzo delle opzioni di configurazione per caricare i plugin in 2.

Fasi successive

- Scopri come caricare`requirements.txt`inserisci questo esempio nel tuo bucket Amazon S3Installazione delle dipendenze in Python.
- Scopri come caricare il codice DAG in questo esempio su`dags`cartella nel tuo bucket Amazon S3 inAggiunta o aggiornamento di DAG.
- Scopri di più su come caricare`plugins.zip`inserisci questo esempio nel tuo bucket Amazon S3Installazione di plugin personalizzati.

Creazione di un plugin personalizzato per Apache AirflowPythonVirtualenvOperator

L'esempio seguente mostra come applicare una patch ad Apache AirflowPythonVirtualenvOperatorcon un plug-in personalizzato su Amazon Managed Workflows per Apache Airflow.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Requisiti
- Codice di esempio di plugin personalizzato
- Plugins.zip
- Esempio di codice
- Opzioni di configurazione del flusso d'aria
- Fasi successive

Versione

- Il codice di esempio in questa pagina può essere utilizzato conApache Airflow v1nelPython 3.7.

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- UnAmbiente Amazon MWAA.

Autorizzazioni

- Non sono necessarie autorizzazioni aggiuntive per utilizzare l'esempio di codice in questa pagina.

Requisiti

Per utilizzare il codice di esempio in questa pagina, aggiungi le seguenti dipendenze al tuo`requirements.txt`. Per ulteriori informazioni, consulta Installazione delle dipendenze in Python.

```
virtualenv
```

Codice di esempio di plugin personalizzato

Apache Airflow eseguirà il contenuto dei file Python nella cartella dei plugin all'avvio. Questo plugin correggerà il sistema integrato`PythonVirtualenvOperator`durante il processo di avvio per renderlo compatibile con Amazon MWAA. I passaggi seguenti mostrano il codice di esempio per il plug-in personalizzato.

Apache Airflow v2

1. Nel prompt dei comandi, accedi a`plugins`elenco qui sopra. Ad esempio:

   ```
   cd plugins
   ```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`virtual_python_plugin.py`.

   ```
   """
   Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

   Permission is hereby granted, free of charge, to any person obtaining a copy of
   this software and associated documentation files (the "Software"), to deal in
   the Software without restriction, including without limitation the rights to
   ```

```
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""
from airflow.plugins_manager import AirflowPlugin
import airflow.utils.python_virtualenv
from typing import List

def _generate_virtualenv_cmd(tmp_dir: str, python_bin: str,
 system_site_packages: bool) -> List[str]:
    cmd = ['python3','/usr/local/airflow/.local/lib/python3.7/site-packages/
virtualenv', tmp_dir]
    if system_site_packages:
        cmd.append('--system-site-packages')
    if python_bin is not None:
        cmd.append(f'--python={python_bin}')
    return cmd

airflow.utils.python_virtualenv._generate_virtualenv_cmd=_generate_virtualenv_cmd

class VirtualPythonPlugin(AirflowPlugin):
    name = 'virtual_python_plugin'
```

Apache Airflow v1

1. Nel prompt dei comandi, accedi a `plugins` elenco qui sopra. Ad esempio:

```
cd plugins
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come `virtual_python_plugin.py`.

```
from airflow.plugins_manager import AirflowPlugin
from airflow.operators.python_operator import PythonVirtualenvOperator

def _generate_virtualenv_cmd(self, tmp_dir):
```

```
    cmd = ['python3','/usr/local/airflow/.local/lib/python3.7/site-packages/
virtualenv', tmp_dir]
    if self.system_site_packages:
        cmd.append('--system-site-packages')
    if self.python_version is not None:
        cmd.append('--python=python{}'.format(self.python_version))
    return cmd
PythonVirtualenvOperator._generate_virtualenv_cmd=_generate_virtualenv_cmd

class EnvVarPlugin(AirflowPlugin):
    name = 'virtual_python_plugin'
```

Plugins.zip

I passaggi seguenti mostrano come creare`plugins.zip`.

1. Nel prompt dei comandi, accedi alla directory contenente`virtual_python_plugin.py`sopra. Ad esempio:

   ```
   cd plugins
   ```

2. Comprimi i contenuti all'interno del tuo`plugins`cartella.

   ```
   zip plugins.zip virtual_python_plugin.py
   ```

Esempio di codice

I passaggi seguenti descrivono come creare il codice DAG per il plug-in personalizzato.

Apache Airflow v2

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il codice DAG. Ad esempio:

   ```
   cd dags
   ```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`virtualenv_test.py`.

   ```
   """
   ```

```
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""
from airflow import DAG
from airflow.operators.python import PythonVirtualenvOperator
from airflow.utils.dates import days_ago
import os

os.environ["PATH"] = os.getenv("PATH") + ":/usr/local/airflow/.local/bin"

def virtualenv_fn():
    import boto3
    print("boto3 version ",boto3.__version__)

with DAG(dag_id="virtualenv_test", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
    virtualenv_task = PythonVirtualenvOperator(
        task_id="virtualenv_task",
        python_callable=virtualenv_fn,
        requirements=["boto3>=1.17.43"],
        system_site_packages=False,
        dag=dag,
    )
```

Apache Airflow v1

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`virtualenv_test.py`.

```
"""
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""
from airflow import DAG
from airflow.operators.python_operator import PythonVirtualenvOperator
from airflow.utils.dates import days_ago
import os

os.environ["PATH"] = os.getenv("PATH") + ":/usr/local/airflow/.local/bin"

def virtualenv_fn():
    import boto3
    print("boto3 version ",boto3.__version__)

with DAG(dag_id="virtualenv_test", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
    virtualenv_task = PythonVirtualenvOperator(
        task_id="virtualenv_task",
        python_callable=virtualenv_fn,
        requirements=["boto3>=1.17.43"],
        system_site_packages=False,
        dag=dag,
    )
```

Opzioni di configurazione del flusso d'aria

Se utilizzi Apache Airflow v2, aggiungi`core.lazy_load_plugins : False`come opzione di configurazione di Apache Airflow. Per saperne di più, consultaUtilizzo delle opzioni di configurazione per caricare i plugin in 2.

Fasi successive

- Scopri come caricare il`requirements.txt`inserisci questo esempio nel tuo bucket Amazon S3Installazione delle dipendenze in Python.
- Scopri come caricare il codice DAG in questo esempio su`dags`cartella nel tuo bucket Amazon S3 inAggiunta o aggiornamento di DAG.
- Scopri di più su come caricare il`plugins.zip`inserisci questo esempio nel tuo bucket Amazon S3Installazione di plugin personalizzati.

Invocare DAG con una funzione Lambda

Il seguente esempio di codice utilizza una AWS Lambdafunzione per ottenere un token CLI Apache Airflow e richiamare un grafo aciclico diretto (DAG) in un ambiente Amazon MWAA.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Dipendenze
- esempio di codice

Versione

- Puoi usare l'esempio di codice in questa pagina con Apache Airflow v2 e versioni successive in Python 3.10.

Prerequisiti

Per utilizzare questo esempio di codice, devi:

- Usa la modalità di accesso alla rete pubblica per il tuo ambiente Amazon MWAA.
- Disponi di una funzione Lambda che utilizzi l'ultimo runtime di Python.

> **Note**
>
> Se la funzione Lambda e l'ambiente Amazon MWAA si trovano nello stesso VPC, puoi usare questo codice su una rete privata. Per questa configurazione, il ruolo di esecuzione della funzione Lambda richiede l'autorizzazione per chiamare l'operazione dell'API Amazon Elastic Compute Cloud (Amazon EC2). CreateNetworkInterface Puoi fornire questa autorizzazione utilizzando la policy gestita `AWSLambdaVPCAccessExecutionRole` AWS .

Autorizzazioni

Per utilizzare l'esempio di codice in questa pagina, il ruolo di esecuzione del tuo ambiente Amazon MWAA deve avere accesso per eseguire l'azione`airflow:CreateCliToken`. Puoi fornire questa autorizzazione utilizzando la policy `AmazonMWAAAirflowCliAccess` AWS gestita:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "airflow:CreateCliToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Per ulteriori informazioni, consulta Politica CLI di Apache Airflow: AmazonMWAA Access AirflowCli.

Dipendenze

- Per utilizzare questo esempio di codice con Apache Airflow v2, non sono richieste dipendenze aggiuntive. Il codice utilizza l'installazione di base di Apache Airflow v2 nell'ambiente in uso.

esempio di codice

1. Apri la AWS Lambda console all'indirizzo https://console.aws.amazon.com/lambda/.
2. Scegli la tua funzione Lambda dall'elenco Funzioni.
3. Nella pagina della funzione, copia il codice seguente e sostituiscilo con i nomi delle tue risorse:
   - YOUR_ENVIRONMENT_NAME— Il nome del tuo ambiente Amazon MWAA.
   - YOUR_DAG_NAME— Il nome del DAG che desideri richiamare.

```
import boto3
import http.client
import base64
import ast
mwaa_env_name = 'YOUR_ENVIRONMENT_NAME'
dag_name = 'YOUR_DAG_NAME'
mwaa_cli_command = 'dags trigger'

client = boto3.client('mwaa')

def lambda_handler(event, context):
    # get web token
    mwaa_cli_token = client.create_cli_token(
        Name=mwaa_env_name
    )

    conn = http.client.HTTPSConnection(mwaa_cli_token['WebServerHostname'])
    payload = mwaa_cli_command + " " + dag_name
    headers = {
      'Authorization': 'Bearer ' + mwaa_cli_token['CliToken'],
      'Content-Type': 'text/plain'
    }
    conn.request("POST", "/aws_mwaa/cli", payload, headers)
    res = conn.getresponse()
    data = res.read()
    dict_str = data.decode("UTF-8")
    mydata = ast.literal_eval(dict_str)
    return base64.b64decode(mydata['stdout'])
```

4. Seleziona Deploy (Implementa).
5. Scegli Test per richiamare la tua funzione utilizzando la console Lambda.

6. Per verificare che Lambda abbia richiamato correttamente il tuo DAG, usa la console Amazon MWAA per accedere all'interfaccia utente Apache Airflow del tuo ambiente, quindi procedi come segue:

   a. Nella pagina DAG, individua il nuovo DAG di destinazione nell'elenco dei DAG.

   b. In Ultima esecuzione, controllate il timestamp dell'ultima esecuzione del DAG. Questo timestamp deve corrispondere molto da vicino al timestamp più recente utilizzato nell'altro ambiente. `invoke_dag`

   c. In Attività recenti, verifica che l'ultima esecuzione sia andata a buon fine.

Richiamo di DAG in diversi ambienti Amazon MWAA

L'esempio di codice seguente crea un token CLI di Apache Airflow. Il codice utilizza quindi un grafo aciclico diretto (DAG) in un ambiente Amazon MWAA per richiamare un DAG in un diverso ambiente Amazon MWAA.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Dipendenze
- Esempio di codice

Versione

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare l'esempio di codice in questa pagina, è necessario quanto segue:

- DueAmbienti Amazon MWAAconrete pubblicaaccesso al server Web, incluso l'ambiente corrente.
- Un DAG di esempio caricato nel bucket Amazon Simple Storage Service (Amazon S3) dell'ambiente di destinazione.

Autorizzazioni

Per utilizzare l'esempio di codice in questa pagina, il ruolo di esecuzione dell'ambiente deve disporre dell'autorizzazione per creare un token CLI di Apache Airflow. Puoi allegare ilAWSpolitica gestitaAmazonMWAAAirflowCliAccessper concedere questa autorizzazione.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "airflow:CreateCliToken"
            ],
            "Resource": "*"
        }
    ]
```

Per ulteriori informazioni, consulta Politica CLI di Apache Airflow: AmazonMWAA Access AirflowCli.

Dipendenze

- Per utilizzare questo esempio di codice con Apache Airflow v2, non sono necessarie dipendenze aggiuntive. Il codice utilizzaInstallazione base di Apache Airflow v2sul tuo ambiente.

Esempio di codice

L'esempio di codice seguente presuppone che tu stia utilizzando un DAG nell'ambiente corrente per richiamare un DAG in un altro ambiente.

1. Nel tuo terminale, accedi alla directory in cui è memorizzato il tuo codice DAG. Ad esempio:

   ```
   cd dags
   ```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente comeinvoke_dag.py. Sostituisci i seguenti valori con le tue informazioni.

   - your-new-environment-name— Il nome dell'altro ambiente in cui si desidera richiamare il DAG.
   - your-target-dag-id— L'ID del DAG nell'altro ambiente che desideri richiamare.

```
from airflow.decorators import dag, task
import boto3
from datetime import datetime, timedelta
import os, requests

DAG_ID = os.path.basename(__file__).replace(".py", "")

@task()
def invoke_dag_task(**kwargs):
    client = boto3.client('mwaa')
    token = client.create_cli_token(Name='your-new-environment-name')
    url = f"https://{token['WebServerHostname']}/aws_mwaa/cli"
    body = 'dags trigger your-target-dag-id'
    headers = {
        'Authorization' : 'Bearer ' + token['CliToken'],
        'Content-Type': 'text/plain'
        }
    requests.post(url, data=body, headers=headers)

@dag(
    dag_id=DAG_ID,
    schedule_interval=None,
    start_date=datetime(2022, 1, 1),
    dagrun_timeout=timedelta(minutes=60),
    catchup=False
    )
def invoke_dag():
    t = invoke_dag_task()

invoke_dag_test = invoke_dag()
```

3. Esegui quanto segueAWS CLIcomando per copiare il DAG nel bucket dell'ambiente, quindi attivare il DAG utilizzando l'interfaccia utente di Apache Airflow.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. Se il DAG viene eseguito correttamente, nei registri delle attività verrà visualizzato un output simile al seguente`invoke_dag_task`.

```
[2022-01-01, 12:00:00 PDT] {{python.py:152}} INFO - Done. Returned value was: None
```

```
[2022-01-01, 12:00:00 PDT] {{taskinstance.py:1280}} INFO - Marking task as SUCCESS.
 dag_id=invoke_dag, task_id=invoke_dag_task, execution_date=20220101T120000,
 start_date=20220101T120000, end_date=20220101T120000
[2022-01-01, 12:00:00 PDT] {{local_task_job.py:154}} INFO - Task exited with return
 code 0
[2022-01-01, 12:00:00 PDT] {{local_task_job.py:264}} INFO - 0 downstream tasks
 scheduled from follow-on schedule check
```

Per verificare che il tuo DAG sia stato richiamato correttamente, accedi all'interfaccia utente di Apache Airflow per il tuo nuovo ambiente, quindi procedi come segue:

a. SulDAGpagina, individua il tuo nuovo DAG di destinazione nell'elenco dei DAG.

b. SottoUltima corsa, controlla il timestamp per l'ultima esecuzione del DAG. Questo timestamp deve corrispondere strettamente al timestamp più recente per`invoke_dag`nel tuo altro ambiente.

c. SottoAttività recenti, controlla che l'ultima corsa sia andata a buon fine.

Utilizzo di Amazon MWAA con Amazon RDS per Microsoft SQL Server

Puoi utilizzare Amazon Managed Workflows per Apache Airflow per connetterti a unRDS per SQL Server. Il codice di esempio seguente utilizza DAG su un ambiente Amazon Managed Workflows for Apache Airflow per connettersi ed eseguire query su Amazon RDS per Microsoft SQL Server.

Argomenti

- Versione
- Prerequisiti
- Dipendenze
- Connessione Apache Airflow v2
- Esempio di codice
- Fasi successive

Versione

- Il codice di esempio in questa pagina può essere utilizzato conApache Airflow v1nelPython 3.7.

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- UnAmbiente Amazon MWAA.
- Amazon MWAA e RDS per SQL Server vengono eseguiti nello stesso Amazon VPC/
- I gruppi di sicurezza VPC di Amazon MWAA e del server sono configurati con le seguenti connessioni:
  - Una regola in entrata per la porta`1433`apri ad Amazon RDS nel gruppo di sicurezza di Amazon MWAA
  - O una regola in uscita per il porto di`1433`apri da Amazon MWAA a RDS
- Apache Airflow Connection for RDS for SQL Server riflette il nome host, la porta, il nome utente e la password del database Amazon RDS SQL Server creato nel processo precedente.

Dipendenze

Per utilizzare il codice di esempio in questa sezione, aggiungi la seguente dipendenza al tuo`requirements.txt`. Per ulteriori informazioni, consulta Installazione delle dipendenze in Python

Apache Airflow v2

```
apache-airflow-providers-microsoft-mssql==1.0.1
apache-airflow-providers-odbc==1.0.1
pymssql==2.2.1
```

Apache Airflow v1

```
apache-airflow[mssql]==1.10.12
```

Connessione Apache Airflow v2

Se utilizzate una connessione in Apache Airflow v2, assicuratevi che l'oggetto di connessione Airflow includa le seguenti coppie chiave-valore:

1. ID di accesso:mssql_default
2. Tipo di connessione:Servizi Web Amazon
3. Ospite: YOUR_DB_HOST
4. Schema:
5. Accedi:amministratore
6. Password:
7. Porta:1433
8. Extra:

Esempio di codice

1. Nel prompt dei comandi, accedi alla directory in cui è archiviato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`sql-server.py`.

```
"""
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""
import pymssql
import logging
import sys
from airflow import DAG
from datetime import datetime
from airflow.operators.mssql_operator import MsSqlOperator
from airflow.operators.python_operator import PythonOperator
```

```
default_args = {
    'owner': 'aws',
    'depends_on_past': False,
    'start_date': datetime(2019, 2, 20),
    'provide_context': True
}

dag = DAG(
    'mssql_conn_example', default_args=default_args, schedule_interval=None)

drop_db = MsSqlOperator(
   task_id="drop_db",
   sql="DROP DATABASE IF EXISTS testdb;",
   mssql_conn_id="mssql_default",
   autocommit=True,
   dag=dag
)

create_db = MsSqlOperator(
   task_id="create_db",
   sql="create database testdb;",
   mssql_conn_id="mssql_default",
   autocommit=True,
   dag=dag
)

create_table = MsSqlOperator(
   task_id="create_table",
   sql="CREATE TABLE testdb.dbo.pet (name VARCHAR(20), owner VARCHAR(20));",
   mssql_conn_id="mssql_default",
   autocommit=True,
   dag=dag
)

insert_into_table = MsSqlOperator(
   task_id="insert_into_table",
   sql="INSERT INTO testdb.dbo.pet VALUES ('Olaf', 'Disney');",
   mssql_conn_id="mssql_default",
   autocommit=True,
   dag=dag
)

def select_pet(**kwargs):
   try:
```

```
        conn = pymssql.connect(
            server='sampledb.<xxxxxx>.<region>.rds.amazonaws.com',
            user='admin',
            password='<yoursupersecretpassword>',
            database='testdb'
        )

        # Create a cursor from the connection
        cursor = conn.cursor()
        cursor.execute("SELECT * from testdb.dbo.pet")
        row = cursor.fetchone()

        if row:
            print(row)
   except:
      logging.error("Error when creating pymssql database connection: %s",
 sys.exc_info()[0])

select_query = PythonOperator(
    task_id='select_query',
    python_callable=select_pet,
    dag=dag,
)

drop_db >> create_db >> create_table >> insert_into_table >> select_query
```

Fasi successive

- Scopri come caricare il`requirements.txt`inserisci questo esempio nel tuo bucket Amazon S3Installazione delle dipendenze in Python.
- Scopri come caricare il codice DAG in questo esempio su`dags`cartella nel tuo bucket Amazon S3 inAggiunta o aggiornamento di DAG.
- Esplora script di esempio e altroesempi di moduli pymssql.
- Scopri di più sull'esecuzione di codice SQL in uno specifico database Microsoft SQL utilizzandooperatore mssqlnelGuida di riferimento di Apache Airflow.

Utilizzo di Amazon MWAA con Amazon EMR

Il seguente esempio di codice dimostra come abilitare un'integrazione utilizzando Amazon EMR e Amazon Managed Workflows for Apache Airflow.

Argomenti

- Versione
- Esempio di codice

Versione

- Il codice di esempio in questa pagina può essere utilizzato con Apache Airflow v1 in Python 3.7.

Esempio di codice

```
    """
    Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

    Permission is hereby granted, free of charge, to any person obtaining a copy of
    this software and associated documentation files (the "Software"), to deal in
    the Software without restriction, including without limitation the rights to
    use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
    the Software, and to permit persons to whom the Software is furnished to do so.

    THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
    IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
    FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
    COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
    IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
    CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
    """
    from airflow import DAG

    from airflow.contrib.operators.emr_add_steps_operator import EmrAddStepsOperator
    from airflow.contrib.operators.emr_create_job_flow_operator import
EmrCreateJobFlowOperator
    from airflow.contrib.sensors.emr_step_sensor import EmrStepSensor

    from airflow.utils.dates import days_ago
    from datetime import timedelta
```

```
import os

DAG_ID = os.path.basename(__file__).replace(".py", "")

DEFAULT_ARGS = {
    'owner': 'airflow',
    'depends_on_past': False,
    'email': ['airflow@example.com'],
    'email_on_failure': False,
    'email_on_retry': False,
}

SPARK_STEPS = [
    {
        'Name': 'calculate_pi',
        'ActionOnFailure': 'CONTINUE',
        'HadoopJarStep': {
            'Jar': 'command-runner.jar',
            'Args': ['/usr/lib/spark/bin/run-example', 'SparkPi', '10'],
        },
    }
]

JOB_FLOW_OVERRIDES = {
    'Name': 'my-demo-cluster',
    'ReleaseLabel': 'emr-5.30.1',
    'Applications': [
        {
            'Name': 'Spark'
        },
    ],
    'Instances': {
        'InstanceGroups': [
            {
                'Name': "Master nodes",
                'Market': 'ON_DEMAND',
                'InstanceRole': 'MASTER',
                'InstanceType': 'm5.xlarge',
                'InstanceCount': 1,
            },
            {
                'Name': "Slave nodes",
                'Market': 'ON_DEMAND',
                'InstanceRole': 'CORE',
```

```
                    'InstanceType': 'm5.xlarge',
                    'InstanceCount': 2,
                }
            ],
            'KeepJobFlowAliveWhenNoSteps': False,
            'TerminationProtected': False,
            'Ec2KeyName': 'mykeypair',
        },
        'VisibleToAllUsers': True,
        'JobFlowRole': 'EMR_EC2_DefaultRole',
        'ServiceRole': 'EMR_DefaultRole'
    }

    with DAG(
        dag_id=DAG_ID,
        default_args=DEFAULT_ARGS,
        dagrun_timeout=timedelta(hours=2),
        start_date=days_ago(1),
        schedule_interval='@once',
        tags=['emr'],
    ) as dag:

        cluster_creator = EmrCreateJobFlowOperator(
            task_id='create_job_flow',
            job_flow_overrides=JOB_FLOW_OVERRIDES
        )

        step_adder = EmrAddStepsOperator(
            task_id='add_steps',
            job_flow_id="{{ task_instance.xcom_pull(task_ids='create_job_flow',
key='return_value') }}",
            aws_conn_id='aws_default',
            steps=SPARK_STEPS,
        )

        step_checker = EmrStepSensor(
            task_id='watch_step',
            job_flow_id="{{ task_instance.xcom_pull('create_job_flow',
key='return_value') }}",
            step_id="{{ task_instance.xcom_pull(task_ids='add_steps',
key='return_value')[0] }}",
            aws_conn_id='aws_default',
        )
```

```
        cluster_creator >> step_adder >> step_checker
```

Utilizzo di Amazon MWAA con Amazon EKS

L'esempio seguente dimostra come utilizzare Amazon Managed Workflows per Apache Airflow con Amazon EKS.

Argomenti

- Versione
- Prerequisiti
- Crea una chiave pubblica per Amazon EC2
- Crea il cluster
- Creare unmwaanamespace
- Crea un ruolo permwaanamespace
- Crea e associa un ruolo IAM per il cluster Amazon EKS
- Creare il file requirements.txt
- Crea una mappatura dell'identità per Amazon EKS
- Creazione del kubeconfig
- Crea un DAG
- Aggiungi il DAG ekube_config.yamlal bucket Amazon S3
- Abilita e attiva l'esempio

Versione

- Il codice di esempio in questa pagina può essere utilizzato conApache Airflow v1nelPython 3.7.

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare l'esempio in questo argomento, avrai bisogno di quanto segue:

- UnAmbiente Amazon MWAA.
- ecctl. Per saperne di più, consultaInstalare eksctl.
- kubectl. Per saperne di più, consultaInstalla e configura kubectl. In alcuni casi viene installato con eksctl.
- Una coppia di chiavi EC2 nella regione in cui crei il tuo ambiente Amazon MWAA. Per saperne di più, consultaCreazione o importazione di una coppia di chiavi.

> **Note**
>
> Quando si utilizza un`eksctl`comando, puoi includere un`--profile`per specificare un profilo diverso da quello predefinito.

Crea una chiave pubblica per Amazon EC2

Usa il comando seguente per creare una chiave pubblica dalla tua coppia di chiavi private.

```
ssh-keygen -y -f myprivatekey.pem > mypublickey.pub
```

Per saperne di più, consultaRecupero della chiave pubblica per la coppia di chiavi.

Crea il cluster

Usa il comando seguente per creare il cluster. Se desideri un nome personalizzato per il cluster o per crearlo in una regione diversa, sostituisci i valori del nome e della regione. Devi creare il cluster nella stessa regione in cui crei l'ambiente Amazon MWAA. Sostituisci i valori delle sottoreti in modo che corrispondano alle sottoreti della tua rete Amazon VPC che utilizzi per Amazon MWAA. Sostituisci il valore per`ssh-public-key`in base alla chiave utilizzata. Puoi utilizzare una chiave esistente di Amazon EC2 che si trova nella stessa regione o crearne una nuova nella stessa regione in cui hai creato il tuo ambiente Amazon MWAA.

```
eksctl create cluster \
--name mwaa-eks \
--region us-west-2 \
--version 1.18 \
--nodegroup-name linux-nodes \
```

```
--nodes 3 \
--nodes-min 1 \
--nodes-max 4 \
--with-oidc \
--ssh-access \
--ssh-public-key MyPublicKey \
--managed \
--vpc-public-subnets "subnet-11111111111111111, subnet-2222222222222222" \
--vpc-private-subnets "subnet-33333333333333333, subnet-44444444444444444"
```

Il completamento della creazione del cluster richiede del tempo. Una volta completato, puoi verificare che il cluster sia stato creato correttamente e che il provider IAM OIDC sia configurato utilizzando il seguente comando:

```
eksctl utils associate-iam-oidc-provider \
--region us-west-2 \
--cluster mwaa-eks \
--approve
```

Creare un**mwaa**namespace

Dopo aver confermato che il cluster è stato creato correttamente, usa il seguente comando per creare un namespace per i pod.

```
kubectl create namespace mwaa
```

Crea un ruolo per**mwaa**namespace

Dopo aver creato il namespace, crea un ruolo e un'associazione di ruoli per un utente Amazon MWAA su EKS in grado di eseguire i pod in uno spazio dei nomi MWAA. Se hai usato un nome diverso per il namespace, sostituisci mwaa in `-n` *mwaa*con il nome che hai usato.

```
cat << EOF | kubectl apply -f - -n mwaa
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: mwaa-role
rules:
  - apiGroups:
      - ""
```

```
      - "apps"
      - "batch"
      - "extensions"
    resources:
      - "jobs"
      - "pods"
      - "pods/attach"
      - "pods/exec"
      - "pods/log"
      - "pods/portforward"
      - "secrets"
      - "services"
    verbs:
      - "create"
      - "delete"
      - "describe"
      - "get"
      - "list"
      - "patch"
      - "update"
---
kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
  name: mwaa-role-binding
subjects:
- kind: User
  name: mwaa-service
roleRef:
  kind: Role
  name: mwaa-role
  apiGroup: rbac.authorization.k8s.io
EOF
```

Conferma che il nuovo ruolo possa accedere al cluster Amazon EKS eseguendo il seguente comando. Assicurati di usare il nome corretto se non l'hai usato*mwaa*:

```
kubectl get pods -n mwaa --as mwaa-service
```

Dovresti vedere un messaggio restituito che dice:

```
No resources found in mwaa namespace.
```

Crea e associa un ruolo IAM per il cluster Amazon EKS

Devi creare un ruolo IAM e quindi associarlo al cluster Amazon EKS (k8s) in modo che possa essere utilizzato per l'autenticazione tramite IAM. Il ruolo viene utilizzato solo per accedere al cluster e non dispone di autorizzazioni per la console o le chiamate API.

Crea un nuovo ruolo per l'ambiente Amazon MWAA seguendo la procedura descritta in[Ruolo di esecuzione di Amazon MWAA](). Tuttavia, anziché creare e allegare le politiche descritte in quell'argomento, allega la seguente politica:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "airflow:PublishMetrics",
            "Resource": "arn:aws:airflow:${MWAA_REGION}:${ACCOUNT_NUMBER}:environment/
${MWAA_ENV_NAME}"
        },
        {
            "Effect": "Deny",
            "Action": "s3:ListAllMyBuckets",
            "Resource": [
                "arn:aws:s3:::{MWAA_S3_BUCKET}",
                "arn:aws:s3:::{MWAA_S3_BUCKET}/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject*",
                "s3:GetBucket*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::{MWAA_S3_BUCKET}",
                "arn:aws:s3:::{MWAA_S3_BUCKET}/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogStream",
```

```
                "logs:CreateLogGroup",
                "logs:PutLogEvents",
                "logs:GetLogEvents",
                "logs:GetLogRecord",
                "logs:GetLogGroupFields",
                "logs:GetQueryResults",
                "logs:DescribeLogGroups"
            ],
            "Resource": [
                "arn:aws:logs:${MWAA_REGION}:${ACCOUNT_NUMBER}:log-group:airflow-
${MWAA_ENV_NAME}-*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "cloudwatch:PutMetricData",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "sqs:ChangeMessageVisibility",
                "sqs:DeleteMessage",
                "sqs:GetQueueAttributes",
                "sqs:GetQueueUrl",
                "sqs:ReceiveMessage",
                "sqs:SendMessage"
            ],
            "Resource": "arn:aws:sqs:${MWAA_REGION}:*:airflow-celery-*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey",
                "kms:GenerateDataKey*",
                "kms:Encrypt"
            ],
            "NotResource": "arn:aws:kms:*:${ACCOUNT_NUMBER}:key/*",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": [
                        "sqs.${MWAA_REGION}.amazonaws.com"
                    ]
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "eks:DescribeCluster"
            ],
            "Resource": "arn:aws:eks:${MWAA_REGION}:${ACCOUNT_NUMBER}:cluster/
${EKS_CLUSTER_NAME}"
        }
    ]
}
```

Dopo aver creato il ruolo, modifica il tuo ambiente Amazon MWAA per utilizzare il ruolo che hai creato come ruolo di esecuzione per l'ambiente. Per cambiare il ruolo, modifica l'ambiente da utilizzare. Seleziona il ruolo di esecuzione inAutorizzazioni.

Problemi noti:

- Esiste un problema noto relativo agli ARN di ruolo con percorsi secondari che non sono in grado di autenticarsi con Amazon EKS. La soluzione alternativa consiste nel creare il ruolo del servizio manualmente anziché utilizzare quello creato da Amazon MWAA stesso. Per saperne di più, consultal ruoli con percorsi non funzionano quando il percorso è incluso nel loro ARN nella configmap di aws-auth
- Se l'elenco dei servizi Amazon MWAA non è disponibile in IAM, devi scegliere una politica di servizio alternativa, come Amazon EC2, e quindi aggiornare la politica di fiducia del ruolo in modo che corrisponda a quanto segue:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": [
          "airflow-env.amazonaws.com",
          "airflow.amazonaws.com"
        ]
      },
      "Action": "sts:AssumeRole"
```

```
    }
  ]
}
```

Per saperne di più, consultaCome utilizzare le politiche di fiducia con i ruoli IAM.

Creare il file requirements.txt

Per utilizzare il codice di esempio in questa sezione, assicurati di aver aggiunto una delle seguenti opzioni di database al tuo`requirements.txt`. Per ulteriori informazioni, consulta Installazione delle dipendenze in Python.

Apache Airflow v2

```
kubernetes
apache-airflow[cncf.kubernetes]==3.0.0
```

Apache Airflow v1

```
awscli
kubernetes==12.0.1
```

Crea una mappatura dell'identità per Amazon EKS

Usa l'ARN per il ruolo che hai creato nel comando seguente per creare una mappatura dell'identità per Amazon EKS. Cambia la regione*la tua regione*nella regione in cui hai creato l'ambiente. Sostituisci l'ARN per il ruolo e, infine, sostituisci*mwaa-execution-role*con il ruolo di esecuzione del tuo ambiente.

```
eksctl create iamidentitymapping \
--region your-region \
--cluster mwaa-eks \
--arn arn:aws:iam::111222333444:role/mwaa-execution-role \
--username mwaa-service
```

Creazione del **kubeconfig**

Utilizzate il comando seguente per creare`kubeconfig`:

```
aws eks update-kubeconfig \
--region us-west-2 \
--kubeconfig ./kube_config.yaml \
--name mwaa-eks \
--alias aws
```

Se hai utilizzato un profilo specifico durante la corsa`update-kubeconfig`è necessario rimuovere il`env:`sezione aggiunta al file kube_config.yaml in modo che funzioni correttamente con Amazon MWAA. A tale scopo, elimina quanto segue dal file e quindi salvalo:

```
env:
- name: AWS_PROFILE
  value: profile_name
```

Crea un DAG

Usa il seguente esempio di codice per creare un file Python, ad esempio`mwaa_pod_example.py`per il DAG.

Apache Airflow v2

```
"""
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""
from airflow import DAG
from datetime import datetime
from airflow.providers.cncf.kubernetes.operators.kubernetes_pod import
 KubernetesPodOperator

default_args = {
```

```
   'owner': 'aws',
   'depends_on_past': False,
   'start_date': datetime(2019, 2, 20),
   'provide_context': True
}

dag = DAG(
   'kubernetes_pod_example', default_args=default_args, schedule_interval=None)

#use a kube_config stored in s3 dags folder for now
kube_config_path = '/usr/local/airflow/dags/kube_config.yaml'

podRun = KubernetesPodOperator(
                        namespace="mwaa",
                        image="ubuntu:18.04",
                        cmds=["bash"],
                        arguments=["-c", "ls"],
                        labels={"foo": "bar"},
                        name="mwaa-pod-test",
                        task_id="pod-task",
                        get_logs=True,
                        dag=dag,
                        is_delete_operator_pod=False,
                        config_file=kube_config_path,
                        in_cluster=False,
                        cluster_context='aws'
                        )
```

Apache Airflow v1

```
"""
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

```
"""
from airflow import DAG
from datetime import datetime
from airflow.contrib.operators.kubernetes_pod_operator import KubernetesPodOperator

default_args = {
   'owner': 'aws',
   'depends_on_past': False,
   'start_date': datetime(2019, 2, 20),
   'provide_context': True
}

dag = DAG(
   'kubernetes_pod_example', default_args=default_args, schedule_interval=None)

#use a kube_config stored in s3 dags folder for now
kube_config_path = '/usr/local/airflow/dags/kube_config.yaml'

podRun = KubernetesPodOperator(
                       namespace="mwaa",
                       image="ubuntu:18.04",
                       cmds=["bash"],
                       arguments=["-c", "ls"],
                       labels={"foo": "bar"},
                       name="mwaa-pod-test",
                       task_id="pod-task",
                       get_logs=True,
                       dag=dag,
                       is_delete_operator_pod=False,
                       config_file=kube_config_path,
                       in_cluster=False,
                       cluster_context='aws'
                       )
```

Aggiungi il DAG e**`kube_config.yaml`**al bucket Amazon S3

Inserisci il DAG che hai creato e il`kube_config.yaml`file nel bucket Amazon S3 per l'ambiente Amazon MWAA. Puoi inserire file nel tuo bucket utilizzando la console Amazon S3 o ilAWS Command Line Interface.

Abilita e attiva l'esempio

In Apache Airflow, abilitate l'esempio e quindi attivatelo.

Dopo l'esecuzione e il completamento con successo, usa il seguente comando per verificare il pod:

```
kubectl get pods -n mwaa
```

Verrà visualizzato un output simile al seguente:

```
NAME READY STATUS RESTARTS AGE
mwaa-pod-test-aa11bb22cc3344445555666677778888 0/1 Completed 0 2m23s
```

È quindi possibile verificare l'output del pod con il seguente comando. Sostituisci il valore del nome con il valore restituito dal comando precedente:

```
kubectl logs -n mwaa mwaa-pod-test-aa11bb22cc3344445555666677778888
```

Connessione ad Amazon ECS tramite`ECSOperator`

L'argomento descrive come è possibile utilizzare`ECSOperator`per connettersi a un container Amazon Elastic Container Service (Amazon ECS) da Amazon MWAA. Nei passaggi seguenti, aggiungerai le autorizzazioni richieste al ruolo di esecuzione del tuo ambiente, usa unAWS CloudFormationmodello per creare un cluster Amazon ECS Fargate e infine creare e caricare un DAG che si connette al nuovo cluster.

Argomenti

- Versione
- Prerequisiti
- Autorizzazioni
- Crea un cluster Amazon ECS
- Esempio di codice

Versione

- Puoi usare l'esempio di codice in questa pagina conApache Airflow v2 e versioni successivenelPython 3.10.

Prerequisiti

Per utilizzare il codice di esempio in questa pagina, avrai bisogno di quanto segue:

- UnAmbiente Amazon MWAA.

Autorizzazioni

- Il ruolo di esecuzione per il tuo ambiente richiede l'autorizzazione per eseguire attività in Amazon ECS. Puoi allegare ilAmazonECS_FullAccess AWS-policy gestita al tuo ruolo di esecuzione oppure crea e allega la seguente politica al tuo ruolo di esecuzione.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "ecs:RunTask",
                "ecs:DescribeTasks"
            ],
            "Resource": "*"
        },
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringLike": {
                    "iam:PassedToService": "ecs-tasks.amazonaws.com"
                }
            }
        }
    ]
}
```

- Oltre ad aggiungere le autorizzazioni necessarie per eseguire attività in Amazon ECS, devi anche modificare ilCloudWatchRegistra la dichiarazione politica nel tuo ruolo di esecuzione di Amazon

MWAA per consentire l'accesso al gruppo di log delle attività di Amazon ECS, come illustrato di seguito. Il gruppo di log Amazon ECS è stato creato daAWS CloudFormationmodello inthe section called “Crea un cluster Amazon ECS”.

```
{
    "Effect": "Allow",
    "Action": [
        "logs:CreateLogStream",
        "logs:CreateLogGroup",
        "logs:PutLogEvents",
        "logs:GetLogEvents",
        "logs:GetLogRecord",
        "logs:GetLogGroupFields",
        "logs:GetQueryResults"
    ],
    "Resource": [
        "arn:aws:logs:region:account-id:log-group:airflow-environment-name-*",
        "arn:aws:logs:*:*:log-group:ecs-mwaa-group:*"
    ]
}
```

Per ulteriori informazioni sul ruolo di esecuzione di Amazon MWAA e su come allegare una policy, consultaRuolo di esecuzione.

Crea un cluster Amazon ECS

Utilizzando quanto segueAWS CloudFormationmodello, creerai un cluster Amazon ECS Fargate da utilizzare con il tuo flusso di lavoro Amazon MWAA. Per ulteriori informazioni, vedereCreazione di una definizione di attivitànelGuida per gli sviluppatori di Amazon Elastic Container Service.

1. Crea un file JSON con il seguente codice e salvalo comeecs-mwaa-cfn.json.

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "This template deploys an ECS Fargate cluster with an Amazon
 Linux image as a test for MWAA.",
    "Parameters": {
        "VpcId": {
            "Type": "AWS::EC2::VPC::Id",
```

```
            "Description": "Select a VPC that allows instances access to ECR, as
used with MWAA."
        },
        "SubnetIds": {
            "Type": "List<AWS::EC2::Subnet::Id>",
            "Description": "Select at two private subnets in your selected VPC, as
used with MWAA."
        },
        "SecurityGroups": {
            "Type": "List<AWS::EC2::SecurityGroup::Id>",
            "Description": "Select at least one security group in your selected
VPC, as used with MWAA."
        }
    },
    "Resources": {
        "Cluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": {
                    "Fn::Sub": "${AWS::StackName}-cluster"
                }
            }
        },
        "LogGroup": {
            "Type": "AWS::Logs::LogGroup",
            "Properties": {
                "LogGroupName": {
                    "Ref": "AWS::StackName"
                },
                "RetentionInDays": 30
            }
        },
        "ExecutionRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": "ecs-tasks.amazonaws.com"
                            },
                            "Action": "sts:AssumeRole"
                        }
```

```
                    ]
                },
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
                ]
            }
        },
        "TaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "Family": {
                    "Fn::Sub": "${AWS::StackName}-task"
                },
                "Cpu": 2048,
                "Memory": 4096,
                "NetworkMode": "awsvpc",
                "ExecutionRoleArn": {
                    "Ref": "ExecutionRole"
                },
                "ContainerDefinitions": [
                    {
                        "Name": {
                            "Fn::Sub": "${AWS::StackName}-container"
                        },
                        "Image": "137112412989.dkr.ecr.us-east-1.amazonaws.com/
amazonlinux:latest",
                        "PortMappings": [
                            {
                                "Protocol": "tcp",
                                "ContainerPort": 8080,
                                "HostPort": 8080
                            }
                        ],
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "awslogs-region": {
                                    "Ref": "AWS::Region"
                                },
                                "awslogs-group": {
                                    "Ref": "LogGroup"
                                },
                                "awslogs-stream-prefix": "ecs"
```

```
                            }
                        }
                    }
                ],
                "RequiresCompatibilities": [
                    "FARGATE"
                ]
            }
        },
        "Service": {
            "Type": "AWS::ECS::Service",
            "Properties": {
                "ServiceName": {
                    "Fn::Sub": "${AWS::StackName}-service"
                },
                "Cluster": {
                    "Ref": "Cluster"
                },
                "TaskDefinition": {
                    "Ref": "TaskDefinition"
                },
                "DesiredCount": 1,
                "LaunchType": "FARGATE",
                "PlatformVersion": "1.3.0",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "ENABLED",
                        "Subnets": {
                            "Ref": "SubnetIds"
                        },
                        "SecurityGroups": {
                            "Ref": "SecurityGroups"
                        }
                    }
                }
            }
        }
    }
}
```

2. Nel prompt dei comandi, usa quanto segueAWS CLIcomando per creare un nuovo stack. È necessario sostituire i valori`SecurityGroups`e`SubnetIds`con valori per i gruppi di sicurezza e le sottoreti del tuo ambiente Amazon MWAA.

```
$ aws cloudformation create-stack \
--stack-name my-ecs-stack --template-body file://ecs-mwaa-cfn.json \
--parameters ParameterKey=SecurityGroups,ParameterValue=your-mwaa-security-group \
ParameterKey=SubnetIds,ParameterValue=your-mwaa-subnet-1\\,your-mwaa-subnet-1 \
--capabilities CAPABILITY_IAM
```

In alternativa, puoi usare il seguente script di shell. Lo script recupera i valori richiesti per i gruppi di sicurezza e le sottoreti dell'ambiente utilizzando il[get-environment](#) AWS CLIcomando, quindi crea lo stack di conseguenza. Per eseguire lo script, effettuate le seguenti operazioni.

a. Copiare e salvare lo script con nome`ecs-stack-helper.sh`nella stessa cartella della tuaAWS CloudFormationmodello.

```
#!/bin/bash

joinByString() {
  local separator="$1"
  shift
  local first="$1"
  shift
  printf "%s" "$first" "${@/#/$separator}"
}

response=$(aws mwaa get-environment --name $1)

securityGroupId=$(echo "$response" | jq -r
 '.Environment.NetworkConfiguration.SecurityGroupIds[]')
subnetIds=$(joinByString '\,' $(echo "$response" | jq -r
 '.Environment.NetworkConfiguration.SubnetIds[]'))

aws cloudformation create-stack --stack-name $2 --template-body file://ecs-
cfn.json \
--parameters ParameterKey=SecurityGroups,ParameterValue=$securityGroupId \
ParameterKey=SubnetIds,ParameterValue=$subnetIds \
--capabilities CAPABILITY_IAM
```

b. Esegui lo script utilizzando i seguenti comandi. Sostituiscienvironment-name`stack-name`con i tuoi dati.

```
$ chmod +x ecs-stack-helper.sh
```

```
$ ./ecs-stack-helper.bash environment-name stack-name
```

In caso di successo, vedrai il seguente output che mostra il tuo nuovoAWS CloudFormationID dello stack.

```
{
    "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-ecs-
stack/123456e7-8ab9-01cd-b2fb-36cce63786c9"
}
```

Dopo il tuoAWS CloudFormationlo stack è completato eAWSha fornito le tue risorse Amazon ECS, sei pronto per creare e caricare il tuo DAG.

Esempio di codice

1. Apri un prompt dei comandi e vai alla directory in cui è archiviato il codice DAG. Ad esempio:

```
cd dags
```

2. Copia il contenuto del seguente esempio di codice e salvalo localmente come`mwaa-ecs-operator.py`, quindi carica il tuo nuovo DAG su Amazon S3.

```
from http import client
from airflow import DAG
from airflow.providers.amazon.aws.operators.ecs import ECSOperator
from airflow.utils.dates import days_ago
import boto3

CLUSTER_NAME="mwaa-ecs-test-cluster" #Replace value for CLUSTER_NAME with your
 information.
CONTAINER_NAME="mwaa-ecs-test-container" #Replace value for CONTAINER_NAME with
 your information.
LAUNCH_TYPE="FARGATE"

with DAG(
    dag_id = "ecs_fargate_dag",
    schedule_interval=None,
    catchup=False,
    start_date=days_ago(1)
) as dag:
```

```
    client=boto3.client('ecs')
    services=client.list_services(cluster=CLUSTER_NAME,launchType=LAUNCH_TYPE)

service=client.describe_services(cluster=CLUSTER_NAME,services=services['serviceArns'])

    ecs_operator_task = ECSOperator(
        task_id = "ecs_operator_task",
        dag=dag,
        cluster=CLUSTER_NAME,
        task_definition=service['services'][0]['taskDefinition'],
        launch_type=LAUNCH_TYPE,
        overrides={
            "containerOverrides":[
                {
                    "name":CONTAINER_NAME,
                    "command":["ls", "-l", "/"],
                },
            ],
        },

        network_configuration=service['services'][0]['networkConfiguration'],
        awslogs_group="mwaa-ecs-zero",
        awslogs_stream_prefix=f"ecs/{CONTAINER_NAME}",
    )
```

> **Note**
>
> Nell'esempio DAG, per`awslogs_group`, potrebbe essere necessario modificare il gruppo di log con il nome del gruppo di log delle attività Amazon ECS. L'esempio presuppone un gruppo di log denominato`mwaa-ecs-zero`. Per`awslogs_stream_prefix`, utilizza il prefisso del flusso di log dei task di Amazon ECS. L'esempio presuppone un prefisso del flusso di log,`ecs`.

3. Esegui quanto segueAWS CLIcomando per copiare il DAG nel bucket dell'ambiente, quindi attivare il DAG utilizzando l'interfaccia utente di Apache Airflow.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. In caso di successo, vedrai un output simile al seguente nei registri delle attività per`ecs_operator_task`nel`ecs_fargate_dag`GIORNO:

```
[2022-01-01, 12:00:00 UTC] {{ecs.py:300}} INFO - Running ECS Task -
Task definition: arn:aws:ecs:us-west-2:123456789012:task-definition/mwaa-ecs-test-
task:1 - on cluster mwaa-ecs-test-cluster
[2022-01-01, 12:00:00 UTC] {{ecs-operator-test.py:302}} INFO - ECSOperator
 overrides:
{'containerOverrides': [{'name': 'mwaa-ecs-test-container', 'command': ['ls', '-l',
 '/']}]}
.
.
.
[2022-01-01, 12:00:00 UTC] {{ecs.py:379}} INFO - ECS task ID is:
 e012340b5e1b43c6a757cf012c635935
[2022-01-01, 12:00:00 UTC] {{ecs.py:313}} INFO - Starting ECS Task Log Fetcher
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] total
 52
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC]
 lrwxrwxrwx   1 root root    7 Jun 13 18:51 bin -> usr/bin
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] dr-xr-
xr-x   2 root root 4096 Apr  9  2019 boot
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x   5 root root  340 Jul 19 17:54 dev
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x   1 root root 4096 Jul 19 17:54 etc
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x   2 root root 4096 Apr  9  2019 home
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC]
 lrwxrwxrwx   1 root root    7 Jun 13 18:51 lib -> usr/lib
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC]
 lrwxrwxrwx   1 root root    9 Jun 13 18:51 lib64 -> usr/lib64
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x   2 root root 4096 Jun 13 18:51 local
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x   2 root root 4096 Apr  9  2019 media
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x   2 root root 4096 Apr  9  2019 mnt
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x   2 root root 4096 Apr  9  2019 opt
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] dr-xr-
xr-x 103 root root    0 Jul 19 17:54 proc
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] dr-xr-
x-\-\-   2 root root 4096 Apr  9  2019 root
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x   2 root root 4096 Jun 13 18:52 run
```

```
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC]
 lrwxrwxrwx   1 root root    8 Jun 13 18:51 sbin -> usr/sbin
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x   2 root root 4096 Apr  9  2019 srv
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] dr-xr-
xr-x  13 root root    0 Jul 19 17:54 sys
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC]
 drwxrwxrwt   2 root root 4096 Jun 13 18:51 tmp
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x  13 root root 4096 Jun 13 18:51 usr
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x  18 root root 4096 Jun 13 18:52 var
.
.
.
[2022-01-01, 12:00:00 UTC] {{ecs.py:328}} INFO - ECS Task has been successfully
 executed
```

Usare dbt con Amazon MWAA

Questo argomento dimostra come utilizzare dbt e Postgres con Amazon MWAA. Nei passaggi seguenti, aggiungerai le dipendenze richieste al tuo `requirements.txt` e caricherai un progetto dbt di esempio nel bucket Amazon S3 del tuo ambiente. Quindi, utilizzerai un DAG di esempio per verificare che Amazon MWAA abbia installato le dipendenze e infine utilizzerai il `BashOperator` per eseguire il progetto dbt.

Argomenti

- Versione
- Prerequisiti
- Dipendenze
- Caricare un progetto dbt su Amazon S3
- Utilizzate un DAG per verificare l'installazione della dipendenza da dbt
- Usa un DAG per eseguire un progetto dbt

Versione

- Puoi usare l'esempio di codice in questa pagina con Apache Airflow v2 e versioni successive in Python 3.10.

Prerequisiti

Prima di poter completare i seguenti passaggi, avrai bisogno di quanto segue:

- Un ambiente Amazon MWAA che utilizza Apache Airflow v2.2.2. Questo esempio è stato scritto e testato con la versione 2.2.2. Potrebbe essere necessario modificare l'esempio per utilizzarlo con altre versioni di Apache Airflow.
- Un esempio di progetto dbt. Per iniziare a usare dbt con Amazon MWAA, puoi creare un fork e clonare il progetto dbt starter dal repository dbt-labs. GitHub

Dipendenze

Per usare Amazon MWAA con dbt, aggiungi il seguente script di avvio al tuo ambiente. Per ulteriori informazioni, consulta Usare uno script di avvio con Amazon MWAA.

```
#!/bin/bash

if [[ "${MWAA_AIRFLOW_COMPONENT}" != "worker" ]]
then
    exit 0
fi

echo "------------------------------"
echo "Installing virtual Python env"
echo "------------------------------"

pip3 install --upgrade pip

echo "Current Python version:"
python3 --version
echo "..."

sudo pip3 install --user virtualenv
sudo mkdir python3-virtualenv
cd python3-virtualenv
```

```
sudo python3 -m venv dbt-env
sudo chmod -R 777 *

echo "------------------------------"
echo "Activating venv in"
$DBT_ENV_PATH
echo "------------------------------"

source dbt-env/bin/activate
pip3 list

echo "------------------------------"
echo "Installing libraries..."
echo "------------------------------"

# do not use sudo, as it will install outside the venv
pip3 install dbt-redshift==1.6.1 dbt-postgres==1.6.1

echo "------------------------------"
echo "Venv libraries..."
echo "------------------------------"

pip3 list
dbt --version

echo "------------------------------"
echo "Deactivating venv..."
echo "------------------------------"

deactivate
```

Nelle sezioni seguenti, caricherai la tua directory di progetto dbt su Amazon S3 ed eseguirai un DAG che convalida se Amazon MWAA ha installato correttamente le dipendenze dbt richieste.

Caricare un progetto dbt su Amazon S3

Per poter utilizzare un progetto dbt con il tuo ambiente Amazon MWAA, puoi caricare l'intera directory del progetto nella cartella del tuo ambiente. `dags` Quando l'ambiente si aggiorna, Amazon MWAA scarica la directory dbt nella cartella locale. `usr/local/airflow/dags/`

Per caricare un progetto dbt su Amazon S3

1. Passa alla directory in cui hai clonato il progetto dbt starter.

2. Esegui il seguente AWS CLI comando Amazon S3 per copiare in modo ricorsivo il contenuto del progetto `dags` nella cartella del tuo ambiente utilizzando il parametro. `--recursive` Il comando crea una sottodirectory chiamata `dbt` che puoi usare per tutti i tuoi progetti dbt. Se la sottodirectory esiste già, i file di progetto vengono copiati nella directory esistente e non viene creata una nuova directory. Il comando crea anche una sottodirectory all'interno della `dbt` directory per questo specifico progetto iniziale.

   ```
   $ aws s3 cp dbt-starter-project s3://mwaa-bucket/dags/dbt/dbt-starter-project --
   recursive
   ```

   È possibile utilizzare nomi diversi per le sottodirectory del progetto per organizzare più progetti dbt all'interno della directory principale. `dbt`

Utilizzate un DAG per verificare l'installazione della dipendenza da dbt

Il seguente DAG utilizza un comando `BashOperator` e un comando bash per verificare se Amazon MWAA ha installato correttamente le dipendenze dbt specificate in. `requirements.txt`

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

with DAG(dag_id="dbt-installation-test", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
    cli_command = BashOperator(
        task_id="bash_command",
        bash_command="/usr/local/airflow/.local/bin/dbt --version"
    )
```

Effettua le seguenti operazioni per visualizzare i log delle attività e verificare che dbt e le sue dipendenze siano state installate.

1. Accedi alla console Amazon MWAA, quindi scegli Open Airflow UI dall'elenco degli ambienti disponibili.
2. Nell'interfaccia utente di Apache Airflow, individua il `dbt-installation-test` DAG dall'elenco, quindi scegli la data sotto la `Last Run` colonna per aprire l'ultima attività riuscita.
3. Utilizzando Graph View, scegliete l'`bash_command`attività per aprire i dettagli dell'istanza dell'operazione.

4. Scegliete Log per aprire i log delle attività, quindi verificate che i log elencino correttamente la versione dbt in cui è stata specificata. `requirements.txt`

Usa un DAG per eseguire un progetto dbt

Il seguente DAG utilizza `BashOperator` a per copiare i progetti dbt caricati su Amazon S3 dalla directory `usr/local/airflow/dags/` locale alla directory `/tmp` accessibile in scrittura, quindi esegue il progetto dbt. I comandi bash presuppongono un progetto dbt iniziale intitolato. `dbt-starter-project` Modifica il nome della directory in base al nome della directory del tuo progetto.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

import os

DAG_ID = os.path.basename(__file__).replace(".py", "")

# assumes all files are in a subfolder of DAGs called dbt

with DAG(dag_id=DAG_ID, schedule_interval=None, catchup=False, start_date=days_ago(1))
 as dag:
    cli_command = BashOperator(
        task_id="bash_command",
        bash_command="source /usr/local/airflow/python3-virtualenv/dbt-env/bin/
activate;\
cp -R /usr/local/airflow/dags/dbt /tmp;\
echo 'listing project files:';\
ls -R /tmp;\
cd /tmp/dbt/mwaa_dbt_test_project;\
/usr/local/airflow/python3-virtualenv/dbt-env/bin/dbt run --project-dir /tmp/dbt/
mwaa_dbt_test_project --profiles-dir ..;\
cat /tmp/dbt_logs/dbt.log;\
rm -rf /tmp/dbt/mwaa_dbt_test_project"
    )
```

AWS blog e tutorial

- Utilizzo di Amazon EKS e Amazon MWAA per Apache Airflow v2.x

Le migliori pratiche per Amazon Managed Workflows per Apache Airflow

Questa guida descrive le best practice che consigliamo per l'utilizzo di Amazon Managed Workflows for Apache Airflow.

Argomenti

- Ottimizzazione delle prestazioni per Apache Airflow su Amazon MWAA
- Gestione delle dipendenze Python in requirements.txt

Ottimizzazione delle prestazioni per Apache Airflow su Amazon MWAA

Questa pagina descrive le best practice che consigliamo per ottimizzare le prestazioni di un ambiente Amazon Managed Workflows for Apache Airflow utilizzando. Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA

Indice

- Aggiungere un'opzione di configurazione Apache Airflow
- Pianificatore Apache Airflow
  - Parametri
  - Limiti
- Cartelle DAG
  - Parametri
- File DAG
  - Parametri
- Attività
  - Parametri

Aggiungere un'opzione di configurazione Apache Airflow

La procedura seguente illustra i passaggi per aggiungere un'opzione di configurazione Airflow al proprio ambiente.

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.
4. Seleziona Successivo.
5. Scegli Aggiungi configurazione personalizzata nel riquadro delle opzioni di configurazione Airflow.
6. Scegli una configurazione dall'elenco a discesa e inserisci un valore, oppure digita una configurazione personalizzata e inserisci un valore.
7. Scegli Aggiungi configurazione personalizzata per ogni configurazione che desideri aggiungere.
8. Selezionare Salva.

Per ulteriori informazioni, consulta Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA.

Pianificatore Apache Airflow

Lo scheduler Apache Airflow è un componente fondamentale di Apache Airflow. Un problema con lo scheduler può impedire l'analisi dei DAG e la pianificazione delle attività. Per ulteriori informazioni sull'ottimizzazione dello scheduler di Apache Airflow, consulta Fine-tuning your scheduler performance nel sito Web di documentazione di Apache Airflow.

Parametri

Questa sezione descrive le opzioni di configurazione disponibili per lo scheduler Apache Airflow e i relativi casi d'uso.

Apache Airflow v2

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	celery.sync_parallelism	1	Il numero di processi utilizzati da Celery Executor per sincronizzare lo stato delle attività.	È possibile utilizzare questa opzione per prevenire i conflitti di coda limitando i processi utilizzati da Celery Executor. Per impostazione predefinita, viene impostato un valore per 1 prevenire errori nella consegna dei registri delle attività ai registri. CloudWatch Impostare il valore su `0` significa utilizzare il numero massimo di processi, ma potrebbe causare errori durante la consegna dei registri delle attività.

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	scheduler.processor_poll_interval	1	Il numero di secondi di attesa tra un'elaborazione consecutiva dei file DAG nel «ciclo» di Scheduler.	È possibile utilizzare questa opzione per liberare l'utilizzo della CPU sullo Scheduler aumentando il tempo di inattività dello Scheduler dopo aver completato il recupero dei risultati dell'analisi DAG, la ricerca e l'accodamento delle attività e l'esecuzione delle attività in coda nell'Executor. L'aumento di questo valore consuma il numero di thread di Scheduler eseguiti su un ambiente in Apache Airflow v2 e Apache Airflow v1. `scheduler.parsing_`

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
				`processes scheduler .max_thre ads` Ciò può ridurre la capacità degli Scheduler di analizzar e i DAG e aumentare il tempo necessari o per la visualizz azione dei DAG nel server Web.
v2	scheduler .max_dagr uns_to_cr eate_per_loop	10	Il numero massimo di DAG da creare per ogni «ciclo» di Scheduler. DagRuns	È possibile utilizzare questa opzione per liberare risorse per la pianificazione delle attività diminuend o il numero massimo di «loop» di DagRunsScheduler .

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	scheduler.parsing_processes	2	Il numero di thread che lo Scheduler può eseguire in parallelo ai DAG di pianificazione.	È possibile utilizzare questa opzione per liberare risorse diminuendo il numero di processi eseguiti dallo Scheduler in parallelo per analizzare i DAG. Si consiglia di mantenere basso questo numero se l'analisi del DAG influisce sulla pianificazione delle attività. È necessario specificare un valore inferiore al numero di vCPU nell'ambiente. Per ulteriori informazioni, consultare Limiti.

Limiti

Questa sezione descrive i limiti da considerare quando si modificano i parametri predefiniti per lo scheduler.

scheduler.parsing_processes, scheduler.max_threads

Sono consentiti due thread per vCPU per una classe di ambiente. Almeno un thread deve essere riservato allo scheduler per una classe di ambiente. Se si nota un ritardo nella pianificazione delle attività, potrebbe essere necessario aumentare la classe di ambiente. Ad esempio, un ambiente di grandi dimensioni ha un'istanza del contenitore Fargate a 4 vCpu come scheduler. Ciò significa che è disponibile un massimo di thread 7 totali da utilizzare per altri processi. Cioè, due thread hanno moltiplicato quattro vCPU, meno uno per lo scheduler stesso. Il valore specificato in `scheduler.max_threads` e non `scheduler.parsing_processes` deve superare il numero di thread disponibili per una classe di ambiente (come illustrato di seguito:

- mw1.small — Non deve superare il numero di 1 thread per altri processi. Il thread rimanente è riservato allo Scheduler.
- mw1.medium — Non deve superare i 3 thread per altri processi. Il thread rimanente è riservato allo Scheduler.
- mw1.large — Non deve superare i 7 thread per altri processi. Il thread rimanente è riservato allo Scheduler.

Cartelle DAG

Apache Airflow Scheduler esegue la scansione continua della cartella DAG nell'ambiente. Qualsiasi `plugins.zip` file contenuto o file Python (`.py`) contenente istruzioni di importazione «airflow». Tutti gli oggetti Python DAG risultanti vengono quindi inseriti in un file DagBagaffinché quel file venga elaborato dallo Scheduler per determinare quali attività, se del caso, devono essere pianificate. L'analisi dei file DAG avviene indipendentemente dal fatto che i file contengano oggetti DAG validi.

Parametri

Questa sezione descrive le opzioni di configurazione disponibili per la cartella DAG e i relativi casi d'uso.

Apache Airflow v2

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	scheduler.dag_dir_list_interval	300 secondi	Il numero di secondi in cui la cartella DaGs deve essere scansionata alla ricerca di nuovi file.	È possibile utilizzare questa opzione per liberare risorse aumentando il numero di secondi necessari per l'analisi della cartella DAg. Ti consigliamo di aumentare questo valore se riscontri lunghi tempi di analisi`total_parse_time` `metrics`, che potrebbero essere dovuti a un numero elevato di file nella cartella DAG.
v2	scheduler.min_file_process_interval	30 secondi	Il numero di secondi dopo i quali lo scheduler analizza un DAG e vengono riflessi gli	È possibile utilizzare questa opzione per liberare risorse aumentando il numero di secondi che

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
			aggiornamenti al DAG.	lo scheduler attende prima di analizzare un DAG. Ad esempio, se si specifica un valore di `30`, il file DAG viene analizzato ogni 30 secondi. Si consiglia di mantenere questo numero elevato per ridurre l'utilizzo della CPU nell'ambiente.

File DAG

Come parte del ciclo di pianificazione di Apache Airflow, i singoli file DAG vengono analizzati per estrarre oggetti DAG Python. In Apache Airflow v2 e versioni successive, lo scheduler analizza un numero massimo di processi di analisi contemporaneamente. Il numero di secondi specificato in `scheduler.min_file_process_interval` deve trascorrere prima che lo stesso file venga nuovamente analizzato.

Parametri

Questa sezione descrive le opzioni di configurazione disponibili per i file Apache Airflow DAG e i relativi casi d'uso.

Apache Airflow v2

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	core.dag_file_processor_timeout	50 secondi	Il numero di secondi prima del timeout del DagFilepr ocessore per l'elaborazione di un file DAG.	È possibile utilizzare questa opzione per liberare risorse aumentando il tempo necessari o prima del timeout del DagFilepr ocessore. Ti consigliamo di aumentare questo valore se riscontri dei timeout nei registri di elaborazi one DAG che impediscono il caricamento di DAG validi.
v2	core.dagbag_import_timeout	30 secondi	Il numero di secondi prima dell'impo rtazione di un file Python scade.	È possibile utilizzare questa opzione per liberare risorse aumentando il tempo necessari o prima che lo Scheduler scada durante l'importazione

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
				di un file Python per estrarre gli oggetti DAG. Questa opzione viene elaborata come parte del «ciclo» dello Scheduler e deve contenere un valore inferiore al valore specificato in. `core.dag_file_processor_timeout`

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	core.min_serialized_dag_update_interval	30	Il numero minimo di secondi dopo il quale vengono aggiornati i DAG serializzati nel database.	È possibile utilizzare questa opzione per liberare risorse aumentando il numero di secondi dopo i quali vengono aggiornati i DAG serializzati nel database. Si consiglia di aumentare questo valore se si dispone di un numero elevato di DAG o di DAG complessi. L'aumento di questo valore riduce il carico sullo Scheduler e sul database man mano che i DAG vengono serializzati.

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	core.min_serialized_dag_fetch_interval	10	Il numero di secondi in cui un DAG serializzato viene recuperato nuovamente dal database quando è già caricato in. DagBag	È possibile utilizzare questa opzione per liberare risorse aumentando il numero di secondi di recupero di un DAG serializzato. Il valore deve essere superiore al valore specificato in `core.min_serialized_dag_update_interval` per ridurre le velocità di «scrittura» del database. L'aumento di questo valore riduce il carico sul server Web e sul database man mano che i DAG vengono serializzati.

Attività

Lo scheduler e gli operatori di Apache Airflow sono entrambi coinvolti nelle attività di accodamento e disaccodamento. Lo scheduler prende le attività analizzate pronte per essere pianificate dallo stato Nessuno allo stato Pianificato. L'esecutore, anch'esso in esecuzione nel contenitore scheduler di Fargate, mette in coda tali attività e ne imposta lo stato su In coda. Quando i lavoratori hanno capacità, preleva l'operazione dalla coda e imposta lo stato su In esecuzione, che successivamente modifica lo stato in Operazione completata o Non riuscita a seconda che l'attività abbia esito positivo o negativo.

Parametri

Questa sezione descrive le opzioni di configurazione disponibili per le attività di Apache Airflow e i relativi casi d'uso.

Le opzioni di configurazione predefinite che Amazon MWAA sostituisce sono contrassegnate in rosso.

Apache Airflow v2

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	core.parallelism	10000	Il numero massimo di istanze di attività che possono avere lo stato «In esecuzione».	È possibile utilizzare questa opzione per liberare risorse aumentando il numero di istanze di attività che possono essere eseguite contemporaneamente. Il valore specificato deve essere il numero di Worker

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
				disponibili «volte» la densità delle attività Workers. Ti consigliamo di modificare questo valore solo quando vedi un gran numero di attività bloccate nello stato «In esecuzione» o «In coda».

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	core.dag_concurrency	10000	Il numero di istanze di attività che possono essere eseguite contempor aneamente per ogni DAG.	È possibile utilizzare questa opzione per liberare risorse aumentando il numero di istanze di attività consentite per l'esecuzione simultanea. Ad esempio, se si dispone di cento DAG con dieci attività parallele e si desidera che tutti i DAG vengano eseguiti contempor aneamente , è possibile calcolare il parallelismo massimo come il numero di Worker disponibili in «volte» la densità delle attività Workers`celery.worker_concurrency` ,

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
				diviso per il numero di DAG (ad esempio 100).
v2	core.execute_tasks_new_python_interpreter	`True`	Determina se Apache Airflow esegue le attività biforcando il processo principale o creando un nuovo processo Python.	Quando è impostato su`True`, Apache Airflow riconosce le modifiche apportate ai plugin come un nuovo processo Python creato per eseguire attività.
v2	celery.worker_concurrency	N/D	Amazon MWAA sostituisce l'installazione di base Airflow per questa opzione per scalare Workers come parte del componente di scalabilità automatica.	*Qualsiasi valore specificato per questa opzione viene ignorato.*

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
v2	celery.worker_autoscale	mw1.small - 5,0 mw1.medium - 10,0 mw1.large - 20,0 mw1.xlarge - 40,0 mw1.2xlarge - 80,0	La concorrenza tra compiti per i lavoratori.	È possibile utilizzare questa opzione per liberare risorse riducendo la *minimum* concomitanza delle *maximum* attività dei lavoratori. I lavoratori accettano fino alle attività maximum simultane e configura te, indipende ntemente dal fatto che vi siano risorse sufficienti per farlo. Se le attività sono pianificate senza risorse sufficienti, le attività falliscon o immediata mente. Si consiglia di modificare questo valore per le attività

Versione	Opzione di configurazione	Predefinito	Descrizione	Caso d'uso
				che richiedon o molte risorse riducendo i valori a un valore inferiore a quello predefinito per consentire una maggiore capacità per attività.

Gestione delle dipendenze Python in requirements.txt

Questa pagina descrive le best practice che consigliamo per installare e gestire le dipendenze Python in un `requirements.txt` file per un ambiente Amazon Managed Workflows for Apache Airflow.

Indice

- Test dei DAG utilizzando l'utilità CLI Amazon MWAA
- Installazione delle dipendenze Python utilizzando il formato file dei requisiti PyPi .org
  - Opzione uno: dipendenze Python dal Python Package Index
  - Opzione due: Python wheels (.whl)
    - Utilizzo del plugins.zip file su un bucket Amazon S3
    - Utilizzando un file WHL ospitato su un URL
    - Creazione di un file WHL da un DAG
  - Opzione tre: dipendenze Python ospitate su un repository privato PyPi conforme a /PEP-503
- Abilitazione dei log sulla console Amazon MWAA
- Visualizzazione dei log nella console Logs CloudWatch
- Visualizzazione degli errori nell'interfaccia utente di Apache Airflow
  - Accesso ad Apache Airflow
- Scenari di esempio requirements.txt

Test dei DAG utilizzando l'utilità CLI Amazon MWAA

- L'utilità CLI (Command Line Interface) replica localmente un ambiente Amazon Managed Workflows for Apache Airflow.
- La CLI crea localmente un'immagine del contenitore Docker simile a un'immagine di produzione Amazon MWAA. Ciò consente di eseguire un ambiente Apache Airflow locale per sviluppare e testare DAG, plug-in personalizzati e dipendenze prima della distribuzione su Amazon MWAA.
- Per eseguire la CLI, consulta aws-mwaa-local-runner su. GitHub

Installazione delle dipendenze Python utilizzando il formato file dei requisiti PyPi .org

La sezione seguente descrive i diversi modi per installare le dipendenze Python in base al PyPi formato.org Requirements File.

Opzione uno: dipendenze Python dal Python Package Index

La sezione seguente descrive come specificare le dipendenze Python dall'indice dei pacchetti Python in un file. `requirements.txt`

Apache Airflow v2

1. Esegui il test localmente. Aggiungi altre librerie in modo iterativo per trovare la giusta combinazione di pacchetti e le relative versioni, prima di creare un `requirements.txt` file. Per eseguire l'utilità CLI di Amazon MWAA, consulta aws-mwaa-local-runner su. GitHub
2. Consulta gli extra del pacchetto Apache Airflow. Per visualizzare un elenco dei pacchetti installati per Apache Airflow v2 su Amazon MWAA, consulta Amazon MWAA local runner sul sito Web. `requirements.txt` GitHub
3. Aggiungi una dichiarazione di vincoli. Aggiungi il file dei vincoli per il tuo ambiente Apache Airflow v2 nella parte superiore del file. `requirements.txt` I file di vincoli di Apache Airflow specificano le versioni del provider disponibili al momento del rilascio di Apache Airflow.

   A partire da Apache Airflow v2.7.2, il file dei requisiti deve includere una dichiarazione. `--constraint` Se non fornisci un vincolo, Amazon MWAA te ne specificherà uno per garantire che i pacchetti elencati nei tuoi requisiti siano compatibili con la versione di Apache Airflow che stai utilizzando.

Nell'esempio seguente, sostituisci *{environment-version}* con il numero di versione del tuo ambiente e *{Python-version} con la versione* di Python compatibile con il tuo ambiente.

Per informazioni sulla versione di Python compatibile con il tuo ambiente Apache Airflow, consulta Versioni di Apache Airflow.

```
--constraint "https://raw.githubusercontent.com/apache/airflow/
constraints-{Airflow-version}/constraints-{Python-version}.txt"
```

Se il file dei vincoli determina che il `xyz==1.0` pacchetto non è compatibile con altri pacchetti dell'ambiente, non `pip3 install` riuscirà a impedire l'installazione di librerie incompatibili nell'ambiente. Se l'installazione di qualsiasi pacchetto non riesce, è possibile visualizzare i log degli errori per ogni componente di Apache Airflow (lo scheduler, il worker e il server web) nel flusso di log corrispondente su Logs. CloudWatch Per ulteriori informazioni sui tipi di registro, vedere. the section called “Visualizzazione dei registri Airflow”

4. Pacchetti Apache Airflow. Aggiungi gli extra del pacchetto e la versione (). == Questo aiuta a evitare che pacchetti con lo stesso nome, ma con una versione diversa, vengano installati nell'ambiente.

```
apache-airflow[package-extra]==2.5.1
```

5. Librerie Python. Aggiungi il nome del pacchetto e la versione (==) nel tuo `requirements.txt` file. In questo modo si evita l'applicazione automatica di future interruzioni del PyPisito .org.

```
library == version
```

Example Boto3 e psycopg2-binary

Questo esempio viene fornito a scopo dimostrativo. Le librerie boto e psycopg2-binary sono incluse nell'installazione di base di Apache Airflow v2 e non devono essere specificate in un file. `requirements.txt`

```
boto3==1.17.54
boto==2.49.0
botocore==1.20.54
```

```
psycopg2-binary==2.8.6
```

Se viene specificato un pacchetto senza una versione, Amazon MWAA installa la versione più recente del pacchetto da .org. PyPi Questa versione può entrare in conflitto con altri pacchetti presenti nel tuo. `requirements.txt`

Apache Airflow v1

1. Esegui il test localmente. Aggiungi altre librerie in modo iterativo per trovare la giusta combinazione di pacchetti e le relative versioni, prima di creare un `requirements.txt` file. Per eseguire l'utilità CLI di Amazon MWAA, consulta aws-mwaa-local-runner su. GitHub
2. Consulta gli extra del pacchetto Airflow. Consulta l'elenco dei pacchetti disponibili per Apache Airflow v1.10.12 all'indirizzo https://raw.githubusercontent.com/apache/airflow/constraints-1.10.12/constraints-3.7.txt.
3. Aggiungere il file dei vincoli. Aggiungi il file dei vincoli per Apache Airflow v1.10.12 nella parte superiore del file. `requirements.txt` Se il file dei vincoli determina che il `xyz==1.0` pacchetto non è compatibile con altri pacchetti presenti nell'ambiente, non `pip3 install` riuscirà a impedire l'installazione di librerie incompatibili nell'ambiente.

   ```
   --constraint "https://raw.githubusercontent.com/apache/airflow/
   constraints-1.10.12/constraints-3.7.txt"
   ```
4. Pacchetti Apache Airflow v1.10.12. Aggiungi gli extra del pacchetto Airflow e la versione Apache Airflow v1.10.12 (). == Questo aiuta a prevenire l'installazione di pacchetti con lo stesso nome, ma con una versione diversa, nell'ambiente.

   ```
   apache-airflow[package]==1.10.12
   ```

   Example Secure Shell (SSH)

   Il seguente `requirements.txt` file di esempio installa SSH per Apache Airflow v1.10.12.

   ```
   apache-airflow[ssh]==1.10.12
   ```
5. Librerie Python. Aggiungi il nome del pacchetto e la versione (==) nel tuo `requirements.txt` file. In questo modo si evita l'applicazione automatica di future interruzioni del PyPisito .org.

```
library == version
```

Example Boto3

Il seguente `requirements.txt` file di esempio installa la libreria Boto3 per Apache Airflow v1.10.12.

```
boto3 == 1.17.4
```

Se viene specificato un pacchetto senza una versione, Amazon MWAA installa la versione più recente del pacchetto da .org. PyPi Questa versione può entrare in conflitto con altri pacchetti presenti nel tuo. `requirements.txt`

Opzione due: Python wheels (.whl)

Una ruota Python è un formato di pacchetto progettato per spedire librerie con artefatti compilati. I pacchetti wheel offrono diversi vantaggi come metodo per installare le dipendenze in Amazon MWAA:

- Installazione più rapida: i file WHL vengono copiati nel contenitore come singolo ZIP e quindi installati localmente, senza dover scaricare ciascuno di essi.
- Meno conflitti: è possibile determinare in anticipo la compatibilità delle versioni dei pacchetti. Di conseguenza, non è necessario elaborare in modo ricorsivo versioni compatibili. `pip`
- Maggiore resilienza: con le librerie ospitate esternamente, i requisiti a valle possono cambiare, con conseguente incompatibilità di versione tra i contenitori in un ambiente Amazon MWAA. Non dipendendo da una fonte esterna per le dipendenze, ogni contenitore ha le stesse librerie indipendentemente dal momento in cui viene creata l'istanza di ogni contenitore.

Consigliamo i seguenti metodi per installare le dipendenze Python da un archivio Python wheel () nel tuo. `.whl` `requirements.txt`

Metodi

- Utilizzo del plugins.zip file su un bucket Amazon S3
- Utilizzando un file WHL ospitato su un URL
- Creazione di un file WHL da un DAG

Utilizzo del **plugins.zip** file su un bucket Amazon S3

Lo scheduler, i worker e il server web di Apache Airflow (per Apache Airflow v2.2.2 e versioni successive) cercano plugin personalizzati durante l'avvio nel contenitore Fargate gestito per il AWS vostro ambiente all'indirizzo. `/usr/local/airflow/plugins/*` Questo processo inizia prima dell'avvio del servizio Amazon MWAA `pip3 install -r requirements.txt` for Python e del servizio Apache Airflow. Un `plugins.zip` file può essere utilizzato per tutti i file che non si desidera modificare continuamente durante l'esecuzione dell'ambiente o ai quali non si desidera concedere l'accesso agli utenti che scrivono DAG. Ad esempio, i file della ruota della libreria Python, i file PEM dei certificati e i file YAML di configurazione.

La sezione seguente descrive come installare una ruota contenuta nel `plugins.zip` file sul tuo bucket Amazon S3.

1. Scarica i file WHL necessari Puoi usarli `pip download`con il tuo local-runner esistente `requirements.txt` su Amazon MWAA o su un altro contenitore Amazon Linux 2 per risolvere e scaricare i file di Python wheel necessari.

   ```
   $ pip3 download -r "$AIRFLOW_HOME/dags/requirements.txt" -d "$AIRFLOW_HOME/plugins"
   $ cd "$AIRFLOW_HOME/plugins"
   $ zip "$AIRFLOW_HOME/plugins.zip" *
   ```

2. Specificate il percorso nel vostro. **`requirements.txt`** Specificate la directory dei plugins nella parte superiore del file requirements.txt utilizzando `--find-links`e istruite a `pip` non effettuare l'installazione da altre fonti utilizzando `--no-index`, come illustrato di seguito

   ```
   --find-links /usr/local/airflow/plugins
   --no-index
   ```

   Example ruota in requirements.txt

   L'esempio seguente presuppone che tu abbia caricato la ruota in un `plugins.zip` file nella radice del tuo bucket Amazon S3. Per esempio:

   ```
   --find-links /usr/local/airflow/plugins
   --no-index

   numpy
   ```

Amazon MWAA recupera la `numpy-1.20.1-cp37-cp37m-manylinux1_x86_64.whl` ruota dalla `plugins` cartella e la installa nel tuo ambiente.

Utilizzando un file WHL ospitato su un URL

La sezione seguente descrive come installare una ruota ospitata su un URL. L'URL deve essere accessibile pubblicamente o dall'interno del VPC Amazon personalizzato specificato per il tuo ambiente Amazon MWAA.

- Fornisci un URL. Fornisci l'URL a una ruota nel tuo`requirements.txt`.

  Example archivio wheel su un URL pubblico

  L'esempio seguente scarica una ruota da un sito pubblico.

  ```
  --find-links https://files.pythonhosted.org/packages/
  --no-index
  ```

  Amazon MWAA recupera la ruota dall'URL specificato e la installa nel tuo ambiente.

  > **Note**
  >
  > Gli URL non sono accessibili dai server Web privati che installano i requisiti in Amazon MWAA v2.2.2 e versioni successive.

Creazione di un file WHL da un DAG

Se disponi di un server Web privato che utilizza Apache Airflow v2.2.2 o versione successiva e non riesci a installare i requisiti perché il tuo ambiente non ha accesso a repository esterni, puoi utilizzare il seguente DAG per prendere i requisiti Amazon MWAA esistenti e impacchettarli su Amazon S3:

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

S3_BUCKET = 'my-s3-bucket'
S3_KEY = 'backup/plugins_whl.zip'
```

```
with DAG(dag_id="create_whl_file", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
    cli_command = BashOperator(
        task_id="bash_command",
        bash_command=f"mkdir /tmp/whls;pip3 download -r /usr/local/airflow/
requirements/requirements.txt -d /tmp/whls;zip -j /tmp/plugins.zip /tmp/whls/*;aws s3
 cp /tmp/plugins.zip s3://{S3_BUCKET}/{S3_KEY}"
    )
```

Dopo aver eseguito il DAG, usa questo nuovo file come Amazon MWAA`plugins.zip`, facoltativamente, impacchettato con altri plugin. Quindi, aggiorna il file preceduto da e senza aggiungere. `requirements.txt` `--find-links /usr/local/airflow/plugins` `--no-index` `--constraint`

Questo metodo consente di utilizzare le stesse librerie offline.

Opzione tre: dipendenze Python ospitate su un repository privato PyPi conforme a / PEP-503

La sezione seguente descrive come installare un extra di Apache Airflow ospitato su un URL privato con autenticazione.

1. Aggiungi nome utente e password come opzioni di configurazione di Apache Airflow. Per esempio:
   - `foo.user` : *YOUR_USER_NAME*
   - `foo.pass` : *YOUR_PASSWORD*
2. Crea il tuo file. `requirements.txt` Sostituisci i segnaposto nell'esempio seguente con il tuo URL privato e il nome utente e la password che hai aggiunto come opzioni di configurazione di Apache Airflow. Per esempio:

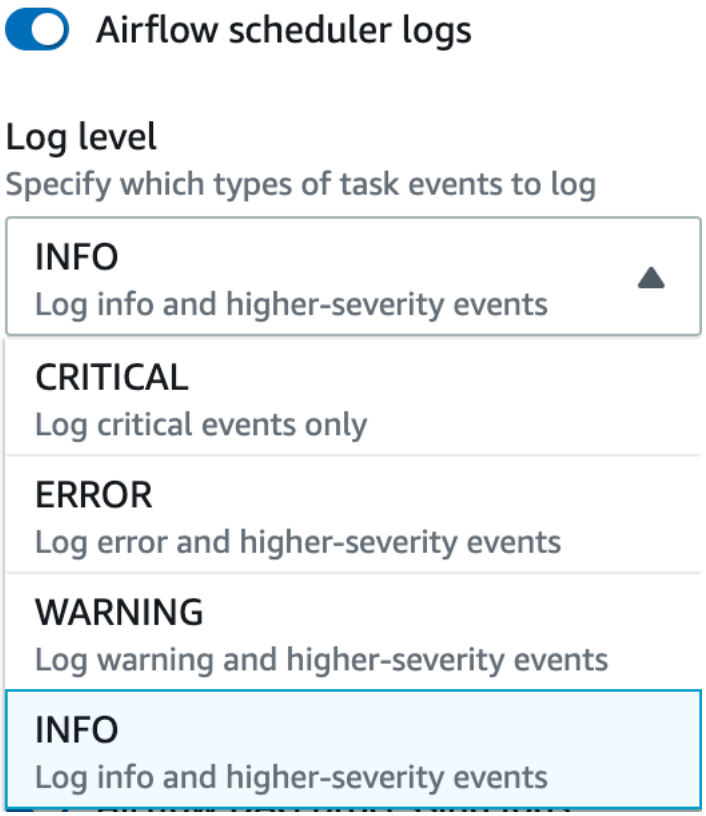
   ```
   --index-url https://${AIRFLOW__FOO__USER}:${AIRFLOW__FOO__PASS}@my.privatepypi.com
   ```
3. Aggiungi eventuali librerie aggiuntive al tuo file. `requirements.txt` Per esempio:

   ```
   --index-url https://${AIRFLOW__FOO__USER}:${AIRFLOW__FOO__PASS}@my.privatepypi.com
   my-private-package==1.2.3
   ```

Abilitazione dei log sulla console Amazon MWAA

Il ruolo di esecuzione per il tuo ambiente Amazon MWAA richiede l'autorizzazione per inviare log a Logs. CloudWatch Per aggiornare le autorizzazioni di un ruolo di esecuzione, consulta. Ruolo di esecuzione di Amazon MWAA

È possibile abilitare i log di Apache Airflow a livello`INFO`,`WARNING`, `ERROR` o. `CRITICAL` Quando scegli un livello di log, Amazon MWAA invia i log per quel livello e tutti i livelli di gravità più elevati. Ad esempio, se abiliti i log a `INFO` livello, Amazon MWAA invia `INFO` log e `WARNINGERROR`, e livelli di `CRITICAL` log a Logs. CloudWatch Ti consigliamo di abilitare i log di Apache Airflow a *`INFO`* livello di Scheduler per visualizzare i log ricevuti per. `requirements.txt`



Visualizzazione dei log nella console Logs CloudWatch

È possibile visualizzare i log di Apache Airflow per Scheduler, pianificando i flussi di lavoro e analizzando la cartella. `dags` I passaggi seguenti descrivono come aprire il gruppo di log per Scheduler sulla console Amazon MWAA e visualizzare i log di Apache Airflow sulla console Logs. CloudWatch

Per visualizzare i log di un **`requirements.txt`**

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli il gruppo di log dello scheduler Airflow nel riquadro Monitoraggio.
4. Scegli il `requirements_install_ip` log in Log Streams.
5. Dovresti vedere l'elenco dei pacchetti che sono stati installati nell'ambiente all'indirizzo`/usr/local/airflow/.local/bin`. Per esempio:

```
Collecting appdirs==1.4.4 (from -r /usr/local/airflow/.local/bin (line 1))
Downloading https://files.pythonhosted.org/
packages/3b/00/2344469e2084fb28kjdsfiuyweb47389789vxbmnbjhsdgf5463acd6cf5e3db69324/
appdirs-1.4.4-py2.py3-none-any.whl
Collecting astroid==2.4.2 (from -r /usr/local/airflow/.local/bin (line 2))
```

6. Controlla l'elenco dei pacchetti e verifica se qualcuno di questi ha riscontrato un errore durante l'installazione. Se qualcosa è andato storto, potresti visualizzare un errore simile al seguente:

```
2021-03-05T14:34:42.731-07:00
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
```

Visualizzazione degli errori nell'interfaccia utente di Apache Airflow

Potresti anche voler controllare l'interfaccia utente di Apache Airflow per identificare se un errore può essere correlato a un altro problema. L'errore più comune che potresti riscontrare con Apache Airflow su Amazon MWAA è:

```
Broken DAG: No module named x
```

Se vedi questo errore nell'interfaccia utente di Apache Airflow, è probabile che nel file manchi una dipendenza richiesta. `requirements.txt`

Accesso ad Apache Airflow

Hai bisogno Politica di accesso all'interfaccia utente di Apache Airflow: AmazonMWAA Access WebServer delle autorizzazioni per il tuo AWS account in AWS Identity and Access Management (IAM) per visualizzare l'interfaccia utente di Apache Airflow.

Per accedere all'interfaccia utente di Apache Airflow

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Open Airflow UI.

Scenari di esempio **`requirements.txt`**

Puoi mescolare e abbinare diversi formati nel tuo`requirements.txt`. L'esempio seguente utilizza una combinazione dei diversi modi per installare gli extra.

Example Extra su PyPi .org e un URL pubblico

È necessario utilizzare l'`--index-url`opzione quando si specificano i pacchetti da PyPi .org, oltre ai pacchetti su un URL pubblico, come gli URL di repository personalizzati conformi a PEP 503.

```
aws-batch == 0.6
phoenix-letter >= 0.3

--index-url http://dist.repoze.org/zope2/2.10/simple
    zopelib
```

Monitoraggio e parametri per Amazon Managed Workflows for Apache Airflow

Il monitoraggio è una parte importante per mantenere l'affidabilità, la disponibilità e le prestazioni di Amazon Managed Workflows for Apache Airflow e della tua soluzione. AWS Ti consigliamo di 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 uno. Questo argomento descrive le risorse disponibili AWS per monitorare l'ambiente Amazon MWAA e rispondere a potenziali eventi.

> **Note**
>
> Le metriche e la registrazione di Apache Airflow sono soggette ai prezzi standard di Amazon. CloudWatch

Per ulteriori informazioni sul monitoraggio di Apache Airflow, consulta Logging & Monitoring nel sito Web della documentazione di Apache Airflow.

Sections

- Panoramica del monitoraggio su Amazon MWAA
- Visualizzazione dei registri di controllo AWS CloudTrail
- Visualizzazione dei log di Airflow in Amazon CloudWatch
- Pannelli di controllo e allarmi su Amazon MWAA
- Metriche di ambiente Apache Airflow v2 in CloudWatch
- Parametri di container, code e database per Amazon MWAA

Panoramica del monitoraggio su Amazon MWAA

Questa pagina descrive i AWS servizi utilizzati per monitorare un ambiente Amazon Managed Workflows for Apache Airflow.

Indice

- CloudWatch Panoramica di Amazon

- AWS CloudTrail panoramica

CloudWatch Panoramica di Amazon

CloudWatch è un archivio di metriche per AWS i servizi che consente di recuperare statistiche in base alle metriche e alle dimensioni pubblicate da un servizio. Puoi utilizzare queste metriche per configurare allarmi, calcolare statistiche e quindi presentare i dati in una dashboard che ti aiuta a valutare lo stato del tuo ambiente nella console Amazon CloudWatch .

Apache Airflow è già configurato per inviare ad Amazon i parametri StatSD per un ambiente Amazon Managed Workflows for Apache Airflow. CloudWatch

Per ulteriori informazioni, consulta What is Amazon CloudWatch? .

AWS CloudTrail panoramica

CloudTrail è un servizio di audit che fornisce una registrazione delle azioni intraprese da un utente, un ruolo o un AWS servizio in Amazon MWAA. Utilizzando le informazioni raccolte da CloudTrail, puoi determinare la richiesta effettuata ad Amazon MWAA, l'indirizzo IP da cui è stata effettuata, chi ha effettuato la richiesta, quando è stata effettuata e ulteriori dettagli disponibili nei log di controllo.

Per ulteriori informazioni, consulta What is? AWS CloudTrail .

Visualizzazione dei registri di controllo AWS CloudTrail

AWS CloudTrail è abilitato sul tuo AWS account al momento della creazione. CloudTrail registra l'attività svolta da un'entità o da un AWS servizio IAM, come Amazon Managed Workflows for Apache Airflow, che viene registrata come evento. CloudTrail Puoi visualizzare, cercare e scaricare la cronologia degli ultimi 90 giorni degli eventi nella console. CloudTrail CloudTrail acquisisce tutti gli eventi sulla console Amazon MWAA e tutte le chiamate alle API Amazon MWAA. Non acquisisce azioni di sola lettura, ad esempio o l'azione. `GetEnvironment` `PublishMetrics` Questa pagina descrive come monitorare gli eventi per Amazon MWAA. CloudTrail

Indice

- Creare un percorso in CloudTrail
- Visualizzazione degli eventi con la cronologia CloudTrail degli eventi
- Esempio di percorso per CreateEnvironment
- Fasi successive

Creare un percorso in CloudTrail

Devi creare un percorso per visualizzare un record continuo di eventi nel tuo AWS account, inclusi gli eventi per Amazon MWAA. Un trail consente di CloudTrail inviare file di log a un bucket Amazon S3. Se non crei un percorso, puoi comunque visualizzare la cronologia degli eventi disponibili nella CloudTrail console. Ad esempio, utilizzando le informazioni raccolte da CloudTrail, puoi determinare la richiesta che è stata fatta ad Amazon MWAA, 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, consulta la sezione Creare un percorso per il tuo AWS account.

Visualizzazione degli eventi con la cronologia CloudTrail degli eventi

Puoi risolvere gli incidenti operativi e di sicurezza degli ultimi 90 giorni nella CloudTrail console visualizzando la cronologia degli eventi. Ad esempio, puoi visualizzare gli eventi relativi alla creazione, alla modifica o all'eliminazione di risorse (come utenti IAM o altre AWS risorse) nel tuo AWS account in base alla regione. Per ulteriori informazioni, consulta la sezione Visualizzazione degli eventi con CloudTrail la cronologia degli eventi.

1. Aprire la console CloudTrail.
2. Scegli Cronologia degli eventi.
3. Seleziona gli eventi che desideri visualizzare, quindi scegli Confronta i dettagli dell'evento.

Esempio di percorso per **`CreateEnvironment`**

Un trail è una configurazione che consente la distribuzione di eventi come i file di log in un bucket Amazon S3 specificato.

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, ad esempio la data e l'ora dell'azione o i parametri della richiesta. CloudTrail i file di registro non sono una traccia ordinata dello stack delle chiamate API pubbliche e non vengono visualizzati in un ordine specifico. L'esempio seguente è una voce di registro per l'`CreateEnvironment`azione che viene negata a causa della mancanza di autorizzazioni. I valori inseriti `AirflowConfigurationOptions` sono stati oscurati per motivi di privacy.

```
{
    "eventVersion": "1.05",
    "userIdentity": {
```

```
        "type": "AssumedRole",
        "principalId": "00123456ABC7DEF8HIJK",
        "arn": "arn:aws:sts::012345678901:assumed-role/root/myuser",
        "accountId": "012345678901",
        "accessKeyId": "",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "00123456ABC7DEF8HIJK",
                "arn": "arn:aws:iam::012345678901:role/user",
                "accountId": "012345678901",
                "userName": "user"
            },
            "webIdFederationData": {},
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2020-10-07T15:51:52Z"
            }
        }
    },
    "eventTime": "2020-10-07T15:52:58Z",
    "eventSource": "airflow.amazonaws.com",
    "eventName": "CreateEnvironment",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.251.233.178",
    "userAgent": "PostmanRuntime/7.26.5",
    "errorCode": "AccessDenied",
    "requestParameters": {
        "SourceBucketArn": "arn:aws:s3:::my-bucket",
        "ExecutionRoleArn": "arn:aws:iam::012345678901:role/AirflowTaskRole",
        "AirflowConfigurationOptions": "***",
        "DagS3Path": "sample_dag.py",
        "NetworkConfiguration": {
            "SecurityGroupIds": [
                "sg-01234567890123456"
            ],
            "SubnetIds": [
                "subnet-01234567890123456",
                "subnet-65432112345665431"
            ]
        },
        "Name": "test-cloudtrail"
    },
    "responseElements": {
```

```
        "message": "Access denied."
    },
    "requestID": "RequestID",
    "eventID": "EventID",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "recipientAccountId": "012345678901"
}
```

Fasi successive

- Scopri come configurare altri AWS servizi per i dati sugli eventi raccolti nei CloudTrail registri in Servizi e CloudTrail integrazioni supportati.
- Scopri come ricevere una notifica quando vengono CloudTrail pubblicati nuovi file di log in un bucket Amazon S3 in Configurazione delle notifiche di Amazon SNS per. CloudTrail

Visualizzazione dei log di Airflow in Amazon CloudWatch

Amazon MWAA può inviare i log di Apache Airflow ad Amazon. CloudWatch Puoi visualizzare i log di più ambienti da un'unica posizione per identificare facilmente i ritardi nelle attività o gli errori del flusso di lavoro di Apache Airflow senza la necessità di strumenti aggiuntivi di terze parti. I log di Apache Airflow devono essere abilitati sulla console Amazon Managed Workflows for Apache Airflow per visualizzare l'elaborazione, le attività, il server Web e gli accessi di Apache Airflow DAG. CloudWatch

Indice

- Prezzi
- Prima di iniziare
- Tipi di log
- Abilitazione dei log di Apache Airflow
- Visualizzazione dei log di Apache Airflow
- Esempi di log dello scheduler
- Fasi successive

Prezzi

- Si applicano le tariffe Standard Logs. CloudWatch Per ulteriori informazioni, consulta la pagina CloudWatch dei prezzi.

Prima di iniziare

- È necessario disporre di un ruolo in grado di visualizzare i log in CloudWatch. Per ulteriori informazioni, consulta Accesso a un ambiente Amazon MWAA.

Tipi di log

Amazon MWAA crea un gruppo di log per ogni opzione di registrazione Airflow abilitata e invia i log ai gruppi Logs associati a un ambiente. CloudWatch I gruppi di log sono denominati nel seguente formato:. `YourEnvironmentName-`**`LogType`** Per esempio, se il nome del tuo ambiente è`Airflow-v202-Public`, i log delle attività di Apache Airflow vengono inviati a. `Airflow-v202-Public-`**`Task`**

Tipo di log	Descrizione
`YourEnvironmentName-` **`DAGProcesing`**	I log del gestore del processore DAG (la parte dello scheduler che elabora i file DAG).
`YourEnvironmentName-` **`Scheduler`**	I log generati dallo scheduler Airflow.
`YourEnvironmentName-` **`Task`**	I log delle attività generati da un DAG.
`YourEnvironmentName-` **`WebServer`**	I log generati dall'interfaccia web Airflow.
`YourEnvironmentName-` **`Worker`**	I log generati come parte del flusso di lavoro e dell'esecuzione del DAG.

Abilitazione dei log di Apache Airflow

È possibile abilitare i log di Apache Airflow a livello,, o. `INFO` `WARNING` `ERROR` `CRITICAL` Quando scegli un livello di log, Amazon MWAA invia i log per quel livello e tutti i livelli di gravità più elevati. Ad

esempio, se abiliti i log a `INFO` livello, Amazon MWAA invia `INFO` log e `WARNINGERROR`, e livelli di `CRITICAL` log a Logs. CloudWatch

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli Modifica.
4. Seleziona Successivo.
5. Scegliete una o più delle seguenti opzioni di registrazione:
    a. Scegli il gruppo di log dello scheduler Airflow nel riquadro Monitoraggio.
    b. Scegli il gruppo di log del server web Airflow nel riquadro Monitoraggio.
    c. Scegli il gruppo di log di lavoro Airflow nel riquadro Monitoraggio.
    d. Scegli il gruppo di log di elaborazione Airflow DAG nel riquadro Monitoraggio.
    e. Scegli il gruppo di log delle attività Airflow nel riquadro Monitoraggio.
    f. Scegliete il livello di registrazione in Livello di registro.
6. Seleziona Successivo.
7. Selezionare Salva.

Visualizzazione dei log di Apache Airflow

La sezione seguente descrive come visualizzare i log di Apache Airflow nella console. CloudWatch

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli un gruppo di log nel riquadro Monitoraggio.
4. Scegli un log in Log stream.

Esempi di log dello scheduler

È possibile visualizzare i log di Apache Airflow per Scheduler, pianificando i flussi di lavoro e analizzando la cartella. `dags` I passaggi seguenti descrivono come aprire il gruppo di log per Scheduler sulla console Amazon MWAA e visualizzare i log di Apache Airflow sulla console Logs. CloudWatch

Per visualizzare i log di un **`requirements.txt`**

1. Apri la pagina Ambienti sulla console Amazon MWAA.
2. Scegli un ambiente.
3. Scegli il gruppo di log dello scheduler Airflow nel riquadro Monitoraggio.
4. Scegli il `requirements_install_ip` log in Log Streams.
5. Dovresti vedere l'elenco dei pacchetti che sono stati installati nell'ambiente all'indirizzo`/usr/local/airflow/.local/bin`. Per esempio:

```
Collecting appdirs==1.4.4 (from -r /usr/local/airflow/.local/bin (line 1))
Downloading https://files.pythonhosted.org/
packages/3b/00/2344469e2084fb28kjdsfiuyweb47389789vxbmnbjhsdgf5463acd6cf5e3db69324/
appdirs-1.4.4-py2.py3-none-any.whl
Collecting astroid==2.4.2 (from -r /usr/local/airflow/.local/bin (line 2))
```

6. Controlla l'elenco dei pacchetti e verifica se qualcuno di questi ha riscontrato un errore durante l'installazione. Se qualcosa è andato storto, potresti visualizzare un errore simile al seguente:

```
2021-03-05T14:34:42.731-07:00
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
```

Fasi successive

- Scopri come configurare un CloudWatch allarme in Uso degli CloudWatch allarmi Amazon.
- Scopri come creare una CloudWatch dashboard in Utilizzo delle CloudWatch dashboard.

Pannelli di controllo e allarmi su Amazon MWAA

Puoi creare una dashboard personalizzata in Amazon CloudWatch e aggiungere allarmi per una particolare metrica per monitorare lo stato di salute di un ambiente Amazon Managed Workflows for Apache Airflow. Quando un allarme è su un pannello di controllo, diventa rosso quando è attivo, facilitando il monitoraggio proattivo `ALARM` dello stato di un ambiente Amazon MWAA.

Apache Airflow espone i parametri per una serie di processi, tra cui il numero di processi DAG, la dimensione del bagaglio DAG, le attività attualmente in esecuzione, le attività non riuscite e i successi. Quando crei un ambiente, Airflow è configurato per inviare automaticamente i parametri per un ambiente Amazon MWAA a. CloudWatch Questa pagina descrive come creare un pannello di controllo dello stato di salute per le metriche Airflow in CloudWatch un ambiente Amazon MWAA.

Indice

- Metriche
- Panoramica degli stati di allarme
- Esempi di dashboard e allarmi personalizzati
  - Informazioni su queste metriche
  - Informazioni sulla dashboard
  - Utilizzo dei tutorial AWS
  - Usando AWS CloudFormation
- Eliminazione di metriche e dashboard
- Fasi successive

Metriche

Puoi creare un pannello di controllo e un allarme personalizzati per qualsiasi metrica disponibile per la tua versione di Apache Airflow. Ogni metrica corrisponde a un indicatore di prestazioni chiave (KPI) di Apache Airflow. Per visualizzare un elenco di metriche, consulta:

- Metriche di ambiente Apache Airflow v2 in CloudWatch

Panoramica degli stati di allarme

Un allarme di parametri può trovarsi nei possibili stati elencati di seguito:

- `OK` - Il parametro o espressione rientra nella soglia definita.
- `ALARM` - Il parametro o espressione non rientra nella soglia definita.
- `INSUFFICIENT_DATA` - L'allarme è stato appena attivato, il parametro non è disponibile o la quantità di dati non è sufficiente affinché il parametro determini lo stato dell'allarme.

Esempi di dashboard e allarmi personalizzati

Puoi creare una dashboard di monitoraggio personalizzata che mostri grafici di parametri selezionati per il tuo ambiente Amazon MWAA.

Informazioni su queste metriche

L'elenco seguente descrive ciascuna delle metriche create nella dashboard personalizzata dalle definizioni del tutorial e dei modelli in questa sezione.

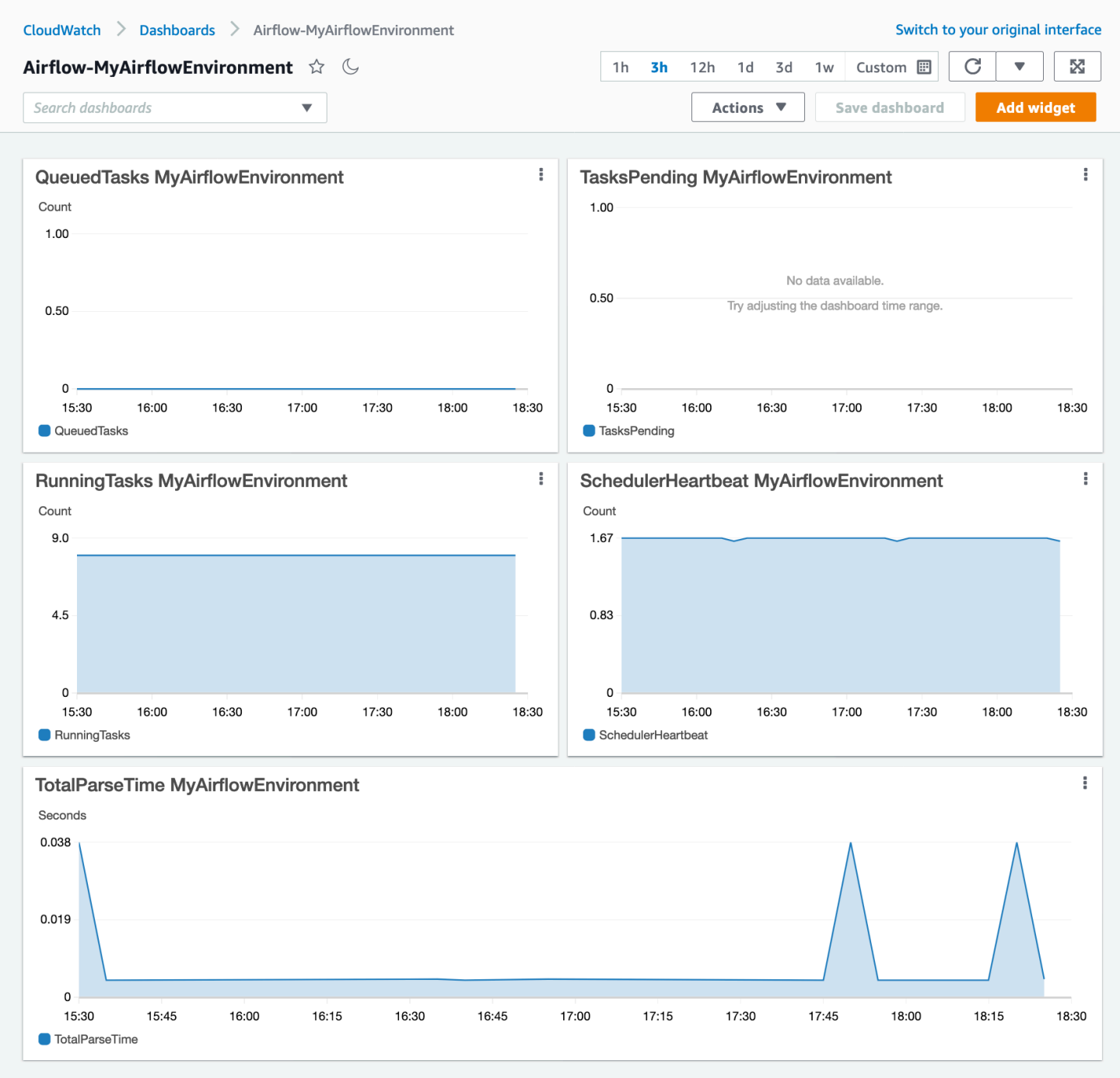
- QueuedTasks- Il numero di attività con stato in coda. Corrisponde alla metrica `executor.queued_tasks` Apache Airflow.
- TasksPending- Il numero di attività in sospeso nell'esecutore. Corrisponde alla metrica `scheduler.tasks.pending` Apache Airflow.

> **Note**
>
> Non si applica ad Apache Airflow v2.2 e versioni successive.

- RunningTasks- Il numero di attività in esecuzione in Executor. Corrisponde alla metrica `executor.running_tasks` Apache Airflow.
- SchedulerHeartbeat- Il numero di check-in che Apache Airflow esegue sul job di pianificazione. Corrisponde alle metriche di Apache `scheduler_heartbeat` Airflow.
- TotalParseTempo: il numero di secondi necessari per scansionare e importare tutti i file DAG una volta. Corrisponde alla metrica `dag_processing.total_parse_time` Apache Airflow.

Informazioni sulla dashboard

L'immagine seguente mostra la dashboard di monitoraggio creata dal tutorial e dalla definizione del modello in questa sezione.



Utilizzo dei tutorial AWS

Puoi utilizzare il seguente AWS tutorial per creare automaticamente un pannello di controllo dello stato di salute per qualsiasi ambiente Amazon MWAA attualmente distribuito. Inoltre, crea CloudWatch allarmi per lavoratori non sani e guasti del battito cardiaco degli scheduler in tutti gli ambienti Amazon MWAA.

- CloudWatch Automazione del pannello di controllo per Amazon MWAA

Usando AWS CloudFormation

Puoi utilizzare la definizione del AWS CloudFormation modello in questa sezione per creare una dashboard di monitoraggio in CloudWatch, quindi aggiungere allarmi sulla CloudWatch console per ricevere notifiche quando una metrica supera una determinata soglia. Per creare lo stack utilizzando questa definizione di modello, consulta Creazione di uno stack sulla console. AWS CloudFormation Per aggiungere un allarme alla dashboard, vedi Uso degli allarmi.

```
AWSTemplateFormatVersion: "2010-09-09"
Description: Creates MWAA Cloudwatch Dashboard
Parameters:
  DashboardName:
    Description: Enter the name of the CloudWatch Dashboard
    Type: String
  EnvironmentName:
    Description: Enter the name of the MWAA Environment
    Type: String
Resources:
  BasicDashboard:
    Type: AWS::CloudWatch::Dashboard
    Properties:
      DashboardName: !Ref DashboardName
      DashboardBody:
        Fn::Sub: '{
              "widgets": [
                  {
                      "type": "metric",
                      "x": 0,
                      "y": 0,
                      "width": 12,
                      "height": 6,
                      "properties": {
                          "view": "timeSeries",
                          "stacked": true,
                          "metrics": [
                              [
                                  "AmazonMWAA",
                                  "QueuedTasks",
                                  "Function",
                                  "Executor",
```

```
                        "Environment",
                        "${EnvironmentName}"
                    ]
                ],
                "region": "${AWS::Region}",
                "title": "QueuedTasks ${EnvironmentName}",
                "period": 300
            }
        },
        {
            "type": "metric",
            "x": 0,
            "y": 6,
            "width": 12,
            "height": 6,
            "properties": {
                "view": "timeSeries",
                "stacked": true,
                "metrics": [
                    [
                        "AmazonMWAA",
                        "RunningTasks",
                        "Function",
                        "Executor",
                        "Environment",
                        "${EnvironmentName}"
                    ]
                ],
                "region": "${AWS::Region}",
                "title": "RunningTasks ${EnvironmentName}",
                "period": 300
            }
        },
        {
            "type": "metric",
            "x": 12,
            "y": 6,
            "width": 12,
            "height": 6,
            "properties": {
                "view": "timeSeries",
                "stacked": true,
                "metrics": [
                    [
```

```
                        "AmazonMWAA",
                        "SchedulerHeartbeat",
                        "Function",
                        "Scheduler",
                        "Environment",
                        "${EnvironmentName}"
                    ]
                ],
                "region": "${AWS::Region}",
                "title": "SchedulerHeartbeat ${EnvironmentName}",
                "period": 300
            }
        },
        {
            "type": "metric",
            "x": 12,
            "y": 0,
            "width": 12,
            "height": 6,
            "properties": {
                "view": "timeSeries",
                "stacked": true,
                "metrics": [
                    [
                        "AmazonMWAA",
                        "TasksPending",
                        "Function",
                        "Scheduler",
                        "Environment",
                        "${EnvironmentName}"
                    ]
                ],
                "region": "${AWS::Region}",
                "title": "TasksPending ${EnvironmentName}",
                "period": 300
            }
        },
        {
            "type": "metric",
            "x": 0,
            "y": 12,
            "width": 24,
            "height": 6,
            "properties": {
```

```
                    "view": "timeSeries",
                    "stacked": true,
                    "region": "${AWS::Region}",
                    "metrics": [
                        [
                            "AmazonMWAA",
                            "TotalParseTime",
                            "Function",
                            "DAG Processing",
                            "Environment",
                            "${EnvironmentName}"
                        ]
                    ],
                    "title": "TotalParseTime  ${EnvironmentName}",
                    "period": 300
                }
            }
        ]
    }'
```

Eliminazione di metriche e dashboard

Se elimini un ambiente Amazon MWAA, viene eliminata anche la dashboard corrispondente. CloudWatch le metriche vengono archiviate per quindici (15) mesi e non possono essere eliminate. La CloudWatch console limita la ricerca delle metriche a due (2) settimane dall'ultima acquisizione di una metrica per garantire che vengano visualizzate le istanze più aggiornate per il tuo ambiente Amazon MWAA. Per ulteriori informazioni, consulta le CloudWatch domande frequenti su Amazon.

Fasi successive

- Scopri come creare un DAG che interroga il database di metadati PostgreSQL di Amazon Aurora per il tuo ambiente e pubblica metriche personalizzate su in. CloudWatch Utilizzo di un DAG per scrivere metriche personalizzate inCloudWatch

Metriche di ambiente Apache Airflow v2 in CloudWatch

Apache Airflow v2 è già configurato per raccogliere e inviare ad Amazon i parametri StatSD per un ambiente Amazon Managed Workflows for Apache Airflow. CloudWatch L'elenco completo delle metriche inviate da Apache Airflow è disponibile nella pagina Metrics della guida di riferimento di

Apache Airflow. Questa pagina descrive le metriche di Apache Airflow disponibili nella CloudWatch console e come accedervi. CloudWatch

Indice

- Termini
- Dimensioni
- Accesso alle metriche nella console CloudWatch
- Le metriche di Apache Airflow sono disponibili in CloudWatch
  - Contatori Apache Airflow
  - Misuratori del flusso d'aria Apache
  - Timer Apache Airflow
- Scelta delle metriche da segnalare
- Fasi successive

Termini

Spazio dei nomi

Un namespace è un contenitore per le metriche di un servizio. CloudWatch AWS Per Amazon MWAA, lo spazio dei nomi è AmazonMWAA.

CloudWatch metriche

Una CloudWatch metrica rappresenta un insieme ordinato nel tempo di punti dati specifici per. CloudWatch

Metriche di Apache Airflow

Le metriche specifiche di Apache Airflow.

Dimensione

Una dimensione è una coppia nome-valore che fa parte dell'identità di un parametro.

Unità

Una statistica ha un'unità di misura. Per Amazon MWAA, le unità includono Count, Secondi e Millisecondi. Per Amazon MWAA, le unità vengono impostate in base alle unità nelle metriche Airflow originali.

Dimensioni

Questa sezione descrive il raggruppamento delle CloudWatch dimensioni per le metriche di Apache Airflow in. CloudWatch

Dimensione	Descrizione
DAG	Indica un nome DAG specifico di Apache Airflow.
Nome del file DAG	Indica un nome di file Apache Airflow DAG specifico.
Funzione	Questa dimensione viene utilizzata per migliorare il raggruppamento delle metriche in. CloudWatch
Processo	Indica un Apache Airflow Job eseguito dallo Scheduler. Job ha sempre il valore di Job.
Operatore	Indica un operatore Apache Airflow specifico.
Pool	Indica un pool di lavoratori Apache Airflow specifico.
Attività	Indica un'attività specifica di Apache Airflow.
HostName	Indica il nome host per uno specifico processo di Apache Airflow in esecuzione.

Accesso alle metriche nella console CloudWatch

Questa sezione descrive come accedere alle metriche delle prestazioni CloudWatch per un DAG specifico.

Per visualizzare le metriche delle prestazioni per una dimensione

1. Apri la pagina Metriche sulla console. CloudWatch
2. Usa il selettore AWS della regione per selezionare la tua regione.
3. Scegli il namespace AmazonMWAA.
4. Nella scheda Tutte le metriche, seleziona una dimensione. Ad esempio, DAG, Environment.
5. Scegli una CloudWatch metrica per una dimensione. Ad esempio, TaskInstanceSuccessi o TaskInstance Durata. Scegli Grafica tutti i risultati della ricerca.
6. Scegli la scheda Metriche grafiche per visualizzare le statistiche sulle prestazioni per le metriche di Apache Airflow, come DAG, Environment, Task.

Le metriche di Apache Airflow sono disponibili in CloudWatch

Questa sezione descrive le metriche e le dimensioni di Apache Airflow inviate a. CloudWatch

Contatori Apache Airflow

Le metriche di Apache Airflow in questa sezione contengono dati su Apache Airflow Counters.

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
SLA è mancato **Note** Disponibile per Apache Airflow v2.4.3 e versioni successive.	sla_missed	Conteggio	Funzione, Scheduler	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
Callback SLA non riuscito **Note** Disponibile per Apache Airflow v2.4.3 e versioni successive.	sla_callback_notification_failure	Conteggio	Funzione, Scheduler	
Aggiornamenti **Note** Disponibile per Apache Airflow v2.6.3 e versioni successive.	dataset.updates	Conteggio	Funzione, Scheduler	
Orfano **Note** Disponibile per Apache Airflow v2.6.3 e versioni successive.	dataset.orphaned	Conteggio	Funzione, Scheduler	
FailedCeleryTaskExecution **Note** Disponibile per Apache Airflow v2.4.3 e versioni successive.	celery.execute_command.failure	Conteggio	Funzione, Celery	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
FilePathQueueUpdateConta **Note** Disponibile per Apache Airflow v2.6.3 e versioni successive.	dag_processing.file_path_queue_update_count	Conteggio	Funzione, Scheduler	
CriticalSectionOccupato	scheduler.critical_section_busy	Conteggio	Funzione, Scheduler	
DagBagDimensioni	dagbag_size	Conteggio	Funzione, elaborazione DAG	
DagCallbackEccezioni	dag.callback_exceptions	Conteggio	DAG, Tutti	
SLA fallito EmailAttempts	sla_email_notification_failure	Conteggio	Funzione, Scheduler	
TaskInstanceFinito	ti.finish.{id_giorno}.{id_attività}.{stato}	Conteggio	GIORNO, {dag_id} Attività, {task_id} Stato, {state}	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione
JobEnd	{job_name} _end	Conteggio	Job, {job_name}
JobHeartbeatFallimento	{job_name } _heartbea t_failure	Conteggio	Job, {job_name}
JobStart	{job_name} _inizio	Conteggio	Job, {job_name}
ManagerStalls	dag_proce ssing.man ager_stalls	Conteggio	Funzione, elaborazione DAG
OperatorFailures	operator_ failures_ {operator _name}	Conteggio	Operatore , {operator _name}
OperatorSuccesses	operator_ successes _ {operator _name}	Conteggio	Operatore , {operator _name}
OtherCallbackConta **Note** Disponibile in Apache Airflow v2.6.3 e versioni successive.	dag_proce ssing.oth er_callba ck_count	Conteggio	Funzione, Scheduler

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
Processes	dag_processing.processes	Conteggio	Funzione, elaborazione DAG	
SchedulerHeartbeat	scheduler_heartbeat	Conteggio	Funzione, Scheduler	
StartedTaskIstanze	ti.start.{id_giorno}.{id_attività}	Conteggio	DAG, Tutti Compito, tutti	
SlaCallbackConta	dag_processing.sla_callback_count **Note** Disponibile per Apache Airflow v2.6.3 e versioni successive.	Conteggio	Funzione, Scheduler	
TasksKilledEsternamente	scheduler.tasks.killed_externally	Conteggio	Funzione, Scheduler	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
TaskTimeoutErrore	celery.task_timeout_error	Conteggio	Funzione, Celery	
TaskInstanceCreatedUsingOperatore	task_instance_created- {operator_name}	Conteggio	Operatore, {operator_name}	
TaskInstancePreviouslySucceeded	precedente_successo_	Conteggio	DAG, Tutti Compito, tutti	
TaskInstanceFallimenti	ti_fallimenti	Conteggio	DAG, Tutti Compito, tutti	
TaskInstanceSuccessi	ti_successes	Conteggio	DAG, Tutti Compito, tutti	
TaskRemovedDa DAG	task_removed_da_dag. {id_giorno}	Conteggio	DAG, {dag_id}	
TaskRestoredA DAG	task_ripristinato_dag. {id_giorno}	Conteggio	DAG, {dag_id}	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
TriggersSucceeded **Note** Disponibile per Apache Airflow v2.7.2 e versioni successive.	triggers.riuscito	Conteggio	Funzione, Trigger	
TriggersFailed **Note** Disponibile per Apache Airflow v2.7.2 e versioni successive.	triggers.failed	Conteggio	Funzione, Trigger	
TriggersBlockedMainThread **Note** Disponibile per Apache Airflow v2.7.2 e versioni successive.	triggers.blocked_main_thread	Conteggio	Funzione, Trigger	
TriggerHeartbeat **Note** Disponibile per Apache Airflow v2.8.1 e versioni successive.	triggerer_heartbeat	Conteggio	Funzione, Triggerer	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
TaskInstanceCreatedUsingOperatore	airflow.task_instance_created_`{operator_name}` **Note** Disponibile per Apache Airflow v2.7.2 e versioni successive.	Conteggio	Operatore, `{operator_name}`	
ZombiesKilled	zombie uccisi	Conteggio	DAG, Tutti Compito, tutti	

Misuratori del flusso d'aria Apache

Le metriche di Apache Airflow in questa sezione contengono dati su Apache Airflow Gauges.

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
Errore FileRefresh DAG	dag_file_refresh_error	Conteggio	Funzione, elaborazione DAG	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
ImportErrors	dag_processing.import_errors	Conteggio	Funzione, elaborazione DAG	
Exception Failures	smart_sensor_operator.exception_failures	Conteggio	Funzione, Smart Sensor Operator	
ExecutedTasks	smart_sensor_operator.executed_tasks	Conteggio	Funzione, Smart Sensor Operator	
InfraFailures	smart_sensor_operator.infra_failures	Conteggio	Funzione, Smart Sensor Operator	
LoadedTasks	smart_sensor_operator.loaded_tasks	Conteggio	Funzione, Smart Sensor Operator	
TotalParseOra	dag_processing.total_parse_time	Secondi	Funzione, elaborazione DAG	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
Triggered DagEsegue **Note** Disponibile in Apache Airflow v2.6.3 e versioni successive.	dataset.triggered_dagruns	Conteggio	Funzione, Scheduler	
TriggersRunning **Note** Disponibile in Apache Airflow v2.7.2 e versioni successive.	triggers.running.*{nome host}*	Conteggio	Funzione, Trigger HostName, *{hostname}*	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
PoolDeferredSlot **Note** Disponibile in Apache Airflow v2.7.2 e versioni successive.	pool.deferred_slots.`{pool_name}`	Conteggio	Piscina, {pool_name}	
DAG FileProcessing LastRunSecondsAgo	dag_processing.last_run.seconds_ago.{nome_file}	Secondi	Nome file DAG, {dag_filename}	
OpenSlots	esecutore.open_slots	Conteggio	Funzione, Executor	
OrphanedTasksAdottato	scheduler.orphaned_tasks.adopted	Conteggio	Funzione, Scheduler	
OrphanedTasksCancellato	scheduler.orphaned_tasks.cleared	Conteggio	Funzione, Scheduler	
PokedExceptions	smart_sensor_operator.poked_exception	Conteggio	Funzione, Smart Sensor Operator	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
PokedSuccess	smart_sensor_operator.poked_success	Conteggio	Funzione, Smart Sensor Operator	
PokedTasks	smart_sensor_operator.poked_tasks	Conteggio	Funzione, Smart Sensor Operator	
PoolFailures	pool.open_slots.{pool_name}	Conteggio	Piscina, {pool_name}	
PoolStarvingAttività	pool.starving_tasks.{pool_name}	Conteggio	Piscina, {pool_name}	
PoolOpenSlot	pool.open_slots.{pool_name}	Conteggio	Piscina, {pool_name}	
PoolQueuedSlot	pool.queued_slots.{nome_pool}	Conteggio	Piscina, {pool_name}	
PoolRunningSlot	pool.running_slots.{pool_name}	Conteggio	Piscina, {pool_name}	
Processor Timeouts	dag_processing.processor_timeouts	Conteggio	Funzione, elaborazione DAG	
QueuedTasks	executor.queued_tasks	Conteggio	Funzione, Executor	
RunningTasks	executor.running_tasks	Conteggio	Funzione, Executor	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
TasksExecutable	scheduler .tasks.ex ecutable	Conteggio	Funzione, Scheduler	
TasksPending **Note** Non si applica ad Apache Airflow v2.2 e versioni successiv e.	scheduler .tasks.pending	Conteggio	Funzione, Scheduler	
TasksRunning	scheduler .tasks.running	Conteggio	Funzione, Scheduler	
TasksStarving	scheduler .tasks.starving	Conteggio	Funzione, Scheduler	
TasksWith outDagRun	scheduler .tasks.wi thout_dagrun	Conteggio	Funzione, Scheduler	

Timer Apache Airflow

Le metriche di Apache Airflow in questa sezione contengono dati sugli Apache Airflow Timer.

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
Raccogli i tag DBD	collect_db_dags	Millisecondi	Funzione, elaborazione DAG	
CriticalSectionDurata	scheduler.critical_section_duration	Millisecondi	Funzione, Scheduler	
CriticalSectionQueryDuration **Note** Disponibile per Apache Airflow v2.5.1 e versioni successive.	scheduler.critical_section_query_duration	Millisecondi	Funzione, Scheduler	
DAG DependencyCheck	dagrun.dependency-check.{id_giorno}	Millisecondi	DAG, {dag_id}	
DAG DurationFailed	dagrun.duration.failed.{dato_id}	Millisecondi	DAG, {dag_id}	
DAG DurationSuccess	dagrun.duration.success.{id_giorno}	Millisecondi	DAG, {dag_id}	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione	
DAG FileProcessing LastDuration	dag_processing.last_duration.{nome_file}	Secondi	Nome file DAG, {dag_filename}	
DAG ScheduleDelay	dagrun.schedule_delay.{id_giorno}	Millisecondi	DAG, {dag_id}	
FirstTaskSchedulingDelay	dagrun.{dag_id} .first_task_scheduling_delay	Millisecondi	Giorno, {dat_id}	
SchedulerLoopDurata **Note** Disponibile per Apache Airflow v2.5.1 e versioni successive.	scheduler.scheduler_loop_duration	Millisecondi	Funzione, Scheduler	
TaskInstanceDurata	giorno.{id_giorno}.{task_id} .durata	Millisecondi	DAG, {dat_id} Attività, {task_id}	

CloudWatch metrico	Metrica Apache Airflow	Unità	Dimensione
TaskInstanceQueuedDuration	giorno.{dag_id}.{task_id}.durata_in coda **Note** Disponibile per Apache Airflow v2.7.2 e versioni successive.	Millisecondi	DAG, {dag_id} Attività, {task_id}
TaskInstanceScheduledDuration **Note** Disponibile per Apache Airflow v2.7.2 e versioni successive.	giorno.{dag_id}.{task_id}.durata_pianificata	Millisecondi	GIORNO, {dag_id} Attività, {task_id}

Scelta delle metriche da segnalare

Puoi scegliere quali metriche di Apache Airflow vengono emesse o bloccate da Apache Airflow utilizzando le seguenti opzioni di configurazione di Amazon MWAA: CloudWatch

- **`metrics.metrics_allow_list`**— Un elenco di prefissi separati da virgole che puoi utilizzare per selezionare le metriche a cui il tuo ambiente emette dati. CloudWatch Utilizzate questa opzione se desiderate che Apache Airflow non invii tutte le metriche disponibili e selezioni invece un sottoinsieme di elementi. Ad esempio, `scheduler,executor,dagrun`.
- **`metrics.metrics_block_list`**— Un elenco di prefissi separati da virgole per filtrare le metriche che iniziano con gli elementi dell'elenco. Ad esempio, `scheduler,executor,dagrun`.

Se configurate entrambi `metrics.metrics_allow_list` e, Apache Airflow li ignora. `metrics.metrics_block_list` `metrics.metrics_block_list` Se configuri `metrics.metrics_block_list` ma non lo fai`metrics.metrics_allow_list`, Apache Airflow filtra gli elementi specificati in. `metrics.metrics_block_list`

> **Note**
>
> Le opzioni di `metrics.metrics_block_list` configurazione `metrics.metrics_allow_list` e si applicano solo ad Apache Airflow v2.6.3 e versioni successive. Per la versione precedente di Apache Airflow, usa e invece. `metrics.statsd_allow_list` `metrics.statsd_block_list`

Fasi successive

- Esplora il funzionamento dell'API Amazon MWAA utilizzato per pubblicare i parametri di salute dell'ambiente su. PublishMetrics

Parametri di container, code e database per Amazon MWAA

Oltre ai parametri di Apache Airflow, puoi monitorare i componenti sottostanti dei tuoi ambienti Amazon Managed Workflows for Apache Airflow utilizzando CloudWatch, che raccoglie dati grezzi ed elabora i dati in metriche leggibili quasi in tempo reale. Grazie a queste metriche ambientali, avrai una maggiore visibilità sugli indicatori chiave di prestazione per aiutarti a dimensionare in

modo appropriato gli ambienti e a risolvere i problemi relativi ai flussi di lavoro. Questi parametri si applicano a tutte le versioni di Apache Airflow supportate su Amazon MWAA.

Amazon MWAA fornirà l'utilizzo della CPU e della memoria per ogni container Amazon Elastic Container Service (Amazon ECS) e Amazon Aurora PostgreSQL e i parametri di Amazon Simple Queue Service (Amazon SQS) per il numero di messaggi e l'età del messaggio più vecchio, Amazon Relational Database Service (Amazon Relational Database Service) Parametri (Amazon RDS) per connessioni al database, profondità della coda su disco, operazioni di scrittura, latenza e velocità effettiva e metriche del proxy Amazon RDS. Queste metriche includono anche il numero di lavoratori di base, lavoratori aggiuntivi, pianificatori e server Web.

Queste statistiche vengono conservate per 15 mesi, in modo da poter accedere alle informazioni storiche e avere una prospettiva migliore sul motivo per cui una pianificazione non funziona e risolvere i problemi sottostanti. È anche possibile impostare allarmi che controllano determinate soglie e inviare notifiche o intraprendere azioni quando queste soglie vengono raggiunte. Per ulteriori informazioni, consulta la Amazon CloudWatch User Guide.

Argomenti

- Termini
- Dimensioni
- Accesso alle metriche nella console CloudWatch
- Elenco delle metriche

Termini

Spazio dei nomi

Un namespace è un contenitore per le CloudWatch metriche di un servizio. AWS Per Amazon MWAA, lo spazio dei nomi è. `AWS/MWAA`

CloudWatch metriche

Una CloudWatch metrica rappresenta un insieme ordinato nel tempo di punti dati specifici per. CloudWatch

Dimensione

Una dimensione è una coppia nome-valore che fa parte dell'identità di un parametro.

Unità

Una statistica ha un'unità di misura. Per Amazon MWAA, le unità includono Count.

Dimensioni

Questa sezione descrive il raggruppamento delle CloudWatch dimensioni per i parametri di Amazon MWAA in. CloudWatch

Dimensione	Descrizione
Cluster	Parametri per il minimo di tre container Amazon ECS utilizzati da un ambiente Amazon MWAA per eseguire i componenti di Apache Airflow: scheduler, worker e web server.
Queue	Metriche per le code di Amazon SQS che separano lo scheduler dai lavoratori. Quando i lavoratori leggono i messaggi, vengono considerati in volo e non disponibili per gli altri lavoratori. I messaggi diventano disponibili per la lettura da parte degli altri lavoratori se non vengono eliminati prima del timeout di visibilità di 12 ore.
Database	Parametri dei cluster Aurora utilizzati da Amazon MWAA. Ciò include i parametri per l'istanza del database principale e una replica di lettura per supportare le operazioni di lettura. Amazon MWAA pubblica i parametri del database per le istanze READER e WRITER.

Accesso alle metriche nella console CloudWatch

Questa sezione descrive come accedere ai parametri di Amazon MWAA in. CloudWatch

Per visualizzare i parametri prestazionali per una dimensione

1. Apri la pagina Metriche sulla console. CloudWatch
2. Usa il selettore AWS della regione per selezionare la tua regione.
3. Scegli lo spazio dei nomi AWS/MWAA.
4. Nella scheda Tutte le metriche, scegli una dimensione. Ad esempio, Cluster.
5. Scegli una CloudWatch metrica per una dimensione. Ad esempio, NumSchedulerso utilizzo della CPU. Quindi, scegli Grafica tutti i risultati della ricerca.
6. Scegli la scheda Metriche grafiche per visualizzare le metriche delle prestazioni.

Elenco delle metriche

Le tabelle seguenti elencano i parametri del cluster, della coda e dei servizi di database per Amazon MWAA. Per visualizzare le descrizioni delle metriche emesse direttamente da Amazon ECS, Amazon SQS o Amazon RDS, scegli il rispettivo link alla documentazione.

Argomenti

- Parametri cluster
- Parametri del database
- Parametri coda
- Parametri di Application Load Balancer

Parametri cluster

Le seguenti metriche si applicano a ogni programmatore, lavoratore di base, lavoratore aggiuntivo e server web. Per ulteriori informazioni e descrizioni di ogni metrica del cluster, consulta Metriche e dimensioni disponibili nella Amazon ECS Developer Guide.

Spazio dei nomi	Parametro	Unità
`AWS/MWAA`	`CPUUtilization`	Percentuale
`AWS/MWAA`	`MemoryUtilization`	Percentuale

Valutazione del numero di container di worker e server Web aggiuntivi

È possibile utilizzare le metriche dei componenti fornite nella dimensione Cluster, come descritto nella procedura seguente, per valutare il numero di worker o server Web aggiuntivi utilizzati da un ambiente in un determinato momento. È possibile farlo rappresentando graficamente l'utilizzo della CPU o la MemoryUtilizationmetrica e impostando il tipo di statistica su Sample Count. Il valore risultante è il numero totale di attività per il componente. `RUNNING AdditionalWorker` La comprensione del numero di istanze di worker aggiuntive utilizzate dall'ambiente può aiutare a valutare la scalabilità dell'ambiente e consentire di ottimizzare il numero di lavoratori aggiuntivi.

Workers

Per valutare il numero di lavoratori aggiuntivi utilizzando il AWS Management Console

1. Scegli lo spazio dei nomi AWS/MWAA.
2. Nella scheda Tutte le metriche, scegli la dimensione Cluster.
3. Nella dimensione Cluster, per AdditionalWorker, scegli l'utilizzo della CPU o la metrica. MemoryUtilization
4. Nella scheda Metriche grafiche, imposta Periodo su 1 minuto e Statistica su Numero di campioni.

Web servers

Per valutare il numero di server Web aggiuntivi utilizzando AWS Management Console

1. Scegli lo spazio dei nomi AWS/MWAA.
2. Nella scheda Tutte le metriche, scegli la dimensione Cluster.
3. Nella dimensione Cluster, per AdditionalWebservers, scegli l'utilizzo della CPU o la metrica. MemoryUtilization
4. Nella scheda Metriche grafiche, imposta Periodo su 1 minuto e Statistica su Numero di campioni.

Per ulteriori informazioni, consulta Service `RUNNING` task count nella Amazon Elastic Container Service Developer Guide.

Parametri del database

Le seguenti metriche si applicano a ogni istanza di database associata all'ambiente Amazon MWAA.

Spazio dei nomi	Parametro	Unità
AWS/MWAA	CPUUtilization	Percentuale
AWS/MWAA	DatabaseConnections	Conteggio
AWS/MWAA	DiskQueueDepth	Conteggio
AWS/MWAA	FreeableMemory	Byte
AWS/MWAA	VolumeWriteIOPS	Conta ogni cinque minuti
AWS/MWAA	WriteIOPS	Conteggio al secondo
AWS/MWAA	WriteLatency	Secondi
AWS/MWAA	WriteThroughput	Byte al secondo

Parametri coda

Per ulteriori informazioni sulle unità e le descrizioni dei seguenti parametri di coda, consulta Parametri disponibili CloudWatch per Amazon SQS nella Amazon Simple Queue Service Developer Guide.

Spazio dei nomi	Parametro	Unità
AWS/MWAA	ApproximateAgeOfOldestTask	Secondi
AWS/MWAA	RunningTasks	Conteggio
AWS/MWAA	QueuedTasks	Conteggio

Parametri di Application Load Balancer

Le metriche di Application Load Balancer si applicano ai server Web in esecuzione nel tuo ambiente. Amazon MWAA utilizza questi parametri per scalare i server Web in base alla quantità di traffico. Per ulteriori informazioni sulle unità e le descrizioni delle seguenti metriche del load balancer, consulta le CloudWatch metriche per il tuo Application Load Balancer nella Application Load Balancer User Guide.

Spazio dei nomi	Parametro	Unità
`AWS/MWAA`	`ActiveConnectionCount`	Conteggio

Sicurezza nei flussi di lavoro gestiti di Amazon per Apache Airflow

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 AWS e te (il cliente). Il modello di responsabilità condivisa descrive questo aspetto come sicurezza del cloud e sicurezza nel cloud:

- Sicurezza del cloud: AWS è responsabile della protezione dell'infrastruttura che gestisce AWS i servizi nel AWS cloud. AWS ti fornisce anche servizi che puoi utilizzare in modo sicuro. I revisori esterni testano e verificano regolarmente l'efficacia della nostra sicurezza nell'ambito dei AWS Programmi di AWS conformità dei Programmi di conformità dei di . Per ulteriori informazioni sui programmi di conformità applicabili ad Amazon MWAA, consulta AWS Services in Scope by Compliance Program Program.
- Sicurezza nel cloud: la tua responsabilità è determinata dal AWS servizio che utilizzi. Sei anche responsabile di altri fattori, tra cui la riservatezza dei dati, i requisiti della tua azienda e le leggi e normative vigenti.

Questa documentazione ti aiuta a capire come applicare il modello di responsabilità condivisa quando usi Amazon Managed Workflows for Apache Airflow. Ti mostra come configurare Amazon MWAA per soddisfare i tuoi obiettivi di sicurezza e conformità. Scopri anche come utilizzare altri AWS servizi che ti aiutano a monitorare e proteggere le tue risorse Amazon MWAA.

In questa sezione:

- Protezione dei dati in Amazon Managed Workflows per Apache Airflow
- AWS Identity and Access Management
- Convalida della conformità per Amazon Managed Workflows for Apache Airflow
- Resilienza nei flussi di lavoro gestiti di Amazon per Apache Airflow
- Sicurezza dell'infrastruttura in Amazon MWAA
- Analisi della configurazione e delle vulnerabilità in Amazon MWAA
- Best practice di sicurezza su Amazon MWAA

Protezione dei dati in Amazon Managed Workflows per Apache Airflow

Il modello di responsabilità AWS condivisa modello si applica alla protezione dei dati in Amazon Managed Workflows for Apache Airflow. Come descritto in questo modello, AWS è responsabile della protezione dell'infrastruttura globale che gestisce tutti i. Cloud AWS L'utente è responsabile di mantenere il controllo sui contenuti ospitati su questa infrastruttura. Questi contenuti comprendono la configurazione della protezione e le 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 account utente individuali con AWS Identity and Access Management (IAM). In questo modo, a ogni utente verranno assegnate solo le autorizzazioni necessarie per svolgere il proprio lavoro. Ti suggeriamo, inoltre, di proteggere i dati nei seguenti modi:

- Utilizza l'autenticazione a più fattori (MFA) con ogni account.
- Utilizza SSL/TLS per comunicare con le risorse. AWS È consigliabile TLS 1.2 o versioni successive.
- 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 AWS dei servizi.
- Utilizza i servizi di sicurezza gestiti avanzati, ad esempio Amazon Macie, che aiutano a individuare e proteggere i dati personali archiviati in Amazon S3.

Ti suggeriamo vivamente di non inserire mai informazioni identificative sensibili, ad esempio i numeri di account dei clienti, in campi a formato libero, ad esempio un campo Nome. Ciò include quando lavori con Amazon MWAA o altri AWS servizi utilizzando la console, l'API o AWS gli AWS CLI SDK. I dati inseriti nei tag o nei campi in formato libero utilizzati per i nomi possono essere utilizzati per i log di fatturazione o di diagnostica. Quando fornisci un URL a un server esterno, non includere informazioni sulle credenziali nell'URL per convalidare la tua richiesta a tale server.

Crittografia su Amazon MWAA

I seguenti argomenti descrivono come Amazon MWAA protegge i dati a riposo e in transito. Utilizza queste informazioni per scoprire come Amazon MWAA si integra AWS KMS per crittografare i dati inattivi e come i dati vengono crittografati utilizzando il protocollo Transport Layer Security (TLS) in transito.

Argomenti

- Crittografia a riposo
- Crittografia in transito

Crittografia a riposo

Su Amazon MWAA, i dati inattivi sono dati che il servizio salva su supporti persistenti.

Puoi utilizzare una chiave AWS proprietaria per la crittografia dei dati inattivi o, facoltativamente, fornire una chiave gestita dal cliente per una crittografia aggiuntiva quando crei un ambiente. Se scegli di utilizzare una chiave KMS gestita dal cliente, questa deve trovarsi nello stesso account delle altre AWS risorse e servizi che utilizzi con il tuo ambiente.

Per utilizzare una chiave KMS gestita dal cliente, è necessario allegare la dichiarazione politica richiesta per CloudWatch l'accesso alla politica chiave. Quando utilizzi una chiave KMS gestita dal cliente per il tuo ambiente, Amazon MWAA assegna quattro sovvenzioni per tuo conto. Per ulteriori informazioni sulle sovvenzioni che Amazon MWAA attribuisce a una chiave KMS gestita dal cliente, consulta Chiavi gestite dal cliente per la crittografia dei dati.

Se non specifichi una chiave KMS gestita dal cliente, per impostazione predefinita, Amazon MWAA utilizza una chiave KMS di AWS proprietà per crittografare e decrittografare i dati. Ti consigliamo di utilizzare una chiave KMS AWS proprietaria per gestire la crittografia dei dati su Amazon MWAA.

> **Note**
>
> Paghi per lo storage e l'uso di chiavi KMS AWS possedute o gestite dal cliente su Amazon MWAA. Per ulteriori informazioni, consulta la sezione Prezzi di AWS KMS.

Artefatti di crittografia

Specifichi gli artefatti di crittografia utilizzati per la crittografia at rest specificando una chiave di AWS proprietà o una chiave gestita dal cliente quando crei il tuo ambiente Amazon MWAA. Amazon MWAA aggiunge le sovvenzioni necessarie alla chiave specificata.

Amazon S3: i dati di Amazon S3 vengono crittografati a livello di oggetto utilizzando Server-Side Encryption (SSE). La crittografia e la decrittografia di Amazon S3 avvengono nel bucket Amazon S3 in cui sono archiviati il codice DAG e i file di supporto. Gli oggetti vengono crittografati quando vengono caricati su Amazon S3 e decrittografati quando vengono scaricati nell'ambiente Amazon MWAA. Per impostazione predefinita, se utilizzi una chiave KMS gestita dal cliente, Amazon MWAA la utilizza per leggere e decrittografare i dati sul tuo bucket Amazon S3.

CloudWatch Registri: se utilizzi una chiave KMS di AWS proprietà, i log di Apache Airflow inviati a Logs vengono crittografati utilizzando Server-Side Encryption (SSE) con la chiave KMS di proprietà di CloudWatch Logs. CloudWatch AWS Se utilizzi una chiave KMS gestita dal cliente, devi aggiungere una politica chiave alla tua chiave KMS per consentire a Logs di utilizzare la tua chiave. CloudWatch

Amazon SQS: Amazon MWAA crea una coda Amazon SQS per il tuo ambiente. Amazon MWAA gestisce la crittografia dei dati trasmessi da e verso la coda utilizzando Server-Side Encryption (SSE) con una chiave KMS di AWS proprietà o una chiave KMS gestita dal cliente specificata. Devi aggiungere le autorizzazioni Amazon SQS al tuo ruolo di esecuzione indipendentemente dal fatto che tu stia utilizzando una chiave KMS di AWS proprietà o gestita dal cliente.

Aurora PostgreSQL — Amazon MWAA crea un cluster PostgreSQL per il tuo ambiente. Aurora PostgreSQL crittografa i contenuti con una chiave KMS di AWS proprietà o gestita dal cliente utilizzando Server-Side Encryption (SSE). Se utilizzi una chiave KMS gestita dal cliente, Amazon RDS aggiunge almeno due concessioni alla chiave: una per il cluster e una per l'istanza del database. Amazon RDS potrebbe creare ulteriori sovvenzioni se scegli di utilizzare la chiave KMS gestita dal cliente in più ambienti. Per ulteriori informazioni, consulta Protezione dei dati in Amazon RDS.

Crittografia in transito

I dati in transito sono indicati come dati che possono essere intercettati mentre viaggiano sulla rete.

Transport Layer Security (TLS) crittografa gli oggetti Amazon MWAA in transito tra i componenti Apache Airflow dell'ambiente e altri servizi AWS che si integrano con Amazon MWAA, come Amazon S3. Per ulteriori informazioni sulla crittografia di Amazon S3, consulta Protezione dei dati tramite crittografia.

Utilizzo di chiavi gestite dal cliente per la crittografia

Facoltativamente, puoi fornire una chiave gestita dal cliente per la crittografia dei dati nel tuo ambiente. È necessario creare la chiave KMS gestita dal cliente nella stessa regione dell'istanza di ambiente Amazon MWAA e del bucket Amazon S3 in cui archiviare le risorse per i flussi di lavoro. Se la chiave KMS gestita dal cliente specificata si trova in un account diverso da quello utilizzato per configurare un ambiente, è necessario specificare la chiave utilizzando il relativo ARN per l'accesso tra account. Per ulteriori informazioni sulla creazione di chiavi, consulta Creating Keys nella Developer Guide.AWS Key Management Service

Cosa è supportato

AWS KMS funzionalità	Supportato	
Un ID AWS KMS chiave o ARN.	Sì	
Qualsiasi alias AWS KMS chiave.	No	
Una chiave AWS KMS multiregionale.	No	

Utilizzo di Grants for Encryption

Questo argomento descrive le sovvenzioni concesse da Amazon MWAA a una chiave KMS gestita dal cliente per crittografare e decrittografare i dati.

Come funziona

Esistono due meccanismi di controllo degli accessi basati sulle risorse supportati da una chiave KMS gestita dal cliente: una policy chiave e una AWS KMS concessione.

Una politica chiave viene utilizzata quando l'autorizzazione è per lo più statica e utilizzata in modalità di servizio sincrona. Una concessione viene utilizzata quando sono richieste autorizzazioni più dinamiche e granulari, ad esempio quando un servizio deve definire autorizzazioni di accesso diverse per se stesso o per altri account.

Amazon MWAA utilizza e associa quattro politiche di concessione alla chiave KMS gestita dal cliente. Ciò è dovuto alle autorizzazioni granulari necessarie per un ambiente per crittografare i dati inattivi da CloudWatch Logs, dalla coda Amazon SQS, dal database del database Aurora PostgreSQL, dai segreti di Secrets Manager, dal bucket Amazon S3 e dalle tabelle DynamoDB.

Quando crei un ambiente Amazon MWAA e specifichi una chiave KMS gestita dal cliente, Amazon MWAA allega le politiche di concessione alla tua chiave KMS gestita dal cliente. Queste politiche consentono ad Amazon MWAA di `airflow.region}.amazonaws.com` utilizzare la chiave KMS gestita dai clienti per crittografare le risorse di proprietà di Amazon MWAA per tuo conto.

Amazon MWAA crea e assegna ulteriori concessioni a una chiave KMS specificata per tuo conto. Ciò include le politiche per ritirare una sovvenzione in caso di eliminazione dell'ambiente, per utilizzare la chiave KMS gestita dal cliente per la crittografia lato client (CSE) e per il ruolo di AWS Fargate esecuzione che deve accedere ai segreti protetti dalla chiave gestita dal cliente in Secrets Manager.

Politiche di concessione

Amazon MWAA aggiunge le seguenti concessioni di policy basate sulle risorse per tuo conto a una chiave KMS gestita dal cliente. Queste politiche consentono al beneficiario e al committente (Amazon MWAA) di eseguire le azioni definite nella politica.

Grant 1: utilizzato per creare risorse sul piano dati

```
{
            "Name": "mwaa-grant-for-env-mgmt-role-environment name",
            "GranteePrincipal": "airflow.region.amazonaws.com",
            "RetiringPrincipal": "airflow.region.amazonaws.com",
            "Operations": [
              "kms:Encrypt",
              "kms:Decrypt",
              "kms:ReEncrypt*",
              "kms:GenerateDataKey*",
              "kms:CreateGrant",
              "kms:DescribeKey",
              "kms:RetireGrant"
            ]
          }
```

Grant 2: utilizzato per **`ControllerLambdaExecutionRole`** l'accesso

```
{
            "Name": "mwaa-grant-for-lambda-exec-environment name",
            "GranteePrincipal": "airflow.region.amazonaws.com",
            "RetiringPrincipal": "airflow.region.amazonaws.com",
            "Operations": [
              "kms:Encrypt",
              "kms:Decrypt",
              "kms:ReEncrypt*",
              "kms:GenerateDataKey*",
              "kms:DescribeKey",
              "kms:RetireGrant"
            ]
          }
```

Grant 3: utilizzato per **`CfnManagementLambdaExecutionRole`** l'accesso

```
{
              "Name": " mwaa-grant-for-cfn-mgmt-environment name",
              "GranteePrincipal": "airflow.region.amazonaws.com",
              "RetiringPrincipal": "airflow.region.amazonaws.com",
              "Operations": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
              ]
            }
```

Grant 4: utilizzato per il ruolo di esecuzione di Fargate per accedere ai segreti del backend

```
{
                "Name": "mwaa-fargate-access-for-environment name",
                "GranteePrincipal": "airflow.region.amazonaws.com",
                "RetiringPrincipal": "airflow.region.amazonaws.com",
                "Operations": [
                  "kms:Encrypt",
                  "kms:Decrypt",
```

```
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey",
                "kms:RetireGrant"
            ]
        }
```

Allegare le politiche chiave a una chiave gestita dal cliente

Se scegli di utilizzare la tua chiave KMS gestita dal cliente con Amazon MWAA, devi allegare la seguente policy alla chiave per consentire ad Amazon MWAA di utilizzarla per crittografare i tuoi dati.

Se la chiave KMS gestita dal cliente che hai utilizzato per il tuo ambiente Amazon MWAA non è già configurata per funzionare CloudWatch, devi aggiornare la policy delle chiavi per consentire i log crittografati. CloudWatch Per ulteriori informazioni, consulta il servizio Encrypt log in using. CloudWatch AWS Key Management Service

L'esempio seguente rappresenta una politica chiave per CloudWatch Logs. Sostituisci i valori di esempio forniti per la regione.

```
{
          "Effect": "Allow",
          "Principal": {
          "Service": "logs.us-west-2.amazonaws.com"
        },
        "Action": [
          "kms:Encrypt*",
          "kms:Decrypt*",
          "kms:ReEncrypt*",
          "kms:GenerateDataKey*",
          "kms:Describe*"
        ],
        "Resource": "*",
        "Condition": {
          "ArnLike": {
            "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-west-2:*:*"
            }
          }
        }
```

AWS Identity and Access Management

AWS Identity and Access Management (IAM) è un AWS servizio che aiuta un amministratore a controllare in modo sicuro l'accesso alle risorse. AWS Gli amministratori IAM controllano chi può essere autenticato (effettuato l'accesso) e autorizzato (disporre delle autorizzazioni) a utilizzare le risorse di Amazon Managed Workflows for Apache Airflow. IAM è un AWS servizio che puoi utilizzare senza costi aggiuntivi.

Questo argomento fornisce una panoramica di base sull'utilizzo di Amazon MWAA AWS Identity and Access Management (IAM). Per ulteriori informazioni sulla gestione dell'accesso ad Amazon MWAA, consulta. Gestione dell'accesso a un ambiente Amazon MWAA

Indice

- Destinatari
- Autenticazione con identità
- Gestione dell'accesso tramite policy
- Consentire agli utenti di visualizzare le loro autorizzazioni
- Risoluzione dei problemi relativi all'identità e all'accesso ad Amazon Managed Workflows per Apache Airflow
- Come funziona Amazon MWAA con IAM

Destinatari

Il modo in cui utilizzi AWS Identity and Access Management (IAM) varia a seconda del lavoro svolto in Amazon MWAA.

Utente del servizio: se utilizzi il servizio Amazon MWAA per svolgere il tuo lavoro, l'amministratore ti fornisce le credenziali e le autorizzazioni necessarie. Man mano che utilizzi più funzionalità di Amazon MWAA 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 di Amazon MWAA, consulta. Risoluzione dei problemi relativi all'identità e all'accesso ad Amazon Managed Workflows per Apache Airflow

Amministratore del servizio: se sei responsabile delle risorse Amazon MWAA della tua azienda, probabilmente hai pieno accesso ad Amazon MWAA. È tuo compito determinare a quali funzionalità

e risorse di Amazon MWAA 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 ulteriori informazioni su come la tua azienda può utilizzare IAM con Amazon MWAA, consulta. Come funziona Amazon MWAA con IAM

Amministratore IAM: se sei un amministratore IAM, potresti voler saperne di più su come scrivere policy per gestire l'accesso ad Amazon MWAA. Per visualizzare esempi di policy basate sull'identità di Amazon MWAA che puoi utilizzare in IAM, consulta. Esempi di policy basate sull'identità di Amazon MWAA

Autenticazione con identità

L'autenticazione è il modo in cui accedi utilizzando le tue credenziali di identità. AWS 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 AWSnella Guida per l'utente IAM.

Account AWS utente root

Quando si crea un account Account AWS, si inizia con un'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. Conserva le credenziali dell'utente root e utilizzale 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 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, se si hanno 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 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 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 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 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 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 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 diretto (FAS): quando utilizzi un utente o un ruolo IAM per eseguire azioni AWS, sei considerato un preside. 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 AWSnella Guida per l'utente 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'azione 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 IAM.

Per informazioni sull'utilizzo dei ruoli IAM, consulta Quando creare un ruolo IAM (invece di un utente) nella Guida per l'utente IAM.

Gestione dell'accesso tramite 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 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 operazioni sulle risorse di cui hanno bisogno, un amministratore

IAM può creare policy IAM. L'amministratore può quindi aggiungere le policy IAM ai ruoli e gli utenti possono assumere i ruoli.

Le policy IAM definiscono le autorizzazioni relative a un'operazione, a prescindere dal metodo utilizzato per eseguirla. Ad esempio, supponiamo di disporre di una policy che consente l'operazione `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 sulle 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 ruolo 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 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 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 utilizzarli 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.

Liste di controllo degli accessi (ACL)

Le liste di controllo degli accessi (ACL) controllano quali principali (membri, utenti o ruoli dell'account) hanno le autorizzazioni per accedere a una risorsa. Le ACL sono simili alle policy basate su risorse, sebbene non utilizzino il formato del documento di policy JSON.

Amazon S3 e Amazon VPC sono esempi di servizi che supportano gli ACL. AWS WAF Per maggiori informazioni sulle ACL, consulta Panoramica delle liste di controllo degli accessi (ACL) nella Guida per gli sviluppatori di Amazon Simple Storage Service.

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 funzionalità 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 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 IAM.

Più tipi di policy

Quando più tipi di policy si applicano a una richiesta, le autorizzazioni risultanti sono più complicate da comprendere. Per sapere come si AWS determina se consentire una richiesta quando sono coinvolti più tipi di policy, consulta Logica di valutazione delle policy nella IAM User Guide.

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 policy include le autorizzazioni per completare questa azione sulla console o utilizzando l'API or 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": "*"
        }
    ]
}
```

Risoluzione dei problemi relativi all'identità e all'accesso ad Amazon Managed Workflows per Apache Airflow

Utilizza le seguenti informazioni per aiutarti a diagnosticare e risolvere i problemi più comuni che potresti riscontrare quando lavori con Amazon MWAA e IAM.

Non sono autorizzato a eseguire un'azione in Amazon MWAA

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 da cui si sono ricevuti il nome utente e la password.

Non sono autorizzato a eseguire iam: PassRole

Se ricevi un messaggio di errore indicante che non sei autorizzato a eseguire l'`iam:PassRole`azione, le tue politiche devono essere aggiornate per consentirti di trasferire un ruolo ad Amazon MWAA.

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 Amazon MWAA. Tuttavia, l'azione richiede che il servizio disponga delle autorizzazioni concesse da un ruolo di servizio. Mary non dispone delle autorizzazioni per passare il ruolo al servizio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In questo caso, le policy di Mary devono essere aggiornate per poter eseguire l'operazione `iam:PassRole`.

Se hai bisogno di aiuto, contatta il tuo AWS amministratore. L'amministratore è la persona che ti ha fornito le credenziali di accesso.

Voglio consentire a persone esterne al mio AWS account di accedere alle mie risorse Amazon MWAA

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 degli accessi (ACL), utilizza tali policy per concedere alle persone l'accesso alle tue risorse.

Per ulteriori informazioni, consulta gli argomenti seguenti:

- Per sapere se Amazon MWAA supporta queste funzionalità, consulta. Come funziona Amazon MWAA con IAM
- Per sapere 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 IAM.
- Per informazioni sulle differenze tra l'utilizzo di ruoli e policy basate su risorse per l'accesso multi-account, consultare Differenza tra i ruoli IAM e le policy basate su risorse nella Guida per l'utente di IAM.

Come funziona Amazon MWAA con IAM

Amazon MWAA utilizza policy basate sull'identità IAM per concedere autorizzazioni ad azioni e risorse Amazon MWAA. Per esempi consigliati di policy IAM personalizzate che puoi utilizzare per controllare l'accesso alle tue risorse Amazon MWAA, consulta. the section called “Accesso a un ambiente Amazon MWAA”

Per avere una visione di alto livello di come Amazon MWAA e altri AWS servizi funzionano con IAM, consulta AWS Services That Work with IAM nella IAM User Guide.

Policy basate sull'identità di Amazon MWAA

Con le policy basate su identità di IAM, è possibile specificare quali operazioni e risorse sono consentite o rifiutate, nonché le condizioni in base alle quali le operazioni sono consentite o rifiutate. Amazon MWAA supporta azioni, risorse e chiavi di condizione specifiche.

I passaggi seguenti mostrano come creare una nuova policy JSON utilizzando la console IAM. Questa policy fornisce l'accesso in sola lettura alle tue risorse Amazon MWAA.

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": [
                "airflow:ListEnvironments",
                "airflow:GetEnvironment",
                "airflow:ListTagsForResource"
            ],
            "Resource": "*"
        }
    ]
}
```

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.

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 azioni 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 istruzioni della policy devono includere un elemento `Action` o un elemento `NotAction`. L'elemento `Action` elenca le azioni consentite dalla policy. L'elemento `NotAction` elenca le operazioni non consentite.

Le azioni definite per Amazon MWAA riflettono le attività che puoi eseguire utilizzando Amazon MWAA. Le operazioni delle policy in Detective hanno il seguente prefisso: `airflow:`.

Per specificare più operazioni, è possibile utilizzare i caratteri jolly (*). Invece di elencare queste azioni separatamente, puoi concedere l'accesso a tutte le azioni che terminano con la parola, ad esempio,. `environment`

Per visualizzare un elenco di azioni Amazon MWAA, consulta Actions Defined by Amazon Managed Workflows for Apache Airflow nella IAM User Guide.

Esempi di policy basate sull'identità di Amazon MWAA

Per visualizzare le politiche di Amazon MWAA, consulta. Gestione dell'accesso a un ambiente Amazon MWAA

Per impostazione predefinita, gli utenti e i ruoli IAM non dispongono dell'autorizzazione per creare o modificare risorse Amazon MWAA. Inoltre, non possono eseguire attività utilizzando l'API AWS Management Console AWS CLI, o AWS .

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 collegare queste policy a utenti o gruppi IAM che richiedono tali autorizzazioni.

> **Important**
>
> Ti consigliamo di utilizzare ruoli IAM e credenziali temporanee per fornire l'accesso alle tue risorse Amazon MWAA. Evita di allegare policy di autorizzazione direttamente agli utenti IAM.

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.

Argomenti

- Best practice delle policy
- Utilizzo della console Amazon MWAA
- Consentire agli utenti di visualizzare le loro autorizzazioni

Best practice delle policy

Le policy basate sull'identità determinano se qualcuno può creare, accedere o eliminare risorse Amazon MWAA 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 AWSo Policy gestite da AWS per le funzioni dei processi nella Guida per l'utente IAM.

- Applica le autorizzazioni con privilegio minimo: 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 IAM.
- Condizioni d'uso nelle policy IAM per limitare ulteriormente l'accesso: per limitare l'accesso a operazioni 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 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 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 IAM.

Per maggiori informazioni sulle best practice in IAM, consulta Best practice di sicurezza in IAM nella Guida per l'utente di IAM.

Utilizzo della console Amazon MWAA

Per utilizzare la console Amazon MWAA, l'utente o il ruolo deve avere accesso alle azioni pertinenti, che corrispondono alle azioni corrispondenti nell'API.

Per visualizzare le politiche di Amazon MWAA, consulta. Gestione dell'accesso a un ambiente Amazon MWAA

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": "*"
        }
    ]
}
```

Convalida della conformità per Amazon Managed Workflows for Apache Airflow

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 i passaggi per l'implementazione di 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)).
- Valutazione delle risorse con regole nella Guida per gli AWS Config sviluppatori: 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— In questo modo Servizio AWS è possibile verificare continuamente AWS l'utilizzo per semplificare la gestione dei rischi e la conformità alle normative e agli standard di settore.

Resilienza nei flussi di lavoro gestiti di Amazon per Apache Airflow

L'infrastruttura AWS globale è costruita attorno AWS a regioni e zone di disponibilità. Le regioni forniscono più zone di disponibilità fisicamente separate e isolate, connesse tramite reti altamente ridondanti, a bassa latenza e throughput elevato. Con le zone di disponibilità, è possibile progettare e gestire applicazioni e database che eseguono il failover automatico 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 Infrastruttura AWS globale.

Sicurezza dell'infrastruttura in Amazon MWAA

In quanto servizio gestito, Amazon Managed Workflows for Apache Airflow è protetto dalla sicurezza di rete AWS globale. Per informazioni sui servizi di AWS 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.

Utilizzi chiamate API AWS pubblicate per accedere ad Amazon MWAA 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.

Analisi della configurazione e delle vulnerabilità in Amazon MWAA

La configurazione e i controlli IT sono una responsabilità condivisa tra te AWS e te, nostro cliente.

Amazon Managed Workflows for Apache Airflow applica periodicamente patch e aggiorna Apache Airflow nei tuoi ambienti. Dovresti assicurarti che vengano utilizzate le politiche di accesso appropriate per i tuoi VPC.

Per ulteriori dettagli, consulta le seguenti risorse :

- Convalida della conformità per Amazon Managed Workflows for Apache Airflow
- Modello di responsabilità condivisa
- Amazon Web Services: panoramica dei processi di sicurezza
- Sicurezza dell'infrastruttura in Amazon MWAA
- Best practice di sicurezza su Amazon MWAA

Best practice di sicurezza su Amazon MWAA

Amazon MWAA offre una serie di funzionalità di sicurezza da prendere in considerazione 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 politiche di autorizzazione meno permissive. Concedi le autorizzazioni solo alle risorse o alle azioni di cui gli utenti hanno bisogno per eseguire le attività.
- Utilizzalo AWS CloudTrail per monitorare l'attività degli utenti nel tuo account.

- Assicurati che la policy del bucket di Amazon S3 e gli ACL degli oggetti concedano le autorizzazioni agli utenti dell'ambiente Amazon MWAA associato per inserire oggetti nel bucket. Ciò garantisce che gli utenti con le autorizzazioni per aggiungere flussi di lavoro al bucket dispongano anche delle autorizzazioni per eseguire i flussi di lavoro in Airflow.
- Usa i bucket Amazon S3 associati agli ambienti Amazon MWAA. Il tuo bucket Amazon S3 può avere qualsiasi nome. Non memorizzare altri oggetti nel bucket o utilizzare il bucket con un altro servizio.

Le migliori pratiche di sicurezza in Apache Airflow

Apache Airflow non è multi-tenant. Sebbene esistano misure di controllo degli accessi per limitare alcune funzionalità a utenti specifici, implementate da Amazon MWAA, i creatori di DAG hanno la possibilità di scrivere DAG in grado di modificare i privilegi utente di Apache Airflow e interagire con il metadatabase sottostante.

Quando lavori con Apache Airflow su Amazon MWAA, consigliamo i seguenti passaggi per garantire la sicurezza dei metadatabase e dei DAG del tuo ambiente.

- Utilizza ambienti separati per team separati con accesso alla scrittura DAG o la possibilità di aggiungere file alla cartella Amazon `/dags` S3, presupponendo che tutto ciò che è accessibile tramite le connessioni Amazon MWAA Execution Role o Apache Airflow sia accessibile anche agli utenti che possono scrivere nell'ambiente.
- Non fornire l'accesso diretto alle cartelle DAG di Amazon S3. Utilizza invece strumenti CI/CD per scrivere DAG su Amazon S3, con una fase di convalida che assicuri che il codice DAG soddisfi le linee guida di sicurezza del tuo team.
- Impedisci l'accesso degli utenti al bucket Amazon S3 del tuo ambiente. Utilizza invece un DAG factory che genera DAG basati su un file di definizione YAML, JSON o altro tipo di definizione archiviato in una posizione separata dal bucket Amazon MWAA Amazon S3 in cui memorizzi i DAG.
- Memorizza i segreti in Secrets Manager. Sebbene ciò non impedirà agli utenti che possono scrivere DAG di leggere i segreti, impedirà loro di modificare i segreti utilizzati dall'ambiente.

Rilevamento delle modifiche ai privilegi utente di Apache Airflow

È possibile utilizzare CloudWatch Logs Insights per rilevare le occorrenze di DAG che modificano i privilegi utente di Apache Airflow. A tale scopo, puoi utilizzare una regola EventBridge pianificata, una

funzione Lambda e CloudWatch Logs Insights per inviare notifiche alle CloudWatch metriche ogni volta che uno dei tuoi DAG modifica i privilegi utente di Apache Airflow.

Prerequisiti

Per completare i seguenti passaggi, avrai bisogno di quanto segue:

- Un ambiente Amazon MWAA con tutti i tipi di log Apache Airflow abilitati a livello di log. `INFO` Per ulteriori informazioni, consulta the section called “Visualizzazione dei registri Airflow”.

Per configurare le notifiche per le modifiche ai privilegi utente di Apache Airflow

1. Crea una funzione Lambda che esegua la seguente stringa di query CloudWatch Logs Insights sui cinque gruppi di log dell'ambiente Amazon MWAA (`DAGProcessing`,,`Scheduler`, `Task` e). `WebServer` `Worker`

   ```
   fields @log, @timestamp, @message | filter @message like "add-role" | stats count()
    by @log
   ```

2. Crea una EventBridge regola che viene eseguita secondo una pianificazione, con la funzione Lambda creata nel passaggio precedente come obiettivo della regola. Configura la tua pianificazione utilizzando un'espressione cron o rate da eseguire a intervalli regolari.

Versioni di Apache Airflow su Amazon Managed Workflows per Apache Airflow

Questa pagina descrive le versioni di Apache Airflow supportate da Amazon Managed Workflows for Apache Airflow e le strategie che consigliamo per eseguire l'aggiornamento alla versione più recente.

Argomenti

- Informazioni sulle versioni di Amazon MWAA
- Versione più recente
- Versioni di Apache Airflow
- Componenti Apache Airflow
- Aggiornamento della versione di Apache Airflow
- Versioni obsolete di Apache Airflow
- Supporto e domande frequenti sulla versione di Apache Airflow

Informazioni sulle versioni di Amazon MWAA

Amazon MWAA crea immagini di container che raggruppano le versioni di Apache Airflow con altri binari e librerie Python comuni. L'immagine utilizza l'installazione di base di Apache Airflow per la versione specificata. Quando si crea un ambiente, si specifica una versione dell'immagine da utilizzare. Una volta creato, l'ambiente continua a utilizzare la versione dell'immagine specificata fino a quando non viene aggiornato a una versione successiva.

Versione più recente

Amazon MWAA supporta più di una versione di Apache Airflow. Se non specifichi una versione dell'immagine quando crei un ambiente, Amazon MWAA crea un ambiente utilizzando l'ultima versione supportata di Apache Airflow.

Versioni di Apache Airflow

Le seguenti versioni di Apache Airflow sono supportate su Amazon Managed Workflows for Apache Airflow.

Note

- A partire da Apache Airflow v2.2.2, Amazon MWAA supporta l'installazione di requisiti Python, pacchetti provider e plug-in personalizzati direttamente sul server Web Apache Airflow.
- A partire da Apache Airflow v2.7.2, il file dei requisiti deve includere una dichiarazione. `--constraint` Se non fornisci un vincolo, Amazon MWAA te ne specificherà uno per garantire che i pacchetti elencati nei tuoi requisiti siano compatibili con la versione di Apache Airflow che stai utilizzando.

  Per ulteriori informazioni sull'impostazione dei vincoli nel file dei requisiti, consulta Installazione delle dipendenze in Python.

Versione Apache Airflow	Guida Apache Airflow	Vincoli di Apache Airflow	Versione di Python
v2.8.1	Guida di riferimento di Apache Airflow v2.8.1	File dei vincoli di Apache Airflow v2.8.1	Python 3.11
v2.7.2	Guida di riferimento di Apache Airflow v2.7.2	File dei vincoli di Apache Airflow v2.7.2	Python 3.11
v2.6.3	Guida di riferimento di Apache Airflow v2.6.3	File dei vincoli di Apache Airflow v2.6.3	Python 3.10
v2.5.1	Guida di riferimento di Apache Airflow v2.5.1	File dei vincoli di Apache Airflow v2.5.1	Python 3.10
v2.4.3	Guida di riferimento di Apache Airflow v2.4.3	File dei vincoli di Apache Airflow v2.4.3	Python 3.10
v2.2.2	Guida di riferimento di Apache Airflow v2.2.2	File dei vincoli di Apache Airflow v2.2.2	Python 3.7
v2.0.2	Guida di riferimento di Apache Airflow v2.0.2	File dei vincoli di Apache Airflow v2.0.2	Python 3.7

Per ulteriori informazioni sulla migrazione delle distribuzioni di Apache Airflow autogestite o sulla migrazione di un ambiente Amazon MWAA esistente, incluse le istruzioni per il backup del database di metadati, consulta la Amazon MWAA Migration Guide.

Componenti Apache Airflow

Questa sezione descrive il numero di scheduler e worker di Apache Airflow disponibili per ogni versione di Apache Airflow su Amazon MWAA e fornisce un elenco delle funzionalità chiave di Apache Airflow, indicando la versione che supporta ciascuna funzionalità.

Pianificatori

Versione Apache Airflow	Scheduler (impostazione predefinita)	Pianificatore (min)	Pianificatore (max)	
Apache Airflow v2 e versioni successive	2	2	5	

Worker

Versione Airflow	Lavoratori (min)	Lavoratori (max)	Lavoratori (impostazione predefinita)	
Apache Airflow v2	1	25	10	

Aggiornamento della versione di Apache Airflow

Amazon MWAA supporta aggiornamenti di versioni minori. Ciò significa che puoi aggiornare il tuo ambiente dalla versione x**.1.**z ax**.2.**z, ma non a una nuova versione principale, ad esempio dalla a. **1.**y.z **2.**y.z

> **Note**
>
> Non puoi effettuare il downgrade della versione di Apache Airflow per il tuo ambiente.

Per ulteriori informazioni e istruzioni dettagliate sull'aggiornamento delle risorse del flusso di lavoro e sull'aggiornamento dell'ambiente a una nuova versione, consulta. the section called “Aggiornamento della versione”

Versioni obsolete di Apache Airflow

La tabella seguente elenca le versioni obsolete di Apache Airflow in Amazon MWAA, insieme alle date di rilascio iniziale e di fine del supporto per ciascuna versione. Per ulteriori informazioni sulla migrazione a una versione più recente, consulta la Amazon MWAA Migration Guide.

Versione Apache Airflow	Data di rilascio di Apache Airflow	Data di disponibilità di Amazon MWAA	Data di supporto limitata per Amazon MWAA	Data di fine del supporto per Amazon MWAA
v1.10.12	25 agosto 2020	24 novembre 2020	21 agosto 2023	21 febbraio 2024
v2.0.2	19 aprile 2021	25 maggio 2021	23 novembre 2023	29 aprile 2024
v2.2.2	15 novembre 2021	27 gennaio 2022	25 gennaio 2024	27 giugno 2024

Supporto e domande frequenti sulla versione di Apache Airflow

In conformità con il processo di rilascio e la politica di versione della community di Apache Airflow, Amazon MWAA si impegna a supportare almeno tre versioni minori di Apache Airflow in qualsiasi momento. Annunceremo la data di fine del supporto per una determinata versione secondaria di Apache Airflow almeno 90 giorni prima della data di fine del supporto.

Domande frequenti

D: Per quanto tempo Amazon MWAA supporta una versione di Apache Airflow?

R: Amazon MWAA supporta una versione secondaria di Apache Airflow per un minimo di 12 mesi dalla prima disponibilità.

D: Ricevo una notifica quando termina il supporto per una versione di Apache Airflow su Amazon MWAA?

R: Sì. Se alcuni ambienti Amazon MWAA del tuo account eseguono la versione prossima alla fine del supporto, Amazon MWAA invia un avviso entro la data di fine del AWS Health Dashboard supporto.

D: Cosa succede nella data di supporto limitata?

R: Alla data di supporto limitata, non è più possibile creare nuovi ambienti Amazon MWAA con la versione associata. Gli ambienti esistenti continueranno a essere disponibili fino alla fine della data di supporto.

D: Cosa succede allo scadere della data di fine supporto?

R: Alla data di fine del supporto, continuerai ad accedere agli ambienti Amazon MWAA esistenti che eseguono la versione obsoleta associata di Apache Airflow a tuo rischio. Per istruzioni sull'aggiornamento a una versione più recente di Apache Airflow su Amazon MWAA, consulta la Amazon MWAA Migration Guide.

Important

È tua responsabilità mantenere aggiornate le versioni di Amazon MWAA. AWS esorta tutti i clienti ad aggiornare i propri ambienti Amazon MWAA alla versione più recente per beneficiare delle più recenti misure di sicurezza, privacy e disponibilità. Se utilizzi il tuo ambiente su una versione o un software non supportato oltre la data di obsolescenza, denominata versione precedente, hai maggiori probabilità di incorrere in rischi operativi, relativi alla sicurezza e alla privacy, inclusi i tempi di inattività. Gestendo il tuo ambiente Amazon MWAA su una versione precedente, confermi di comprendere e di assumerti consapevolmente questi rischi e accetti di completare l'aggiornamento alla versione più recente il prima possibile. Il funzionamento continuato del tuo ambiente su una versione precedente è soggetto all'accordo che disciplina l'uso dei servizi. AWS
Le versioni precedenti non sono considerate generalmente disponibili e AWS non forniscono più supporto per la versione precedente. Di conseguenza, AWS può porre limiti all'accesso

o all'uso di qualsiasi versione precedente in qualsiasi momento, se AWS determina che la versione precedente rappresenta un rischio per la sicurezza o la responsabilità, o un rischio di danno, per i servizi AWS, le sue Affiliate o qualsiasi altra terza parte. La tua decisione di continuare a eseguire i tuoi carichi di lavoro su una versione precedente potrebbe rendere i tuoi contenuti non disponibili, danneggiati o irrecuperabili. Gli ambienti che utilizzano una versione precedente sono soggetti alle eccezioni del Service Level Agreement (SLA).
Gli ambienti e il relativo software in esecuzione su una versione precedente potrebbero contenere bug, errori, difetti e componenti dannosi. Di conseguenza, e nonostante qualsiasi informazione contraria contenuta nel contratto o nei termini di servizio, la versione precedente viene AWS fornita così com'è.
Per ulteriori informazioni sul modello AWS di responsabilità condivisa, vedere Responsabilità condivisa nel AWS Well-Architected Framework.

Endpoint e quote del servizio Amazon Managed Workflows per Apache Airflow

Amazon Managed Workflows for Apache Airflow ha le seguenti quote di servizio ed endpoint. Le quote di servizio, note anche come limiti, sono il numero massimo di risorse o operazioni di servizio per il tuo account. AWS

Indice

- Endpoint di servizio
- Quote del servizio
- Quote crescenti

Endpoint di servizio

Per visualizzare un elenco di endpoint per Amazon MWAA, consulta Amazon Managed Workflows for Apache Airflow endpoint e quote.

Quote del servizio

Nome quota	Descrizione	Quota predefinita	Regolabile
Ambienti	Il numero massimo di ambienti Amazon MWAA per account per regione.	10	Sì
Worker per ambiente	Il numero massimo di lavoratori per ambiente Amazon MWAA.	25	Sì
Server Web per ambiente	Il numero massimo di server Web per ambiente Amazon MWAA.	5	Sì

Quote crescenti

È possibile richiedere un aumento a una quota regolabile inviando una richiesta di aumento della quota.

Domande frequenti su Amazon MWAA

Questa pagina descrive le domande più comuni che potresti incontrare quando usi Amazon Managed Workflows for Apache Airflow.

Indice

- Versioni supportate
  - Cosa supporta Amazon MWAA per Apache Airflow v2?
  - Perché le versioni precedenti di Apache Airflow non sono supportate?
  - Quale versione di Python devo usare?
  - Quale versione di pip Amazon MWAA utilizza?
- Casi d'uso
  - Quando AWS Step Functions devo usare vs. Amazon MWAA?
- Specifiche ambientali
  - Quanto spazio di archiviazione delle attività è disponibile per ogni ambiente?
  - Qual è il sistema operativo predefinito utilizzato per gli ambienti Amazon MWAA?
  - Posso usare un'immagine personalizzata per il mio ambiente Amazon MWAA?
  - Amazon MWAA è conforme alla normativa HIPAA?
  - Amazon MWAA supporta le istanze Spot?
  - Amazon MWAA supporta un dominio personalizzato?
  - Posso accedere al mio ambiente tramite SSH?
  - Perché è richiesta una regola di autoreferenziazione nel gruppo di sicurezza VPC?
  - Posso nascondere ambienti a gruppi diversi in IAM?
  - Posso archiviare dati temporanei su Apache Airflow Worker?
  - Posso specificare più di 25 Apache Airflow Workers?
  - Amazon MWAA supporta Amazon VPC o sottoreti condivise?
- Metriche
  - Quali metriche vengono utilizzate per determinare se scalare Workers?
  - Posso creare metriche personalizzate in? CloudWatch
- DAG, operatori, connessioni e altre domande
  - Posso usare il? PythonVirtualenvOperator

- Quanto tempo impiega Amazon MWAA a riconoscere un nuovo file DAG?
- Perché il mio file DAG non viene prelevato da Apache Airflow?
- Posso rimuovere un plugins.zip or requirements.txt da un ambiente?
- Perché non vedo i miei plugin nel menu Admin Plugins di Apache Airflow v2.0.2?
- Posso usare gli operatori AWS del Database Migration Service (DMS)?

Versioni supportate

Cosa supporta Amazon MWAA per Apache Airflow v2?

Per scoprire cosa supporta Amazon MWAA, consulta. Versioni di Apache Airflow su Amazon Managed Workflows per Apache Airflow

Perché le versioni precedenti di Apache Airflow non sono supportate?

Supportiamo solo la versione più recente (al momento del lancio) di Apache Airflow Apache Airflow v1.10.12 a causa di problemi di sicurezza delle versioni precedenti.

Quale versione di Python devo usare?

Le seguenti versioni di Apache Airflow sono supportate su Amazon Managed Workflows for Apache Airflow.

> **Note**
>
> - A partire da Apache Airflow v2.2.2, Amazon MWAA supporta l'installazione di requisiti Python, pacchetti provider e plug-in personalizzati direttamente sul server Web Apache Airflow.
> - A partire da Apache Airflow v2.7.2, il file dei requisiti deve includere una dichiarazione. `--constraint` Se non fornisci un vincolo, Amazon MWAA te ne specificherà uno per garantire che i pacchetti elencati nei tuoi requisiti siano compatibili con la versione di Apache Airflow che stai utilizzando.
>
>   Per ulteriori informazioni sull'impostazione dei vincoli nel file dei requisiti, consulta Installazione delle dipendenze in Python.

Versione Apache Airflow	Guida Apache Airflow	Vincoli di Apache Airflow	Versione di Python
v2.8.1	Guida di riferimento di Apache Airflow v2.8.1	File dei vincoli di Apache Airflow v2.8.1	Python 3.11
v2.7.2	Guida di riferimento di Apache Airflow v2.7.2	File dei vincoli di Apache Airflow v2.7.2	Python 3.11
v2.6.3	Guida di riferimento di Apache Airflow v2.6.3	File dei vincoli di Apache Airflow v2.6.3	Python 3.10
v2.5.1	Guida di riferimento di Apache Airflow v2.5.1	File dei vincoli di Apache Airflow v2.5.1	Python 3.10
v2.4.3	Guida di riferimento di Apache Airflow v2.4.3	File dei vincoli di Apache Airflow v2.4.3	Python 3.10
v2.2.2	Guida di riferimento di Apache Airflow v2.2.2	File dei vincoli di Apache Airflow v2.2.2	Python 3.7
v2.0.2	Guida di riferimento di Apache Airflow v2.0.2	File dei vincoli di Apache Airflow v2.0.2	Python 3.7

Per ulteriori informazioni sulla migrazione delle distribuzioni di Apache Airflow autogestite o sulla migrazione di un ambiente Amazon MWAA esistente, incluse le istruzioni per il backup del database di metadati, consulta la Amazon MWAA Migration Guide.

Quale versione di **`pip`** Amazon MWAA utilizza?

Per gli ambienti che eseguono Apache Airflow v1.10.12, Amazon MWAA installa la versione 21.1.2. `pip`

> **Note**
>
> Amazon MWAA non eseguirà l'aggiornamento `pip` per gli ambienti Apache Airflow v1.10.12.

Per gli ambienti che eseguono Apache Airflow v2 e versioni successive, Amazon MWAA installa la versione 21.3.1. `pip`

Casi d'uso

Quando AWS Step Functions devo usare vs. Amazon MWAA?

1. È possibile utilizzare Step Functions per elaborare gli ordini dei singoli clienti, poiché Step Functions è in grado di scalare per soddisfare la domanda di un ordine o di un milione di ordini.
2. Se esegui un flusso di lavoro notturno che elabora gli ordini del giorno precedente, puoi utilizzare Step Functions o Amazon MWAA. Amazon MWAA ti offre un'opzione open source per astrarre il flusso di lavoro dalle AWS risorse che stai utilizzando.

Specifiche ambientali

Quanto spazio di archiviazione delle attività è disponibile per ogni ambiente?

Lo spazio di archiviazione delle attività è limitato a 10 GB ed è specificato da Amazon ECS Fargate 1.3. La quantità di RAM è determinata dalla classe di ambiente specificata. Per ulteriori informazioni sulle classi di ambiente, vedereConfigurazione della classe di ambiente Amazon MWAA.

Qual è il sistema operativo predefinito utilizzato per gli ambienti Amazon MWAA?

Gli ambienti Amazon MWAA vengono creati su istanze che eseguono AMI Amazon Linux.

Posso usare un'immagine personalizzata per il mio ambiente Amazon MWAA?

Le immagini personalizzate non sono supportate. Amazon MWAA utilizza immagini basate su AMI Amazon Linux. Amazon MWAA installa i requisiti aggiuntivi eseguendo `pip3 -r install` i requisiti specificati nel file requirements.txt che aggiungi al bucket Amazon S3 per l'ambiente.

Amazon MWAA è conforme alla normativa HIPAA?

Amazon MWAA è idoneo all'Health Insurance Portability and Accountability Act (HIPAA). Se disponi di un HIPAA Business Associate Addendum (BAA) AWS, puoi utilizzare Amazon MWAA per flussi di lavoro che gestiscono Protected Health Information (PHI) in ambienti creati a partire dal 14 novembre 2022.

Amazon MWAA supporta le istanze Spot?

Amazon MWAA attualmente non supporta i tipi di istanze Spot Amazon EC2 on-demand per Apache Airflow. Tuttavia, un ambiente Amazon MWAA può attivare istanze Spot su, ad esempio, Amazon EMR e Amazon EC2.

Amazon MWAA supporta un dominio personalizzato?

Per poter utilizzare un dominio personalizzato per il tuo nome host Amazon MWAA, esegui una delle seguenti operazioni:

- Per le implementazioni Amazon MWAA con accesso pubblico a server Web, puoi utilizzare Amazon con CloudFront Lambda @Edge per indirizzare il traffico verso il tuo ambiente e mappare un nome di dominio personalizzato. CloudFront Per ulteriori informazioni e un esempio di configurazione di un dominio personalizzato per un ambiente pubblico, consulta l'esempio di dominio personalizzato Amazon MWAA per server Web pubblico nell'archivio degli esempi Amazon MWAA. GitHub
- Per le implementazioni Amazon MWAA con accesso privato al server Web, puoi utilizzare un Application Load Balancer (ALB) per indirizzare il traffico verso Amazon MWAA e mappare un nome di dominio personalizzato all'ALB. Per ulteriori informazioni, consulta the section called “Utilizzo di un Load Balancer (avanzato)”.

Posso accedere al mio ambiente tramite SSH?

Sebbene SSH non sia supportato in un ambiente Amazon MWAA, è possibile utilizzare un DAG per eseguire comandi bash utilizzando. `BashOperator` Per esempio:

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago
with DAG(dag_id="any_bash_command_dag", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
```

```
    cli_command = BashOperator(
        task_id="bash_command",
        bash_command="{{ dag_run.conf['command'] }}"
    )
```

Per attivare il DAG nell'interfaccia utente di Apache Airflow, usa:

```
{ "command" : "your bash command"}
```

Perché è richiesta una regola di autoreferenziazione nel gruppo di sicurezza VPC?

Creando una regola di autoreferenziazione, limiti l'origine allo stesso gruppo di sicurezza nel VPC e non è aperta a tutte le reti. Per ulteriori informazioni, vedi the section called “Sicurezza nel tuo VPC”.

Posso nascondere ambienti a gruppi diversi in IAM?

Puoi limitare l'accesso specificando un nome di ambiente in AWS Identity and Access Management, tuttavia, il filtro di visibilità non è disponibile nella AWS console: se un utente può vedere un ambiente, può vedere tutti gli ambienti.

Posso archiviare dati temporanei su Apache Airflow Worker?

I tuoi operatori Apache Airflow possono archiviare dati temporanei sui Workers. Apache Airflow Workers può accedere ai file temporanei `/tmp` nei container Fargate del tuo ambiente.

> **Note**
>
> Lo spazio di archiviazione totale delle attività è limitato a 10 GB, secondo Amazon ECS Fargate 1.3. Non è garantito che le attività successive vengano eseguite sulla stessa istanza del contenitore Fargate, che potrebbe utilizzare una cartella diversa`/tmp`.

Posso specificare più di 25 Apache Airflow Workers?

Sì. Sebbene sia possibile specificare fino a 25 worker Apache Airflow sulla console Amazon MWAA, è possibile configurarne fino a 50 in un ambiente richiedendo un aumento della quota. Per ulteriori informazioni, consulta la sezione Richiesta di un aumento di quota.

Amazon MWAA supporta Amazon VPC o sottoreti condivise?

Amazon MWAA non supporta Amazon VPC o sottoreti condivise di Amazon. L'Amazon VPC selezionato quando crei un ambiente deve essere di proprietà dell'account che sta tentando di creare l'ambiente. Tuttavia, puoi indirizzare il traffico da un Amazon VPC nell'account Amazon MWAA a un VPC condiviso. Per ulteriori informazioni e per vedere un esempio di routing del traffico verso un Amazon VPC condiviso, consulta Routing centralizzato in uscita verso Internet nella Amazon VPC Transit Gateways Guide.

Metriche

Quali metriche vengono utilizzate per determinare se scalare Workers?

Amazon MWAA monitora QueuedTasksand RunningTasksin ingresso CloudWatch per determinare se scalare Apache Airflow Workers nel tuo ambiente. Per ulteriori informazioni, vedi Monitoraggio e metriche.

Posso creare metriche personalizzate in? CloudWatch

Non sulla CloudWatch console. Tuttavia, puoi creare un DAG in cui scrivere metriche personalizzate. CloudWatch Per ulteriori informazioni, consulta the section called “Utilizzo di un DAG per scrivere metriche personalizzate”.

DAG, operatori, connessioni e altre domande

Posso usare il? **`PythonVirtualenvOperator`**

Non `PythonVirtualenvOperator` è supportato in modo esplicito su Amazon MWAA, ma puoi creare un plug-in personalizzato che utilizza il. `PythonVirtualenvOperator` Per il codice di esempio, consulta the section called “Plugin personalizzato per la patchPythonVirtualenvOperator ”.

Quanto tempo impiega Amazon MWAA a riconoscere un nuovo file DAG?

I DAG vengono periodicamente sincronizzati dal bucket Amazon S3 al tuo ambiente. Se aggiungi un nuovo file DAG, Amazon MWAA impiega circa 300 secondi per iniziare a utilizzare il nuovo file. Se aggiorni un DAG esistente, Amazon MWAA impiega circa 30 secondi per riconoscere gli aggiornamenti.

Questi valori, 300 secondi per i nuovi DAG e 30 secondi per gli aggiornamenti dei DAG esistenti, corrispondono rispettivamente alle opzioni di configurazione di Apache Airflow. `dag_dir_list_interval` `min_file_process_interval`

Perché il mio file DAG non viene prelevato da Apache Airflow?

Di seguito sono riportate le possibili soluzioni per questo problema:

1. Verifica che il tuo ruolo di esecuzione disponga di autorizzazioni sufficienti per il tuo bucket Amazon S3. Per ulteriori informazioni, vedi Ruolo di esecuzione di Amazon MWAA.
2. Verifica che nel bucket Amazon S3 sia configurato Block Public Access e che il controllo delle versioni sia abilitato. Per ulteriori informazioni, vedi Creare are are are are are are are are are are are are are are are are are are are are are are are are.
3. Verifica il file DAG stesso. Ad esempio, assicuratevi che ogni DAG abbia un ID DAG univoco.

Posso rimuovere un **`plugins.zip`** or **`requirements.txt`** da un ambiente?

Al momento, non è possibile rimuovere plugins.zip o requirements.txt da un ambiente una volta aggiunti, ma stiamo lavorando al problema. Nel frattempo, una soluzione alternativa consiste nel puntare rispettivamente a un file di testo o zip vuoto. Per ulteriori informazioni, vedi Eliminazione di file su Amazon S3.

Perché non vedo i miei plugin nel menu Admin Plugins di Apache Airflow v2.0.2?

Per motivi di sicurezza, il server Web Apache Airflow su Amazon MWAA ha un accesso di rete limitato e non installa plugin né dipendenze Python direttamente sul server Web Apache Airflow per ambienti versione 2.0.2. Il plug-in mostrato consente ad Amazon MWAA di autenticare gli utenti Apache Airflow in (IAM). AWS Identity and Access Management

Per poter installare plugin e dipendenze Python direttamente sul server web, consigliamo di creare un nuovo ambiente con Apache Airflow v2.2 e versioni successive. Amazon MWAA installa dipendenze Python e plugin personalizzati direttamente sul server Web per Apache Airflow v2.2 e versioni successive.

Posso usare gli operatori AWS del Database Migration Service (DMS)?

Amazon MWAA supporta gli operatori DMS. Tuttavia, questo operatore non può essere utilizzato per eseguire azioni sul database di metadati PostgreSQL di Amazon Aurora associato a un ambiente Amazon MWAA.

Amazon Managed Workflows for Apache Airflow

Questo argomento descrive i problemi e gli errori comuni che potresti riscontrare durante l'utilizzo di Apache Airflow su Amazon Managed Workflow for Apache Airflow e i passaggi consigliati per risolvere questi errori.

Indice

- Risoluzione dei problemi: DAG, operatori, connessioni e altri problemi in Apache Airflow v2
  - Connessioni
    - Non riesco a connettermi a Secrets Manager
    - Come posso configurare le condizioni disecretsmanager:ResourceTag/<tag-key> Secrets Manager o una limitazione delle risorse nella mia politica del ruolo di esecuzione?
    - Non riesco a connettermi a Snowflake
    - Non riesco a vedere la mia connessione nell'interfaccia utente di Airflow
  - Server Web
    - Vedo un errore 5xx durante l'accesso al server Web
    - Viene visualizzato l'errore «Lo scheduler sembra non essere in esecuzione»
  - Processi
    - Vedo che le mie attività sono bloccate o non vengono completate
  - CLI
    - Vedo un errore '503' quando si attiva un DAG nella CLI
    - Perché il comando CLI didags backfill Apache Airflow fallisce? C'è una soluzione alternativa?
  - Operatori
    - Ho ricevuto unPermissionError: [Errno 13] Permission denied errore utilizzando l'operatore S3Transform
- Risoluzione dei problemi: DAG, operatori, connessioni e altri problemi in Apache Airflow v1
  - Aggiornamento di requirements.txt
    - L'aggiuntaapache-airflow-providers-amazon causa il malfunzionamento del mio ambiente
  - DAG rotto
    - Ho ricevuto un errore «Broken DAG» quando utilizzo gli operatori Amazon DynamoDB
    - Ho ricevuto l'errore «Broken DAG: Nessun modulo chiamato psycopg2"

        - Ho ricevuto un errore «Broken DAG» durante l'utilizzo degli operatori Slack
        - Ho ricevuto vari errori durante l'installazione di Google/GCP/BigQuery
        - Ho ricevuto l'errore «Broken DAG: Nessun modulo chiamato Cython»
    - Operatori
        - Ho ricevuto un errore utilizzando l'BigQueryoperatore
    - Connessioni
        - Non riesco a connettermi a Snowflake
        - Non riesco a connettermi a Secrets Manager
        - Non riesco a connettermi al mio server MySQL su '<DB-identifier-name>.cluster-id.<region>.rds.amazonaws.com»
    - Server Web
        - Sto usando ilBigQueryOperator e sta causando il crash del mio server web
        - Vedo un errore 5xx durante l'accesso al server Web
        - Viene visualizzato l'errore «Lo scheduler sembra non essere in esecuzione»
    - Processi
        - Vedo che le mie attività sono bloccate o non vengono completate
    - CLI
        - Vedo un errore '503' quando si attiva un DAG nella CLI
- Risoluzione dei problemi: creazione e aggiornamento di un ambiente Amazon MWAA
    - Aggiornamento degli requirements.txt
        - Ho specificato una nuova versione del miorequirements.txt e sono necessari più di 20 minuti per aggiornare il mio ambiente
    - Plug-in
        - Amazon MWAA supporta l'implementazione dell'interfaccia utente personalizzata?
        - Sono in grado di implementare modifiche personalizzate all'interfaccia utente sul runner locale Amazon MWAA tramite plugin, tuttavia quando provo a fare lo stesso su Amazon MWAA, non vedo le mie modifiche né alcun errore. Perché sta succedendo questo?
    - Creare bucket.
        - Non riesco a selezionare l'opzione per le impostazioni dell'accesso pubblico ai blocchi Amazon S3
    - Creazione dell'ambiente

        - Ho provato a creare un ambiente ed è bloccato nello stato «Creazione»
        - Ho provato a creare un ambiente ma mostra lo stato «Creazione fallita»
        - Ho provato a selezionare un VPC e ho ricevuto un errore «Errore di rete»
        - Ho provato a creare un ambiente e ho ricevuto un errore di servizio, partizione o risorsa «deve essere passato»
        - Ho provato a creare un ambiente e mostra lo stato come «Disponibile», ma quando provo ad accedere all'interfaccia utente di Airflow viene visualizzato un errore «Risposta vuota dal server» o «502 Bad Gateway»
        - Ho provato a creare un ambiente e il mio nome utente è un gruppo di nomi di caratteri casuali
    - Ambiente di aggiornamento
        - Ho provato a cambiare la classe dell'ambiente ma l'aggiornamento non è riuscito
    - Ambiente di accesso
        - Non riesco ad accedere all'interfaccia utente Apache Airflow
- Risoluzione dei problemi: CloudWatch log ed CloudTrail errori
    - Log
        - Non riesco a visualizzare i miei registri delle attività o ho ricevuto l'errore «Leggere il registro remoto da Cloudwatch log_group»
        - Le attività non vanno a buon fine senza alcun registro
        - Vedo un errore '' in ResourceAlreadyExistsException CloudTrail
        - Vedo un errore «Richiesta non valida» in CloudTrail
        - Nei log di Apache Airflow viene visualizzato il messaggio «Impossibile individuare una libreria Oracle Client a 64 bit: «libclntsh.so: impossibile aprire il file o la directory condivisa: nessun file o directory di questo tipo»
        - Vedo psycopg2 «il server ha chiuso la connessione in modo imprevisto» nei miei log di Scheduler
        - Nei miei registri di elaborazione DAG vedo «Executor riporta che l'istanza dell'operazione %s è terminata (%s) anche se l'attività dice che è %s»
        - Vedo 'Impossibile leggere i log remoti da log_group: airflow-* {*EnvironmentName} -Task log_stream: * {*DAG_ID} /* {*TASK_ID} /* {*time} /* {*n} .log. ' nei miei registri delle attività

Risoluzione dei problemi: DAG, operatori, connessioni e altri problemi in Apache Airflow v2

Gli argomenti di questa pagina descrivono le risoluzioni alle dipendenze di Apache Airflow v2 Python, ai plug-in personalizzati, ai DAG, agli operatori, alle connessioni, alle attività e ai problemi del server Web che potresti riscontrare in un ambiente Amazon Managed Workflow for Apache Airflow.

Indice

- Connessioni
  - Non riesco a connettermi a Secrets Manager
  - Come posso configurare le condizioni disecretsmanager:ResourceTag/<tag-key> Secrets Manager o una limitazione delle risorse nella mia politica del ruolo di esecuzione?
  - Non riesco a connettermi a Snowflake
  - Non riesco a vedere la mia connessione nell'interfaccia utente di Airflow
- Server Web
  - Vedo un errore 5xx durante l'accesso al server Web
  - Viene visualizzato l'errore «Lo scheduler sembra non essere in esecuzione»
- Processi
  - Vedo che le mie attività sono bloccate o non vengono completate
- CLI
  - Vedo un errore '503' quando si attiva un DAG nella CLI
  - Perché il comando CLI didags backfill Apache Airflow fallisce? C'è una soluzione alternativa?
- Operatori
  - Ho ricevuto unPermissionError: [Errno 13] Permission denied errore utilizzando l'operatore S3Transform

Connessioni

L'argomento seguente descrive gli errori che potresti ricevere quando usi una connessione Apache Airflow o usi un altroAWS database.

Non riesco a connettermi a Secrets Manager

È consigliabile eseguire le operazioni seguenti:

1. Scopri come creare chiavi segrete per la connessione e le variabili di Apache Airflow inthe section called “Configurazione di Secrets Manager”.
2. Scopri come usare la chiave segreta per una variabile Apache Airflow (`test-variable`) inUtilizzo di una chiave segreta inAWS Secrets Managerper una variabile Apache Airflow.
3. Scopri come usare la chiave segreta per una connessione Apache Airflow (`myconn`) inUtilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow.

Come posso configurare le condizioni di**secretsmanager:ResourceTag/<tag-key>** Secrets Manager o una limitazione delle risorse nella mia politica del ruolo di esecuzione?

> **Note**
>
> Si applica ad Apache Airflow versione 2.0 e precedenti.

Attualmente, non è possibile limitare l'accesso ai segreti di Secrets Manager utilizzando chiavi di condizione o altre restrizioni di risorse nel ruolo di esecuzione del proprio ambiente, a causa di un problema noto in Apache Airflow.

Non riesco a connettermi a Snowflake

È consigliabile eseguire le operazioni seguenti:

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Aggiungi le seguenti voci al file requirements.txt per il tuo ambiente.

```
apache-airflow-providers-snowflake==1.3.0
```

3. Aggiungi le seguenti importazioni al tuo DAG:

```
from airflow.providers.snowflake.operators.snowflake import SnowflakeOperator
```

Assicurati che l'oggetto di connessione Apache Airflow includa le seguenti coppie chiave-valore:

1. ID di connessione: snowflake_conn

2. Tipo di connettore: Snowflake
3. Ospite:<my account>. <my region if not us-west-2>.snowflakecomputing.com
4. Schema: <my schema>
5. Accedi: <my user name>
6. Password: ********
7. Porta: <port, if any>
8. Supplementare:

```
{
    "account": "<my account>",
    "warehouse": "<my warehouse>",
    "database": "<my database>",
    "region": "<my region if not using us-west-2 otherwise omit this line>"
}
```

Ad esempio:

```
>>> import json
>>> from airflow.models.connection import Connection
>>> myconn = Connection(
...    conn_id='snowflake_conn',
...    conn_type='Snowflake',
...    host='YOUR_ACCOUNT.YOUR_REGION.snowflakecomputing.com',
...    schema='YOUR_SCHEMA'
...    login='YOUR_USERNAME',
...    password='YOUR_PASSWORD',
...    port='YOUR_PORT'
...    extra=json.dumps(dict(account='YOUR_ACCOUNT', warehouse='YOUR_WAREHOUSE',
 database='YOUR_DB_OPTION', region='YOUR_REGION')),
... )
```

Non riesco a vedere la mia connessione nell'interfaccia utente di Airflow

Apache Airflow fornisce modelli di connessione nell'interfaccia utente di Apache Airflow. Lo usa per generare la stringa URI di connessione, indipendentemente dal tipo di connessione. Se un modello di connessione non è disponibile nell'interfaccia utente di Apache Airflow, è possibile utilizzare un modello di connessione alternativo per generare una stringa URI di connessione, ad esempio utilizzando il modello di connessione HTTP.

È consigliabile eseguire le operazioni seguenti:

1. Visualizza i tipi di connessione forniti da Amazon MWAA nell'interfaccia utente di Apache Airflow all'indirizzoPacchetti del provider Apache Airflow installati in ambienti Amazon MWAA.
2. Visualizza i comandi per creare una connessione Apache Airflow nella CLI all'indirizzoRiferimento ai comandi CLI di Apache Airflow.
3. Scopri come utilizzare i modelli di connessione nell'interfaccia utente di Apache Airflow in modo intercambiabile per i tipi di connessione che non sono disponibili nell'interfaccia utente di Apache Airflow su Amazon MWAA all'indirizzoPanoramica dei tipi di connessione.

Server Web

L'argomento seguente descrive gli errori che potresti ricevere per il tuo server Web Apache Airflow su Amazon MWAA.

Vedo un errore 5xx durante l'accesso al server Web

È consigliabile eseguire le operazioni seguenti:

1. Controlla le opzioni di configurazione di Apache Airflow. Verifica che le coppie chiave-valore che hai specificato come opzione di configurazione di Apache Airflow, ad esempioAWS Secrets Manager, siano state configurate correttamente. Per ulteriori informazioni, consulta the section called “Non riesco a connettermi a Secrets Manager”.
2. Controlla il`requirements.txt`. Verifica che il pacchetto «extra» di Airflow e le altre librerie elencate nel tuo`requirements.txt` siano compatibili con la tua versione di Apache Airflow.
3. Esplora i modi per specificare le dipendenze Python in un`requirements.txt` file, vediGestione delle dipendenze Python in requirements.txt.

Viene visualizzato l'errore «Lo scheduler sembra non essere in esecuzione»

Se lo scheduler non sembra essere in esecuzione o l'ultimo «battito cardiaco» è stato ricevuto diverse ore fa, i DAG potrebbero non essere visualizzati in Apache Airflow e le nuove attività non verranno pianificate.

È consigliabile eseguire le operazioni seguenti:

1. Verifica che il tuo gruppo di sicurezza VPC consenta l'accesso in entrata alla porta`5432`. Questa porta è necessaria per connettersi al database di metadati Amazon Aurora PostgreSQL per il tuo

ambiente. Dopo aver aggiunto questa regola, concedi ad Amazon MWAA alcuni minuti e l'errore dovrebbe scomparire. Per ulteriori informazioni, consulta the section called “Sicurezza nel tuo VPC”.

> **Note**
>
> - Il metadatabase Aurora PostgreSQL fa parte dell'architettura del servizio Amazon MWAA e non è visibile nel tuoAccount AWS.
> - Gli errori relativi al database sono in genere un sintomo di un errore dello scheduler e non la causa principale.

2. Se lo scheduler non è in esecuzione, potrebbe essere dovuto a una serie di fattori come errori di installazione delle dipendenze o un programma di pianificazione sovraccarico. Verifica che i DAG, i plugin e i requisiti funzionino correttamente visualizzando i gruppi di log corrispondenti inCloudWatch Logs. Per ulteriori informazioni, consulta Monitoraggio e metriche.

Processi

L'argomento seguente descrive gli errori che potresti ricevere per le attività di Apache Airflow in un ambiente.

Vedo che le mie attività sono bloccate o non vengono completate

Se le tue attività di Apache Airflow sono «bloccate» o non vengono completate, ti consigliamo i seguenti passaggi:

1. È possibile che sia definito un numero elevato di DAG. Riduci il numero di DAG ed esegui un aggiornamento dell'ambiente (ad esempio modificando un livello di registro) per forzare il ripristino.

    a. Airflow analizza i DAG indipendentemente dal fatto che siano abilitati o meno. Se stai utilizzando più del 50% della capacità del tuo ambiente, potresti iniziare a sovraccaricare l'Apache Airflow Scheduler. Ciò comporta tempi di analisi totali elevati nelleCloudWatch metriche o lunghi tempi di elaborazione DAG neiCloudWatch log. Esistono altri modi per ottimizzare le configurazioni di Apache Airflow che non rientrano nell'ambito di questa guida.

b. Per ulteriori informazioni sulle best practice che consigliamo per ottimizzare le prestazioni del tuo ambiente, consultathe section called “Ottimizzazione delle prestazioni per Apache Airflow”.

2. È possibile che ci sia un gran numero di attività in coda. Questo appare spesso come un numero elevato e crescente di attività nello stato «Nessuna» o come un numero elevato in Attività in coda e/o Attività in sospesoCloudWatch. Questo può accadere per i seguenti motivi:

   a. Se ci sono più attività da eseguire rispetto alla capacità dell'ambiente e/o un numero elevato di attività messe in coda prima della scalabilità automatica ha il tempo di rilevare le attività e distribuire Worker aggiuntivi.

   b. Se ci sono più attività da eseguire di quante un ambiente sia in grado di eseguire, si consiglia di ridurre il numero di attività eseguite contemporaneamente dai DAG e/o di aumentare il numero minimo di Apache Airflow Worker.

   c. Se ci sono molte attività in coda prima che la scalabilità automatica abbia avuto il tempo di rilevare e distribuire altri lavoratori, consigliamo di scaglionare la distribuzione delle attività e/o aumentare il numero minimo di Apache Airflow Workers.

   d. È possibile utilizzare il comando update-environment inAWS Command Line Interface (AWS CLI) per modificare il numero minimo o massimo di Worker eseguiti nel proprio ambiente.

   ```
   aws mwaa update-environment --name MyEnvironmentName --min-workers 2 --max-workers 10
   ```

   e. Per ulteriori informazioni sulle best practice che consigliamo per ottimizzare le prestazioni del tuo ambiente, consultathe section called “Ottimizzazione delle prestazioni per Apache Airflow”.

3. È possibile che alcune attività vengano eliminate durante l'esecuzione che appaiono come registri delle attività che si interrompono senza ulteriori indicazioni in Apache Airflow. Questo può accadere per i seguenti motivi:

   a. Se c'è un breve momento in cui 1) le attività correnti superano la capacità attuale dell'ambiente, seguite da 2) alcuni minuti senza esecuzione o in coda, quindi 3) nuove attività in coda.

   b. La scalabilità automatica di Amazon MWAA reagisce al primo scenario aggiungendo altri lavoratori. Nel secondo scenario, rimuove i lavoratori aggiuntivi. Alcune delle attività in coda possono comportare la rimozione dei lavoratori e terminare quando il contenitore viene eliminato.

c. Ti consigliamo di aumentare il numero minimo di lavoratori nel tuo ambiente. Un'altra opzione è modificare la tempistica dei DAG e delle attività per garantire che questi scenari non si verifichino.

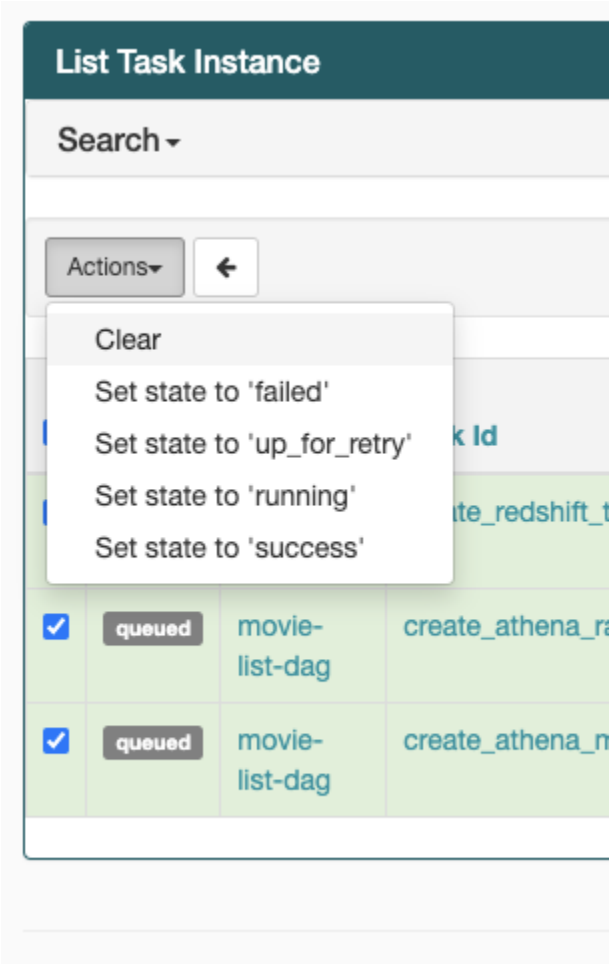
d. Puoi anche impostare il numero minimo di lavoratori pari al numero massimo di lavoratori nel tuo ambiente, disattivando in modo efficace la scalabilità automatica. Usa il comando update-environment inAWS Command Line Interface (AWS CLI) per disabilitare la scalabilità automatica impostando lo stesso numero minimo e massimo di lavoratori.

```
aws mwaa update-environment --name MyEnvironmentName --min-workers 5 --max-
workers 5
```

e. Per ulteriori informazioni sulle best practice che consigliamo per ottimizzare le prestazioni del tuo ambiente, consultathe section called “Ottimizzazione delle prestazioni per Apache Airflow”.

4. Se le tue attività sono bloccate nello stato «in esecuzione», puoi anche cancellarle o contrassegnarle come riuscite o non riuscite. Ciò consente al componente di scalabilità automatica del tuo ambiente di ridurre il numero di lavoratori in esecuzione nell'ambiente. La seguente immagina mostra un esempio di operazione bloccata.

- Scegli il cerchio per l'attività bloccata, quindi seleziona Cancella (come mostrato). Ciò consente ad Amazon MWAA di ridurre il numero di dipendenti; in caso contrario, Amazon MWAA non può determinare quali DAG sono abilitati o disattivati e non può ridimensionare se ci sono ancora attività in coda.



5. Scopri di più sul ciclo di vita delle attività di Apache Airflow in Concepts nella guida di riferimento di Apache Airflow.

CLI

L'argomento seguente descrive gli errori che potresti ricevere quando esegui i comandi della CLI di Airflow inAWS Command Line Interface.

Vedo un errore '503' quando si attiva un DAG nella CLI

L'interfaccia a riga di comando di Airflow viene eseguita sul server Web Apache Airflow, che ha una concorrenza limitata. In genere è possibile eseguire contemporaneamente un massimo di 4 comandi CLI.

Perché il comando CLI di**`dags backfill`** Apache Airflow fallisce? C'è una soluzione alternativa?

> **Note**
>
> Quanto segue si applica solo agli ambienti Apache Airflow v2.0.2.

Il`backfill` comando, come altri comandi CLI di Apache Airflow, analizza tutti i DAG localmente prima che i DAG vengano elaborati, indipendentemente dal DAG a cui si applica l'operazione CLI. Negli ambienti Amazon MWAA che utilizzano Apache Airflow v2.0.2, poiché i plug-in e i requisiti non sono ancora installati sul server Web al momento dell'esecuzione del comando CLI, l'operazione di analisi fallisce e l'`backfill`operazione non viene richiamata. Se non aveste requisiti né plugin nel vostro ambiente, l'`backfill`operazione avrebbe esito positivo.

Per poter eseguire il comando`backfill` CLI, consigliamo di richiamarlo in un operatore bash. In un operatore bash,`backfill` viene avviato dal lavoratore, consentendo ai DAG di analizzare correttamente quando tutti i requisiti e i plugin necessari sono disponibili e installati. L'esempio seguente mostra come è possibile creare un DAG con un`BashOperator` da eseguire`backfill`.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

with DAG(dag_id="backfill_dag", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
    cli_command = BashOperator(
        task_id="bash_command",
        bash_command="airflow dags backfill my_dag_id"
    )
```

Operatori

L'argomento seguente descrive gli errori che potresti ricevere quando usi gli operatori.

Ho ricevuto un**`PermissionError: [Errno 13] Permission denied`** errore utilizzando l'operatore S3Transform

Ti consigliamo i seguenti passaggi se stai tentando di eseguire uno script di shell con l'operatore S3Transform e ricevi un`PermissionError: [Errno 13] Permission denied` errore. I

passaggi seguenti presuppongono che tu abbia un file plugins.zip esistente. Se stai creando un nuovo plugins.zip, vediInstallazione di plugin personalizzati.

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Crea il tuo script di «trasformazione».

```
#!/bin/bash
cp $1 $2
```

3. (opzionale) Gli utenti macOS e Linux potrebbero dover eseguire il seguente comando per assicurarsi che lo script sia eseguibile.

```
chmod 777 transform_test.sh
```

4. Aggiungi lo script al tuo file plugins.zip.

```
zip plugins.zip transform_test.sh
```

5. Segui i passaggi indicati in Caricare il file plugins.zip su Amazon S3.
6. Segui i passaggi indicati in Specificare la versione di plugins.zip sulla console Amazon MWAA.
7. Utilizza i seguenti file DAG.

```
from airflow import DAG
from airflow.providers.amazon.aws.operators.s3_file_transform import
 S3FileTransformOperator
from airflow.utils.dates import days_ago
import os

DAG_ID = os.path.basename(__file__).replace(".py", "")

with DAG (dag_id=DAG_ID, schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
    file_transform = S3FileTransformOperator(
        task_id='file_transform',
        transform_script='/usr/local/airflow/plugins/transform_test.sh',
        source_s3_key='s3://YOUR_S3_BUCKET/files/input.txt',
        dest_s3_key='s3://YOUR_S3_BUCKET/files/output.txt'
    )
```

8. Segui i passaggi descritti in Caricare il codice DAG su Amazon S3.

Risoluzione dei problemi: DAG, operatori, connessioni e altri problemi in Apache Airflow v1

Gli argomenti di questa pagina contengono le risoluzioni alle dipendenze di Apache Airflow v1.10.12 Python, ai plug-in personalizzati, ai DAG, agli operatori, alle connessioni, alle attività e ai problemi del server Web che potresti riscontrare in un ambiente Amazon Managed Workflows for Apache Airflow.

Indice

- Aggiornamento di requirements.txt
  - L'aggiuntaapache-airflow-providers-amazon causa il malfunzionamento del mio ambiente
- DAG rotto
  - Ho ricevuto un errore «Broken DAG» quando utilizzo gli operatori Amazon DynamoDB
  - Ho ricevuto l'errore «Broken DAG: Nessun modulo chiamato psycopg2"
  - Ho ricevuto un errore «Broken DAG» durante l'utilizzo degli operatori Slack
  - Ho ricevuto vari errori durante l'installazione di Google/GCP/BigQuery
  - Ho ricevuto l'errore «Broken DAG: Nessun modulo chiamato Cython»
- Operatori
  - Ho ricevuto un errore utilizzando l'BigQueryoperatore
- Connessioni
  - Non riesco a connettermi a Snowflake
  - Non riesco a connettermi a Secrets Manager
  - Non riesco a connettermi al mio server MySQL su '<DB-identifier-name>.cluster-id.<region>.rds.amazonaws.com»
- Server Web
  - Sto usando ilBigQueryOperator e sta causando il crash del mio server web
  - Vedo un errore 5xx durante l'accesso al server Web
  - Viene visualizzato l'errore «Lo scheduler sembra non essere in esecuzione»
- Processi
  - Vedo che le mie attività sono bloccate o non vengono completate
- CLI
  - Vedo un errore '503' quando si attiva un DAG nella CLI

Aggiornamento di requirements.txt

L'argomento seguente descrive gli errori che potresti ricevere durante l'aggiornamento del tuo`requirements.txt`.

L'aggiunta**`apache-airflow-providers-amazon`** causa il malfunzionamento del mio ambiente

`apache-airflow-providers-`**`xyz`**è compatibile solo con Apache Airflow v2. `apache-airflow-backport-providers-`**`xyz`**è compatibile con Apache Airflow 1.10.12.

DAG rotto

L'argomento seguente descrive gli errori che potresti ricevere durante l'esecuzione dei DAG.

Ho ricevuto un errore «Broken DAG» quando utilizzo gli operatori Amazon DynamoDB

È consigliabile eseguire le operazioni seguenti:

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Aggiungi il seguente pacchetto al tuo`requirements.txt`.

   ```
   boto
   ```
3. Esplora i modi per specificare le dipendenze Python in un`requirements.txt` file, vediGestione delle dipendenze Python in requirements.txt.

Ho ricevuto l'errore «Broken DAG: Nessun modulo chiamato psycopg2"

È consigliabile eseguire le operazioni seguenti:

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Aggiungi quanto segue al tuo`requirements.txt` con la tua versione di Apache Airflow. Ad esempio:

   ```
   apache-airflow[postgres]==1.10.12
   ```

3. Esplora i modi per specificare le dipendenze Python in un`requirements.txt` file, vediGestione delle dipendenze Python in requirements.txt.

Ho ricevuto un errore «Broken DAG» durante l'utilizzo degli operatori Slack

È consigliabile eseguire le operazioni seguenti:

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Aggiungi il seguente pacchetto al tuo`requirements.txt` e specifica la tua versione di Apache Airflow. Ad esempio:

```
apache-airflow[slack]==1.10.12
```

3. Esplora i modi per specificare le dipendenze Python in un`requirements.txt` file, vediGestione delle dipendenze Python in requirements.txt.

Ho ricevuto vari errori durante l'installazione di Google/GCP/BigQuery

Amazon MWAA utilizza Amazon Linux che richiede una versione specifica di Cython e delle librerie di crittografia. È consigliabile eseguire le operazioni seguenti:

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Aggiungi il seguente pacchetto al tuo`requirements.txt`.

```
grpcio==1.27.2
cython==0.29.21
pandas-gbq==0.13.3
cryptography==3.3.2
apache-airflow-backport-providers-amazon[google]
```

3. Se non utilizzi provider di backport, puoi utilizzare:

```
grpcio==1.27.2
cython==0.29.21
pandas-gbq==0.13.3
cryptography==3.3.2
apache-airflow[gcp]==1.10.12
```

4. Esplora i modi per specificare le dipendenze Python in un`requirements.txt` file, vediGestione delle dipendenze Python in requirements.txt.

Ho ricevuto l'errore «Broken DAG: Nessun modulo chiamato Cython»

Amazon MWAA utilizza Amazon Linux che richiede una versione specifica di Cython. È consigliabile eseguire le operazioni seguenti:

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Aggiungi il seguente pacchetto al tuo`requirements.txt`.

```
cython==0.29.21
```

3. Le librerie Cython hanno varie versioni di dipendenza pip richieste. Ad esempio, utilizzando require`awswrangler==2.4.0`pyarrow<3.1.0,>=2.0.0, pip3 tenta di eseguire l'installazione, il`pyarrow==3.0.0` che causa un errore Broken DAG. Si consiglia di specificare esplicitamente la versione più vecchia accettabile. Ad esempio, se si specifica il valore minimo`pyarrow==2.0.0` prima`awswrangler==2.4.0`, l'errore scompare e l'`requirements.txt`installazione viene eseguita correttamente. I requisiti finali dovrebbero apparire come segue:

```
cython==0.29.21
pyarrow==2.0.0
awswrangler==2.4.0
```

4. Esplora i modi per specificare le dipendenze Python in un`requirements.txt` file, vediGestione delle dipendenze Python in requirements.txt.

Operatori

L'argomento seguente descrive gli errori che potresti ricevere quando usi gli operatori.

Ho ricevuto un errore utilizzando l'BigQueryoperatore

Amazon MWAA non supporta gli operatori con estensioni dell'interfaccia utente. È consigliabile eseguire le operazioni seguenti:

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Una soluzione alternativa consiste nell'ignorare l'estensione aggiungendo una riga nel DAG da impostare`<operator name>.operator_extra_links = None` dopo aver importato gli operatori problematici. Ad esempio:

```
from airflow.contrib.operators.bigquery_operator import BigQueryOperator
BigQueryOperator.operator_extra_links = None
```

3. Puoi utilizzare questo approccio per tutti i DAG aggiungendo quanto sopra a un plugin. Per un esempio, consulta the section called “Plugin personalizzato per la patchPythonVirtualenvOperator ”.

Connessioni

L'argomento seguente descrive gli errori che potresti ricevere quando usi una connessione Apache Airflow o usi un altroAWS database.

Non riesco a connettermi a Snowflake

È consigliabile eseguire le operazioni seguenti:

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Aggiungi le seguenti voci a requirements.txt per il tuo ambiente.

```
asn1crypto == 0.24.0
snowflake-connector-python == 1.7.2
```

3. Aggiungi le seguenti importazioni al tuo DAG:

```
from airflow.contrib.hooks.snowflake_hook import SnowflakeHook
from airflow.contrib.operators.snowflake_operator import SnowflakeOperator
```

Assicurati che l'oggetto di connessione Apache Airflow includa le seguenti coppie chiave-valore:

1. ID di connessione: snowflake_conn
2. Tipo di connettore: Snowflake

3. Ospite:<my account>. <my region if not us-west-2>.snowflakecomputing.com
4. Schema: <my schema>
5. Accedi: <my user name>
6. Password: ********
7. Porta: <port, if any>
8. Extra:

```
{
    "account": "<my account>",
    "warehouse": "<my warehouse>",
    "database": "<my database>",
    "region": "<my region if not using us-west-2 otherwise omit this line>"
}
```

Ad esempio:

```
>>> import json
>>> from airflow.models.connection import Connection
>>> myconn = Connection(
...    conn_id='snowflake_conn',
...    conn_type='Snowflake',
...    host='YOUR_ACCOUNT.YOUR_REGION.snowflakecomputing.com',
...    schema='YOUR_SCHEMA'
...    login='YOUR_USERNAME',
...    password='YOUR_PASSWORD',
...    port='YOUR_PORT'
...    extra=json.dumps(dict(account='YOUR_ACCOUNT', warehouse='YOUR_WAREHOUSE',
 database='YOUR_DB_OPTION', region='YOUR_REGION')),
... )
```

Non riesco a connettermi a Secrets Manager

È consigliabile eseguire le operazioni seguenti:

1. Scopri come creare chiavi segrete per la connessione e le variabili di Apache Airflow inthe section called “Configurazione di Secrets Manager”.
2. Scopri come usare la chiave segreta per una variabile Apache Airflow (`test-variable`) inUtilizzo di una chiave segreta inAWS Secrets Managerper una variabile Apache Airflow.

3. Scopri come usare la chiave segreta per una connessione Apache Airflow (`myconn`) in[Utilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow](#).

Non riesco a connettermi al mio server MySQL su '<DB-identifier-name>.cluster-id.<region>.rds.amazonaws.com»

Il gruppo di sicurezza di Amazon MWAA e il gruppo di sicurezza RDS necessitano di una regola di ingresso per consentire il traffico da e verso l'altro. È consigliabile eseguire le operazioni seguenti:

1. Modifica il gruppo di sicurezza RDS per consentire tutto il traffico proveniente dal gruppo di sicurezza VPC di Amazon MWAA.
2. Modifica il gruppo di sicurezza VPC di Amazon MWAA per consentire tutto il traffico dal gruppo di sicurezza RDS.
3. Riesegui nuovamente le attività e verifica se la query SQL ha avuto successo controllando i log di Apache Airflow inCloudWatch Logs.

Server Web

L'argomento seguente descrive gli errori che potresti ricevere per il tuo server Web Apache Airflow su Amazon MWAA.

Sto usando ilBigQueryOperator e sta causando il crash del mio server web

È consigliabile eseguire le operazioni seguenti:

1. Gli operatori di Apache Airflow come`BigQueryOperator` e`QuboleOperator` che contengono`operator_extra_links` potrebbero causare l'arresto anomalo del server web Apache Airflow. Questi operatori tentano di caricare codice sul tuo server web, cosa non consentita per motivi di sicurezza. Ti consigliamo di applicare le patch agli operatori nel tuo DAG aggiungendo il seguente codice dopo le dichiarazioni di importazione:

   ```
   BigQueryOperator.operator_extra_links = None
   ```

2. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione [aws-mwaa-local-runner](#)onGitHub.

Vedo un errore 5xx durante l'accesso al server Web

È consigliabile eseguire le operazioni seguenti:

1. Controlla le opzioni di configurazione di Apache Airflow. Verifica che le coppie chiave-valore che hai specificato come opzione di configurazione di Apache Airflow, ad esempioAWS Secrets Manager, siano state configurate correttamente. Per ulteriori informazioni, consulta the section called “Non riesco a connettermi a Secrets Manager”.
2. Controlla il`requirements.txt`. Verifica che il pacchetto «extra» di Airflow e le altre librerie elencate nel tuo`requirements.txt` siano compatibili con la tua versione di Apache Airflow.
3. Esplora i modi per specificare le dipendenze Python in un`requirements.txt` file, vediGestione delle dipendenze Python in requirements.txt.

Viene visualizzato l'errore «Lo scheduler sembra non essere in esecuzione»

Se lo scheduler non sembra essere in esecuzione o l'ultimo «battito cardiaco» è stato ricevuto diverse ore fa, i DAG potrebbero non essere visualizzati in Apache Airflow e le nuove attività non verranno pianificate.

È consigliabile eseguire le operazioni seguenti:

1. Verifica che il tuo gruppo di sicurezza VPC consenta l'accesso in entrata alla porta`5432`. Questa porta è necessaria per connettersi al database di metadati Amazon Aurora PostgreSQL per il tuo ambiente. Dopo aver aggiunto questa regola, concedi ad Amazon MWAA alcuni minuti e l'errore dovrebbe scomparire. Per ulteriori informazioni, consulta the section called “Sicurezza nel tuo VPC”.

   > **Note**
   >
   > - Il metadatabase Aurora PostgreSQL fa parte dell'architettura del servizio Amazon MWAA e non è visibile nel tuoAccount AWS.
   > - Gli errori relativi al database sono in genere un sintomo di un errore dello scheduler e non la causa principale.

2. Se lo scheduler non è in esecuzione, potrebbe essere dovuto a una serie di fattori come errori di installazione delle dipendenze o un programma di pianificazione sovraccarico. Verifica che i

DAG, i plugin e i requisiti funzionino correttamente visualizzando i gruppi di log corrispondenti inCloudWatch Logs. Per ulteriori informazioni, consulta Monitoraggio e metriche.

Processi

L'argomento seguente descrive gli errori che potresti ricevere per le attività di Apache Airflow in un ambiente.

Vedo che le mie attività sono bloccate o non vengono completate

Se le tue attività di Apache Airflow sono «bloccate» o non vengono completate, ti consigliamo i seguenti passaggi:

1. È possibile che sia definito un numero elevato di DAG. Riduci il numero di DAG ed esegui un aggiornamento dell'ambiente (ad esempio modificando un livello di registro) per forzare il ripristino.

   a. Airflow analizza i DAG indipendentemente dal fatto che siano abilitati o meno. Se stai utilizzando più del 50% della capacità del tuo ambiente, potresti iniziare a sovraccaricare l'Apache Airflow Scheduler. Ciò comporta un tempo totale di analisi elevato nelleCloudWatch metriche o lunghi tempi di elaborazione DAG neiCloudWatch log. Esistono altri modi per ottimizzare le configurazioni di Apache Airflow che non rientrano nell'ambito di questa guida.

   b. Per ulteriori informazioni sulle best practice che consigliamo per ottimizzare le prestazioni del tuo ambiente, consultathe section called “Ottimizzazione delle prestazioni per Apache Airflow”.

2. È possibile che ci sia un gran numero di attività in coda. Questo appare spesso come un numero elevato e crescente di attività nello stato «Nessuna» o come un numero elevato in Attività in coda e/o Attività in sospesoCloudWatch. Questo può accadere per i seguenti motivi:

   a. Se ci sono più attività da eseguire rispetto alla capacità dell'ambiente e/o un numero elevato di attività messe in coda prima della scalabilità automatica ha il tempo di rilevare le attività e distribuire Worker aggiuntivi.

   b. Se ci sono più attività da eseguire di quante un ambiente sia in grado di eseguire, si consiglia di ridurre il numero di attività eseguite contemporaneamente dai DAG e/o di aumentare il numero minimo di Apache Airflow Worker.

c. Se ci sono molte attività in coda prima che la scalabilità automatica abbia avuto il tempo di rilevare e distribuire altri lavoratori, consigliamo di scaglionare la distribuzione delle attività e/o aumentare il numero minimo di Apache Airflow Workers.

d. È possibile utilizzare il comando update-environment inAWS Command Line Interface (AWS CLI) per modificare il numero minimo o massimo di Worker eseguiti nel proprio ambiente.

```
aws mwaa update-environment --name MyEnvironmentName --min-workers 2 --max-
workers 10
```

e. Per ulteriori informazioni sulle best practice che consigliamo per ottimizzare le prestazioni del tuo ambiente, consultathe section called “Ottimizzazione delle prestazioni per Apache Airflow”.

3. È possibile che alcune attività vengano eliminate durante l'esecuzione che appaiono come registri delle attività che si interrompono senza ulteriori indicazioni in Apache Airflow. Questo può accadere per i seguenti motivi:

a. Se c'è un breve momento in cui 1) le attività correnti superano la capacità attuale dell'ambiente, seguite da 2) alcuni minuti senza esecuzione o in coda, quindi 3) nuove attività in coda.

b. La scalabilità automatica di Amazon MWAA reagisce al primo scenario aggiungendo altri lavoratori. Nel secondo scenario, rimuove i lavoratori aggiuntivi. Alcune delle attività in coda possono comportare la rimozione dei lavoratori e terminare quando il contenitore viene eliminato.

c. Ti consigliamo di aumentare il numero minimo di lavoratori nel tuo ambiente. Un'altra opzione è modificare la tempistica dei DAG e delle attività per garantire che questi scenari non si verifichino.

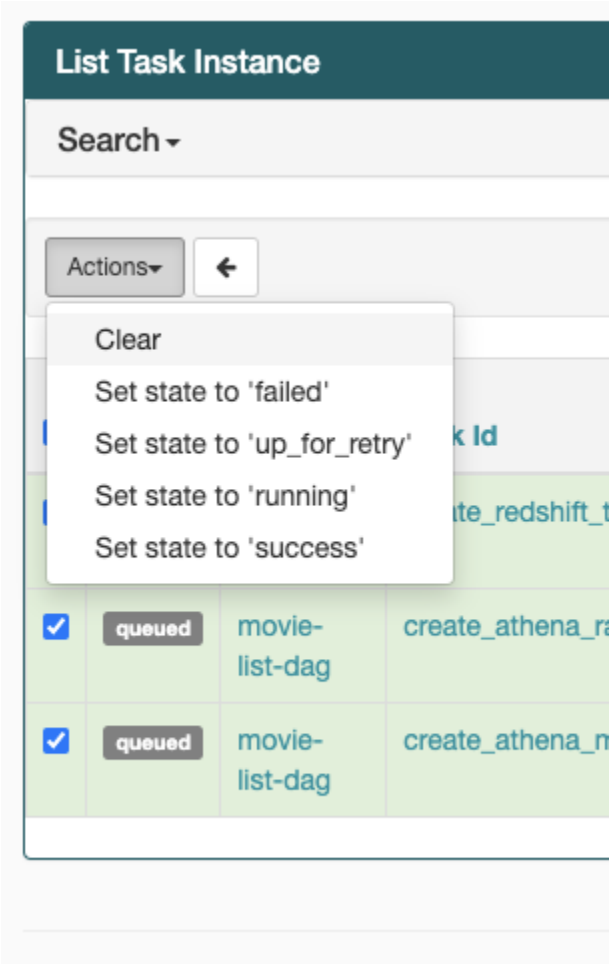
d. Puoi anche impostare il numero minimo di lavoratori pari al numero massimo di lavoratori nel tuo ambiente, disattivando in modo efficace la scalabilità automatica. Usa il comando update-environment inAWS Command Line Interface (AWS CLI) per disabilitare la scalabilità automatica impostando lo stesso numero minimo e massimo di lavoratori.

```
aws mwaa update-environment --name MyEnvironmentName --min-workers 5 --max-
workers 5
```

e. Per ulteriori informazioni sulle best practice che consigliamo per ottimizzare le prestazioni del tuo ambiente, consultathe section called “Ottimizzazione delle prestazioni per Apache Airflow”.

4. Se le tue attività sono bloccate nello stato «in esecuzione», puoi anche cancellarle o contrassegnarle come riuscite o non riuscite. Ciò consente al componente di scalabilità automatica del tuo ambiente di ridurre il numero di lavoratori in esecuzione nell'ambiente. La seguente immagina mostra un esempio di operazione bloccata.

   - Scegli il cerchio per l'attività bloccata, quindi seleziona Cancella (come mostrato). Ciò consente ad Amazon MWAA di ridurre il numero di dipendenti; in caso contrario, Amazon MWAA non può determinare quali DAG sono abilitati o disattivati e non può ridimensionare se ci sono ancora attività in coda.



5. Scopri di più sul ciclo di vita delle attività di Apache Airflow in Concepts nella guida di riferimento di Apache Airflow.

CLI

L'argomento seguente descrive gli errori che potresti ricevere quando esegui i comandi della CLI di Airflow inAWS Command Line Interface.

Vedo un errore '503' quando si attiva un DAG nella CLI

L'interfaccia a riga di comando di Airflow viene eseguita sul server Web Apache Airflow, che ha una concorrenza limitata. In genere è possibile eseguire contemporaneamente un massimo di 4 comandi CLI.

Risoluzione dei problemi: creazione e aggiornamento di un ambiente Amazon MWAA

Gli argomenti di questa pagina contengono gli errori che potresti riscontrare durante la creazione e l'aggiornamento di un ambiente Amazon Managed Workflows for Apache Airflow e come risolverli.

Indice

- Aggiornamento degli requirements.txt
  - Ho specificato una nuova versione del miorequirements.txt e sono necessari più di 20 minuti per aggiornare il mio ambiente
- Plug-in
  - Amazon MWAA supporta l'implementazione dell'interfaccia utente personalizzata?
  - Sono in grado di implementare modifiche personalizzate all'interfaccia utente sul runner locale Amazon MWAA tramite plugin, tuttavia quando provo a fare lo stesso su Amazon MWAA, non vedo le mie modifiche né alcun errore. Perché sta succedendo questo?
- Creare bucket.
  - Non riesco a selezionare l'opzione per le impostazioni dell'accesso pubblico ai blocchi Amazon S3
- Creazione dell'ambiente
  - Ho provato a creare un ambiente ed è bloccato nello stato «Creazione»
  - Ho provato a creare un ambiente ma mostra lo stato «Creazione fallita»
  - Ho provato a selezionare un VPC e ho ricevuto un errore «Errore di rete»
  - Ho provato a creare un ambiente e ho ricevuto un errore di servizio, partizione o risorsa «deve essere passato»

  - Ho provato a creare un ambiente e mostra lo stato come «Disponibile», ma quando provo ad accedere all'interfaccia utente di Airflow viene visualizzato un errore «Risposta vuota dal server» o «502 Bad Gateway»
  - Ho provato a creare un ambiente e il mio nome utente è un gruppo di nomi di caratteri casuali
- Ambiente di aggiornamento
  - Ho provato a cambiare la classe dell'ambiente ma l'aggiornamento non è riuscito
- Ambiente di accesso
  - Non riesco ad accedere all'interfaccia utente Apache Airflow

Aggiornamento degli **`requirements.txt`**

L'argomento seguente descrive gli errori che potresti ricevere durante l'aggiornamento del tuo`requirements.txt`.

Ho specificato una nuova versione del mio**`requirements.txt`** e sono necessari più di 20 minuti per aggiornare il mio ambiente

Se l'ambiente impiega più di venti minuti per installare una nuova versione di un`requirements.txt` file, l'aggiornamento dell'ambiente non è riuscito e Amazon MWAA sta tornando all'ultima versione stabile dell'immagine del contenitore.

1. Controlla le versioni dei pacchetti. Ti consigliamo di specificare sempre una versione specifica (==) o una versione massima (>=) per le dipendenze Python presenti nel tuo`requirements.txt`.
2. Controlla i log di Apache Airflow. Se hai abilitato i log di Apache Airflow, verifica che i gruppi di log siano stati creati correttamente nella pagina Gruppi di log dellaCloudWatch console. Se vedi log vuoti, il motivo più comune è dovuto alla mancanza di autorizzazioni nel tuo ruolo di esecuzioneCloudWatch per Amazon S3 in cui vengono scritti i log. Per ulteriori informazioni, consulta Ruolo di esecuzione.
3. Controlla le opzioni di configurazione di Apache Airflow. Se utilizzi Secrets Manager, verifica che le coppie chiave-valore che hai specificato come opzione di configurazione di Apache Airflow siano state configurate correttamente. Per ulteriori informazioni, consulta the section called “Configurazione di Secrets Manager”.
4. Controlla la configurazione della rete VPC. Per ulteriori informazioni, consulta the section called “Ambiente bloccato”.

5. Verifica le autorizzazioni del ruolo di esecuzione. Un ruolo di esecuzione è un ruoloAWS Identity and Access Management (IAM) con una politica di autorizzazioni che concede ad Amazon MWAA il permesso di richiamare le risorse di altriAWS servizi (come Amazon S3CloudWatch, Amazon SQS, Amazon ECR) per tuo conto. È inoltre necessario consentire l'accesso alla chiaveAWS gestita dal cliente o alla chiave di proprietà. Per ulteriori informazioni, consulta Ruolo di esecuzione.
6. Per eseguire uno script di risoluzione dei problemi che verifichi la configurazione e la configurazione della rete Amazon VPC per il tuo ambiente Amazon MWAA, consulta lo script Verify Environment inAWS Support Tools suGitHub.

Plug-in

L'argomento seguente descrive i problemi che potresti riscontrare durante la configurazione o l'aggiornamento dei plugin di Apache Airflow.

Amazon MWAA supporta l'implementazione dell'interfaccia utente personalizzata?

A partire da Apache Airflow v2.2.2, Amazon MWAA supporta l'installazione di plugin sul server Web Apache Airflow e l'implementazione di un'interfaccia utente personalizzata. Se il tuo ambiente Amazon MWAA esegue Apache Airflow v2.0.2 o versioni precedenti, non sarai in grado di implementare un'interfaccia utente personalizzata.

Per ulteriori informazioni sulla gestione delle versioni e sull'aggiornamento degli ambienti esistenti, consultaVersioni.

Sono in grado di implementare modifiche personalizzate all'interfaccia utente sul runner locale Amazon MWAA tramite plugin, tuttavia quando provo a fare lo stesso su Amazon MWAA, non vedo le mie modifiche né alcun errore. Perché sta succedendo questo?

il runner locale Amazon MWAA ha tutti i componenti di Apache Airflow raggruppati in un'unica immagine, consentendoti di applicare modifiche personalizzate ai plug-in dell'interfaccia utente.

Creare bucket.

L'argomento seguente descrive gli errori che potresti ricevere durante la creazione di un bucket Amazon S3.

Non riesco a selezionare l'opzione per le impostazioni dell'accesso pubblico ai blocchi Amazon S3

Il ruolo di esecuzione per il tuo ambiente Amazon MWAA richiede l'autorizzazione all'`GetBucketPublicAccessBlock`azione sul bucket Amazon S3 per verificare che il bucket abbia bloccato l'accesso pubblico. È consigliabile eseguire le operazioni seguenti:

1. Segui i passaggi per allegare una policy JSON al tuo ruolo di esecuzione.
2. Allega la seguente policy JSON:

```
{
   "Effect":"Allow",
   "Action":[
      "s3:GetObject*",
      "s3:GetBucket*",
      "s3:List*"
   ],
   "Resource":[
      "arn:aws:s3:::YOUR_S3_BUCKET_NAME",
      "arn:aws:s3:::YOUR_S3_BUCKET_NAME/*"
   ]
}
```

   Sostituisci i segnaposti di esempio in *YOUR_S3_BUCKET_NAME con il nome* del tuo bucket Amazon S3, ad esempio *my-mwaa-unique-s3-bucket-name*.
3. Per eseguire uno script di risoluzione dei problemi che verifichi la configurazione e la configurazione della rete Amazon VPC per il tuo ambiente Amazon MWAA, consulta lo script Verify Environment inAWS Support Tools suGitHub.

Creazione dell'ambiente

L'argomento seguente descrive gli errori che potresti ricevere durante la creazione di un ambiente.

Ho provato a creare un ambiente ed è bloccato nello stato «Creazione»

È consigliabile eseguire le operazioni seguenti:

1. Controlla la rete VPC con il routing pubblico. Se utilizzi un Amazon VPC con accesso a Internet, verifica quanto segue:

- Che il tuo Amazon VPC sia configurato per consentire il traffico di rete tra le diverseAWS risorse utilizzate dal tuo ambiente Amazon MWAA, come definito inthe section called "Informazioni sul networking". Ad esempio, il gruppo di sicurezza VPC deve consentire tutto il traffico in una regola di autoreferenziazione o, facoltativamente, specificare l'intervallo di porte per l'intervallo di porte HTTPS 443 e un intervallo di porte TCP 5432.

2. Controlla la rete VPC con routing privato. Se utilizzi un Amazon VPC senza accesso a Internet, verifica quanto segue:

   - Che il tuo Amazon VPC sia configurato per consentire il traffico di rete tra le diverseAWS risorse per il tuo ambiente Amazon MWAA, come definito inthe section called "Informazioni sul networking". Ad esempio, le due sottoreti private non devono avere una tabella di routing verso un gateway NAT (o istanza NAT) né un gateway Internet.

3. Per eseguire uno script di risoluzione dei problemi che verifichi la configurazione e la configurazione della rete Amazon VPC per il tuo ambiente Amazon MWAA, consulta lo script Verify Environment inAWS Support Tools suGitHub.

Ho provato a creare un ambiente ma mostra lo stato «Creazione fallita»

È consigliabile eseguire le operazioni seguenti:

1. Controlla la configurazione della rete VPC. Per ulteriori informazioni, consulta the section called "Ambiente bloccato".
2. Controlla le autorizzazioni degli utenti. Amazon MWAA esegue una prova a secco con le credenziali di un utente prima di creare un ambiente. Il tuoAWS account potrebbe non avere l'autorizzazione inAWS Identity and Access Management (IAM) per creare alcune risorse per un ambiente. Ad esempio, se hai scelto la modalità di accesso Apache Airflow alla rete privata, al tuoAWS account deve essere stato concesso l'accesso dall'amministratore alla politica di controllo degliFullConsoleAccess accessi AmazonMWAA per il tuo ambiente, che consente al tuo account di creare endpoint VPC.
3. Verifica le autorizzazioni del ruolo di esecuzione. Un ruolo di esecuzione è un ruoloAWS Identity and Access Management (IAM) con una politica di autorizzazioni che concede ad Amazon MWAA il permesso di richiamare le risorse di altriAWS servizi (come Amazon S3CloudWatch, Amazon SQS, Amazon ECR) per tuo conto. È inoltre necessario consentire l'accesso alla chiaveAWS gestita dal cliente o alla chiave di proprietà. Per ulteriori informazioni, consulta Ruolo di esecuzione.

4. Controlla i log di Apache Airflow. Se hai abilitato i log di Apache Airflow, verifica che i gruppi di log siano stati creati correttamente nella pagina Gruppi di log dellaCloudWatch console. Se vedi log vuoti, il motivo più comune è dovuto alla mancanza di autorizzazioni nel tuo ruolo di esecuzioneCloudWatch per Amazon S3 in cui vengono scritti i log. Per ulteriori informazioni, consulta Ruolo di esecuzione.
5. Per eseguire uno script di risoluzione dei problemi che verifichi la configurazione e la configurazione della rete Amazon VPC per il tuo ambiente Amazon MWAA, consulta lo script Verify Environment inAWS Support Tools suGitHub.
6. Se utilizzi un Amazon VPC senza accesso a Internet, assicurati di aver creato un endpoint gateway Amazon S3 e di aver concesso le autorizzazioni minime richieste ad Amazon ECR per accedere ad Amazon S3. Per ulteriori informazioni sulla creazione di un endpoint gateway Amazon S3, consulta le seguenti risorse:
   - Creazione di una rete Amazon VPC senza accesso a Internet
   - Crea l'endpoint gateway Amazon S3 nella Guida per l'utente di Amazon Elastic Container Registry

Ho provato a selezionare un VPC e ho ricevuto un errore «Errore di rete»

È consigliabile eseguire le operazioni seguenti:

- Se visualizzi un errore «Errore di rete» quando provi a selezionare un Amazon VPC durante la creazione del tuo ambiente, disattiva tutti i proxy del browser in esecuzione, quindi riprova.

Ho provato a creare un ambiente e ho ricevuto un errore di servizio, partizione o risorsa «deve essere passato»

È consigliabile eseguire le operazioni seguenti:

- Potresti ricevere questo errore perché l'URI che hai specificato per il tuo bucket Amazon S3 include un '/' alla fine dell'URI. Ti consigliamo di rimuovere il '/' nel percorso. Il valore deve avere il formato seguente:

```
s3://your-bucket-name
```

Ho provato a creare un ambiente e mostra lo stato come «Disponibile», ma quando provo ad accedere all'interfaccia utente di Airflow viene visualizzato un errore «Risposta vuota dal server» o «502 Bad Gateway»

È consigliabile eseguire le operazioni seguenti:

1. Verifica la configurazione del gruppo di sicurezza VPC. Per ulteriori informazioni, consulta the section called “Ambiente bloccato”.
2. Verifica che tutti i pacchetti Apache Airflow elencati`requirements.txt` corrispondano alla versione di Apache Airflow in esecuzione su Amazon MWAA. Per ulteriori informazioni, consulta Installazione delle dipendenze in Python.
3. Per eseguire uno script di risoluzione dei problemi che verifichi la configurazione e la configurazione della rete Amazon VPC per il tuo ambiente Amazon MWAA, consulta lo script Verify Environment inAWS Support Tools suGitHub.

Ho provato a creare un ambiente e il mio nome utente è un gruppo di nomi di caratteri casuali

- Apache Airflow ha un massimo di 64 caratteri per i nomi utente. Se il tuo ruoloAWS Identity and Access Management (IAM) supera questa lunghezza, viene utilizzato un algoritmo hash per ridurlo, pur rimanendo univoco.

Ambiente di aggiornamento

L'argomento seguente descrive gli errori che potresti ricevere durante l'aggiornamento di un ambiente.

Ho provato a cambiare la classe dell'ambiente ma l'aggiornamento non è riuscito

Se si aggiorna l'ambiente a una classe di ambiente diversa (ad esempio la modifica di una in una`mw1.small`) e la richiesta di aggiornamento dell'ambiente non riesce, lo`UPDATE_FAILED` stato dell'ambiente passa a uno stato e l'ambiente viene ripristinato e fatturato in base alla versione stabile precedente di un ambiente.`mw1.medium`

È consigliabile eseguire le operazioni seguenti:

1. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando l'opzione aws-mwaa-local-runneronGitHub.
2. Per eseguire uno script di risoluzione dei problemi che verifichi la configurazione e la configurazione della rete Amazon VPC per il tuo ambiente Amazon MWAA, consulta lo script Verify Environment inAWS Support Tools suGitHub.

Ambiente di accesso

L'argomento seguente descrive gli errori che potresti ricevere quando accedi a un ambiente.

Non riesco ad accedere all'interfaccia utente Apache Airflow

È consigliabile eseguire le operazioni seguenti:

1. Controlla le autorizzazioni degli utenti. È possibile che non ti sia stato concesso l'accesso a una politica di autorizzazioni che ti consenta di visualizzare l'interfaccia utente di Apache Airflow. Per ulteriori informazioni, consulta the section called “Accesso a un ambiente Amazon MWAA”.
2. Verifica l'accesso alla rete. Ciò può essere dovuto al fatto che hai selezionato la modalità di accesso alla rete privata. Se l'URL dell'interfaccia utente di Apache Airflow è nel seguente formato`387fbcn-8dh4-9hfj-0dnd-834jhdfb-vpce.c10.us-west-2.airflow.amazonaws.com`, significa che stai utilizzando un routing privato per il tuo server Web Apache Airflow. È possibile aggiornare la modalità di accesso Apache Airflow alla modalità di accesso alla rete pubblica oppure creare un meccanismo per accedere all'endpoint VPC per il server Web Apache Airflow. Per ulteriori informazioni, consulta the section called “Gestione dell'accesso agli endpoint VPC”.

Risoluzione dei problemi: CloudWatch log ed CloudTrail errori

Gli argomenti di questa pagina contengono le risoluzioni ad Amazon CloudWatch Logs e AWS CloudTrail gli errori che potresti riscontrare in un ambiente Amazon Managed Workflows for Apache Airflow.

Indice

- Log
  - Non riesco a visualizzare i miei registri delle attività o ho ricevuto l'errore «Leggere il registro remoto da Cloudwatch log_group»

- Le attività non vanno a buon fine senza alcun registro
- Vedo un errore '' in ResourceAlreadyExistsException CloudTrail
- Vedo un errore «Richiesta non valida» in CloudTrail
- Nei log di Apache Airflow viene visualizzato il messaggio «Impossibile individuare una libreria Oracle Client a 64 bit: «libclntsh.so: impossibile aprire il file o la directory condivisa: nessun file o directory di questo tipo»
- Vedo psycopg2 «il server ha chiuso la connessione in modo imprevisto» nei miei log di Scheduler
- Nei miei registri di elaborazione DAG vedo «Executor riporta che l'istanza dell'operazione %s è terminata (%s) anche se l'attività dice che è %s»
- Vedo 'Impossibile leggere i log remoti da log_group: airflow-* {*EnvironmentName} -Task log_stream: * {*DAG_ID} /* {*TASK_ID} /* {*time} /* {*n} .log. ' nei miei registri delle attività

Log

L'argomento seguente descrive gli errori che potresti ricevere durante la visualizzazione dei log di Apache Airflow.

Non riesco a visualizzare i miei registri delle attività o ho ricevuto l'errore «Leggere il registro remoto da Cloudwatch log_group»

Amazon MWAA ha configurato Apache Airflow per leggere e scrivere i log direttamente da e verso Amazon Logs. CloudWatch Se un lavoratore non riesce ad avviare un'attività o non riesce a scrivere alcun registro, verrà visualizzato l'errore:

```
*** Reading remote log from Cloudwatch log_group: airflow-environmentName-Task
 log_stream: DAG_ID/TASK_ID/timestamp/n.log.Could not read remote logs from log_group:
 airflow-environmentName-Task log_stream: DAG_ID/TASK_ID/time/n.log.
```

- È consigliabile eseguire le operazioni seguenti:

  a. Verifica di aver abilitato i registri delle attività a `INFO` livello del tuo ambiente. Per ulteriori informazioni, consulta Visualizzazione dei log di Airflow in Amazon CloudWatch.

  b. Verifica che il ruolo di esecuzione dell'ambiente disponga delle politiche di autorizzazione corrette.

c. Verifica che l'operatore o l'attività funzioni correttamente, disponga di risorse sufficienti per analizzare il DAG e disponga delle librerie Python appropriate da caricare. Per verificare se hai le dipendenze corrette, prova a eliminare le importazioni finché non trovi quella che causa il problema. Ti consigliamo di testare le tue dipendenze Python utilizzando lo strumento local-runner di Amazon MWAA.

Le attività non vanno a buon fine senza alcun registro

Se le attività non riescono in un flusso di lavoro e non riesci a individuare alcun registro delle attività non riuscite, controlla se stai impostando il `queue` parametro negli argomenti predefiniti, come illustrato di seguito.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

# Setting queue argument to default.
default_args = {
 "start_date": days_ago(1),
 "queue": "default"
}

with DAG(dag_id="any_command_dag", schedule_interval=None, catchup=False,
 default_args=default_args) as dag:
    cli_command = BashOperator(
        task_id="bash_command",
        bash_command="{{ dag_run.conf['command'] }}"
    )
```

Per risolvere il problema, rimuovi `queue` dal codice e richiama nuovamente il DAG.

Vedo un errore " in ResourceAlreadyExistsException CloudTrail

```
"errorCode": "ResourceAlreadyExistsException",
    "errorMessage": "The specified log stream already exists",
    "requestParameters": {
        "logGroupName": "airflow-MyAirflowEnvironment-DAGProcessing",
        "logStreamName": "scheduler_cross-account-eks.py.log"
    }
```

Alcuni requisiti di Python, come il `apache-airflow-backport-providers-amazon` ripristino della `watchtower` libreria utilizzata da Amazon MWAA per comunicare a una versione precedente CloudWatch . È consigliabile eseguire le operazioni seguenti:

- Aggiungi la seguente libreria alla tua `requirements.txt`

```
watchtower==1.0.6
```

Vedo un errore «Richiesta non valida» in CloudTrail

```
Invalid request provided: Provided role does not have sufficient permissions for s3
 location airflow-xxx-xxx/dags
```

Se stai creando un ambiente Amazon MWAA e un bucket Amazon S3 utilizzando lo AWS CloudFormation stesso modello, devi aggiungere `DependsOn` una sezione all'interno del modello. AWS CloudFormation Le due risorse (MWAA Environment e MWAA Execution Policy) dipendono da. AWS CloudFormation È consigliabile eseguire le operazioni seguenti:

- Aggiungi la seguente dichiarazione al tuo modello**`DependsOn`**. AWS CloudFormation

```
...
      MaxWorkers: 5
      NetworkConfiguration:
        SecurityGroupIds:
          - !GetAtt SecurityGroup.GroupId
        SubnetIds: !Ref subnetIds
      WebserverAccessMode: PUBLIC_ONLY
    DependsOn: MwaaExecutionPolicy

    MwaaExecutionPolicy:
    Type: AWS::IAM::ManagedPolicy
    Properties:
      Roles:
        - !Ref MwaaExecutionRole
      PolicyDocument:
        Version: 2012-10-17
        Statement:
          - Effect: Allow
            Action: airflow:PublishMetrics
            Resource:
```

```
...
```

Per un esempio, consulta Tutorial di avvio rapido per Amazon Managed Workflows for Apache Airflow.

Nei log di Apache Airflow viene visualizzato il messaggio «Impossibile individuare una libreria Oracle Client a 64 bit: «libclntsh.so: impossibile aprire il file o la directory condivisa: nessun file o directory di questo tipo»

- È consigliabile eseguire le operazioni seguenti:
  - Se utilizzi Apache Airflow v2, aggiungilo `core.lazy_load_plugins : False` come opzione di configurazione Apache Airflow. Per saperne di più, consulta Usare le opzioni di configurazione per caricare i plugin in 2.

Vedo psycopg2 «il server ha chiuso la connessione in modo imprevisto» nei miei log di Scheduler

Se viene visualizzato un errore simile al seguente, è possibile che Apache Airflow Scheduler abbia esaurito le risorse.

```
2021-06-14T10:20:24.581-05:00    sqlalchemy.exc.OperationalError:
 (psycopg2.OperationalError) server closed the connection unexpectedly
2021-06-14T10:20:24.633-05:00    This probably means the server terminated abnormally
2021-06-14T10:20:24.686-05:00    before or while processing the request.
```

È consigliabile eseguire le operazioni seguenti:

- Prendi in considerazione l'aggiornamento ad Apache Airflow v2.0.2, che consente di specificare fino a 5 Scheduler.

Nei miei registri di elaborazione DAG vedo «Executor riporta che l'istanza dell'operazione %s è terminata (%s) anche se l'attività dice che è %s»

Se visualizzi un errore simile al seguente, è possibile che le attività a esecuzione prolungata abbiano raggiunto il limite di tempo consentito per le attività su Amazon MWAA. Amazon MWAA ha un limite

di 12 ore per ogni attività Airflow, per evitare che le attività rimangano bloccate in coda e blocchino attività come la scalabilità automatica.

```
Executor reports task instance %s finished (%s) although the task says its %s. (Info:
 %s) Was the task killed externally
```

È consigliabile eseguire le operazioni seguenti:

- Prendi in considerazione la possibilità di suddividere l'attività in più attività con esecuzione più breve. Airflow ha in genere un modello in base al quale gli operatori sono asincroni. Richiama attività su sistemi esterni e Apache Airflow Sensors effettua un sondaggio per vedere quando è completo. Se un sensore si guasta, può essere riprovato in sicurezza senza influire sulla funzionalità dell'operatore.

Vedo 'Impossibile leggere i log remoti da log_group: airflow-* {*EnvironmentName} -Task log_stream: * {*DAG_ID} /* {*TASK_ID} /* {*time} /* {*n} .log. ' nei miei registri delle attività

Se viene visualizzato un errore simile al seguente, è possibile che il ruolo di esecuzione per l'ambiente in uso non contenga una politica di autorizzazioni per la creazione di flussi di registro per i registri delle attività.

```
Could not read remote logs from log_group: airflow-*{*environmentName}-Task
 log_stream:* {*DAG_ID}/*{*TASK_ID}/*{*time}/*{*n}.log.
```

È consigliabile eseguire le operazioni seguenti:

- Modifica il ruolo di esecuzione per il tuo ambiente utilizzando una delle politiche di esempio disponibili in. the section called “Ruolo di esecuzione”

È inoltre possibile che nel `requirements.txt` file sia stato specificato un pacchetto provider incompatibile con la versione di Apache Airflow in uso. Ad esempio, se utilizzi Apache Airflow v2.0.2, potresti aver specificato un pacchetto, come il pacchetto, compatibile solo con Airflow apache-airflow-providers-databricks2.1+.

È consigliabile eseguire le operazioni seguenti:

1. Se utilizzi Apache Airflow v2.0.2, modifica il file e aggiungi. `requirements.txt` `apache-airflow[databricks]` Questo installa la versione corretta del pacchetto Databricks compatibile con Apache Airflow v2.0.2.
2. Testa i tuoi DAG, i plugin personalizzati e le dipendenze Python localmente usando on. aws-mwaa-local-runner GitHub

Cronologia dei documenti Amazon MWAA

La tabella seguente descrive importanti aggiunte alla documentazione del servizio Amazon MWAA, a partire da novembre 2020. Per ricevere notifiche sugli aggiornamenti di questa documentazione, iscriviti al feed RSS.

Modifica	Descrizione	Data
Amazon MWAA supporta il ridimensionamento automatico dei server Web e l'API REST di Apache Airflow	Amazon MWAA ora supporta il ridimensionamento automatico dei server Web, nonché la possibilità di accedere e utilizzare l'API REST di Apache Airflow. • the section called “Configurazione della scalabilità automatica del server Web” • the section called “Utilizzo dell'API REST di Apache Airflow”	16 maggio 2024
Descrizione migliorata del comportamento di ridimensionamento automatico	L'argomento seguente è stato aggiornato per riflettere il nuovo comportamento di scalabilità automatica di Amazon MWAA quando i lavoratori assumono nuove attività mentre i lavoratori di Fargate effettuano la scalabilità. • the section called “Configurazione della scalabilità automatica dei lavoratori”	10 maggio 2024

Support per istanze di dimensioni maggiori	Amazon MWAA ora supporta due opzioni di istanze di dimensioni maggiori per carichi di lavoro più grandi:, e `mw1.xlarge mw1.2xlarge` • the section called “Funzionalità ambientali”	16 aprile 2024
Nuova versione di Apache Airflow	Amazon MWAA ora supporta Apache Airflow v2.8.1. Questo aggiornamento include informazioni sui pacchetti provider aggiornati e dettagli sull'utilizzo di Apache Airflow v2.8.1 su Amazon MWAA. • Versioni • the section called “Pacchetti provider per connessioni Apache Airflow v2.8.1”	22 febbraio 2024

Support per Amazon VPC condiviso	Amazon MWAA supporta la creazione di ambienti tra account per le organizza zioni che utilizzano Amazon OpenSearch Service per gestire le risorse Amazon MWAA utilizzando un Amazon VPC condiviso centralizzato in un account proprietario. Come parte di questo lancio, Amazon MWAA ti consente di scegliere di creare e gestire i tuoi endpoint Amazon VPC. • the section called “Gestire i propri endpoint Amazon VPC”	15 novembre 2023
Nuova versione di Apache Airflow	Amazon MWAA ora supporta Apache Airflow v2.7.2. Questo aggiornamento include informazioni sui pacchetti provider aggiornati e dettagli sull'uso di Apache Airflow v2.7.2 su Amazon MWAA. • Versioni • the section called “Pacchetti provider per connessioni Apache Airflow v2.7.2”	6 novembre 2023

Nuova versione di Apache Airflow	Amazon MWAA ora supporta Apache Airflow v2.6.3. Questo aggiornamento include informazioni sui pacchetti provider aggiornati e dettagli sull'utilizzo di Apache Airflow v2.6.3 su Amazon MWAA, • Versioni • the section called “Pacchetti provider per connessioni Apache Airflow v2.6.3”	9 agosto 2023
Informazioni sulla deprecazione della versione	Argomento aggiornato sulla deprecazione della versione per includere avvisi e tempistiche di deprecazione per Apache Airflow v2.0.2 e Apache Airflow v2.2.2. • the section called “Versioni obsolete di Apache Airflow”	31 luglio 2023
Nuovi argomenti e casi d'uso	Amazon MWAA supporta aggiornamenti di versioni minori. Questo aggiornamento include il seguente nuovo argomento che descrive come aggiornare l'ambiente e assicurarsi che le risorse del flusso di lavoro siano compatibili con la versione di Apache Airflow a cui si sta effettuando l'aggiornamento: • the section called “Aggiornamento della versione”	5 giugno 2023

Argomento aggiornato	Policy IAM aggiornate gestite dal cliente che garantisc ono a un utente l'accesso completo alla console e all'API ad Amazon MWAA. L'aggiornamento descrive il motivo per cui è necessario fornire l'autorizzazione per `iam:PassRole` consentir e a un utente di trasferire ruoli ad Amazon MWAA. Amazon MWAA utilizza queste autorizzazioni per eseguire azioni per conto di un utente. • the section called “Accesso a un ambiente Amazon MWAA”	12 aprile 2023

Nuove linee guida	Argomento aggiornato sulla configurazione AWS Secrets Manager come backend per Amazon MWAA per fornire indicazioni sull'uso dei modelli di ricerca. L'uso di modelli di ricerca restringe i segreti ricercati da Apache Airflow e riduce il numero di chiamate API che Amazon MWAA effettua a Secrets Manager per recuperare connessioni e variabili. Ciò riduce i costi associati all'utilizzo di Secrets Manager come backend. • Crea il backend Secrets Manager come opzione di configurazione di Apache Airflow	12 aprile 2023
Nuova versione di Apache Airflow	Amazon MWAA ora supporta Apache Airflow v2.5.1. Questo aggiornamento include informazioni sui pacchetti provider aggiornati e dettagli sull'utilizzo di Apache Airflow v2.5.1 su Amazon MWAA, • Versioni • the section called “Pacchetti provider per connessioni Apache Airflow v2.5.1”	11 aprile 2023

Nuovi argomenti e casi d'uso	È stato aggiunto un nuovo argomento sull'uso di uno script di avvio con un ambiente Amazon MWAA. Questo argomento descrive la configurazione di uno script di avvio per un ambiente esistente, il suo utilizzo per installare i runtime Linux e l'impostazione delle variabili di ambiente. • the section called “Utilizzo di uno script di avvio”	3 aprile 2023
Sezione aggiornata sull'accesso ai server Web privati	È stato aggiornato il seguente argomento sull'accesso ai server Web privati. L'aggiornamento chiarisce che, in ambienti con accesso privato al server Web, è necessario utilizzare un Python wheel archive `.whl` () per impacchettare e installare le dipendenze. • Modalità di accesso al server web privato	24 febbraio 2023

Sono state aggiunte informazioni sulle versioni obsolete di Apache Airflow	È stato aggiornato l'argomento Versioni con nuove informazioni su come Amazon MWAA ha gestito le versioni obsolete di Apache Airflow. È stata rimossa una sezione sull'aggiornamento a una versione più recente di Apache Airflow e una sezione che descrive le modifiche tra Apache Airflow v1 e Apache Airflow v2. Per ulteriori informazioni sulla migrazione a una nuova versione di Apache Airflow, consulta la Amazon MWAA Migration Guide. • the section called “Versioni obsolete di Apache Airflow” • the section called “Supporto e domande frequenti sulla versione di Apache Airflow”	17 febbraio 2023

Correzioni nelle metriche dei container Amazon MWAA	È stato aggiornato l'argomento sulle metriche del contenitore e rimosso un set di parametri errati che non esistevano nella dimensione. `Cluster` È stata aggiunta una sezione aggiuntiva che descrive come valutare il numero di lavoratori aggiuntivi utilizzati da un ambiente in un determinato momento rappresentando graficamente la metrica `CPUUtilization` o la `MemoryUtilization` metrica del `AdditionalWorker` componente e impostando il tipo di statistica su. `Sample Count` • the section called “Valutazione del numero di container di worker e server Web aggiuntivi”	20 gennaio 2023

Nuova versione di Apache Airflow	Amazon MWAA ora supporta Apache Airflow v2.4.3. Questo aggiornamento include informazioni sui pacchetti provider aggiornati, dettagli sull'uso di Apache Airflow v2.4.3 su Amazon MWAA e informazioni consolidate sulle funzionalità supportate in ciascuna versione di Apache Airflow su Amazon MWAA. • Versioni • the section called “Pacchetti provider per connessioni Apache Airflow v2.4.3”	5 gennaio 2023
Argomento aggiornato sul ruolo collegato ai servizi	Informazioni aggiornate sul ruolo collegato ai servizi che Amazon MWAA utilizza per creare e gestire AWS risorse per tuo conto, incluse informazioni su come eliminare il ruolo collegato al servizio quando non è più necessario. Ciò include una politica di autorizzazione aggiornata dei ruoli collegati al servizio che consente ad Amazon MWAA di pubblicare parametri aggiuntivi nello spazio dei nomi. CloudWatch `AWS/MWAA` • the section called “Ruolo collegato al servizio”	18 novembre 2022

Nuovo argomento sulle metriche dei servizi	È stato aggiunto un nuovo argomento che descrive i parametri di servizio emessi da Amazon MWAA nello spazio dei nomi. `AWS/MWAA` Questi includono gli scheduler , i worker e i server Web dei parametri del cluster Amazon ECS, i parametri di Amazon SQS per le code che consentono ad Amazon MWAA di disaccoppiare scheduler e worker, nonché i parametri Amazon RDS per il database dei metadati. • the section called “Metriche relative a contenitori, code e database”	18 novembre 2022
Nuovo argomento	Sono state aggiunte nuove linee guida sulla modifica di un file di vincoli per specifica re nuove versioni dei pacchetti di provider da utilizzare con l'ambiente Amazon MWAA. • the section called “Specific are pacchetti di provider più recenti”	18 novembre 2022
Voce FAQ aggiornata	Informazioni aggiornate relative all'idoneità HIPAA di Amazon MWAA. • the section called “Conformi tà agli standard HIPAA”	15 novembre 2022

Nuovo argomento	È stato aggiunto un nuovo argomento sull'utilizzo `aws:SourceArn` delle chiavi di contesto delle condizioni `aws:SourceAccount` globali in una policy di fiducia per i ruoli di esecuzione di Amazon MWAA, al fine di evitare la confusione tra servizi. • the section called “Prevenzione del confused deputy tra servizi”	21 ottobre 2022
Nuovo codice di esempio	Sono state aggiunte istruzioni aggiornate e un esempio di codice DAG in cui vengono scritte metriche personalizzate a livello di sistema operativo. CloudWatch • the section called “Utilizzo di un DAG per scrivere metriche personalizzate”	13 settembre 2022
Nuovo codice di esempio	Sono state aggiunte istruzioni aggiornate e un nuovo esempio di codice AWS Lambda Python che recupera un token CLI Apache Airflow, quindi richiama un DAG in un ambiente Amazon MWAA specificato. • the section called “Richiamare i DAG con Lambda”	12 settembre 2022

Nuovi diagrammi architettonici	Sono stati aggiunti nuovi diagrammi architettonici che illustrano un ambiente Amazon MWAA con un server Web pubblico e privato. • the section called “Modalità di accesso Apache Airflow”	12 settembre 2022
Nuovo codice di esempio	Sono state aggiunte istruzioni aggiornate e un nuovo esempio di codice DAG che recupera un token CLI Apache Airflow, quindi richiama un altro DAG in un ambiente Amazon MWAA diverso. • the section called “Richiamare i DAG in diversi ambienti”	16 agosto 2022
Nuovo codice di esempio	Sono state aggiunte istruzioni aggiornate e un nuovo DAG che interroga Aurora PostgreSQL di un ambiente per ottenere informazioni sui metadati, scrive il risultato in file CSV e archivia i file in Amazon S3. • the section called “Esportazione di metadati ambientali in Amazon S3”	12 agosto 2022

Nuovo codice di esempio	Sono state aggiunte istruzioni aggiornate e nuovo DAG che aggiorna un AWS CodeArtifact token in fase di esecuzione e archivia il risultato in Amazon S3. • the section called “Rinfrescante eAWS CodeArtifacttoken in fase di esecuzione”	3 agosto 2022
Nuovo codice di esempio	Sono state aggiunte istruzioni aggiornate e un esempio di codice DAG per l'`ECSOperator` utilizzo di Amazon MWAA. • the section called “Utilizzo di `ECSOperator` ”	26 luglio 2022
Nuovo codice di esempio	Sono state aggiunte istruzioni aggiornate e un esempio di codice DAG per l'`SSHOperator` utilizzo di Amazon MWAA. • the section called “Utilizzo di `SSHOperator` ”	15 luglio 2022
Nuovo codice di esempio	Sono state aggiunte nuove istruzioni e un esempio di codice DAG per l'utilizzo di dbt Postgres con Amazon MWAA. • the section called “Usare dbt con Amazon MWAA”	17 giugno 2022

Nuovi argomenti e casi d'uso	Sono state aggiunte nuove istruzioni e un esempio di codice DAG per l'installazione delle dipendenze utilizzando i file wheel Python per ambienti Amazon MWAA con accesso pubblico e privato. • Gestire le dipendenze usando le ruote Python	13 maggio 2022
Nuovi argomenti e casi d'uso	Sono state aggiunte nuove linee guida sulla scelta dei parametri di Apache Airflow a cui inviare Amazon MWAA. CloudWatch • Scelta dei parametri di Apache Airflow da riportare	19 aprile 2022
Nuove guide	Amazon MWAA offre una guida alla migrazione per migrare i flussi di lavoro Apache Airflow da implement azioni autogestite e ambienti Amazon MWAA esistenti. • Guida alla migrazione ad Amazon MWAA	7 marzo 2022

Nuovi argomenti e casi d'uso	Sono state aggiunte nuove best practice di sicurezza per l'utilizzo di Apache Airflow, inclusa una soluzione per rilevare le modifiche ai privilegi utente di Apache Airflow. • the section called “Le migliori pratiche di sicurezza in Apache Airflow”	18 febbraio 2022
Nuovo codice di esempio	È stato aggiunto un nuovo esempio di codice per la creazione di DAG compatibili con il fuso orario utilizzando Pendulum e ha chiarito come utilizzare un plug-in personalizzato per modificare il fuso orario in cui vengono creati i log di Apache Airflow. • the section called “Modifica del fuso orario di un DAG”	11 febbraio 2022

Lancio di Apache Airflow v2.2.2	Amazon Managed Workflows per Apache Airflow ora supporta Apache Airflow v2.2.2. A partire dalla versione 2.2, Amazon MWAA installer à pacchetti Python e plug-in personalizzati direttamente sul server Web Apache Airflow, consentendoti una maggiore flessibilità nella gestione degli ambienti. Per ulteriori informazioni, consulta gli argomenti seguenti. • Versioni di Apache Airflow su Amazon Managed Workflows per Apache Airflow. • the section called “Pacchetti provider per connessioni Apache Airflow v2.2.2”. • Registro delle modifiche di Apache Airflow v2.2.2 sul sito Web di documentazione di Apache Airflow.	27 gennaio 2022

Nuovi tutorial	È stato aggiunto un nuovo tutorial che dimostra la creazione di un nuovo ruolo Apache Airflow personalizzato e l'assegnazione del ruolo a un utente Apache Airflow mappato da IAM per limitare l'accesso dell'utente a un sottoinsieme di DAG specifica ti. • the section called “Tutorial: limitazione degli utenti a un sottoinsieme di DAG”	8 dicembre 2021

Correzioni	È stata corretta una raccomandazione sulle migliori pratiche per l'impostazione del valore di al fine di ottimizzare l'utilizzo della CPU. `scheduler.min_file_process_interval` È stato aggiunto un esempio di policy IAM che concede l'accesso alle risorse di Secrets Manager nel ruolo di esecuzione. È stato aggiunto un argomento di risoluzione dei problemi sull'utilizzo delle chiavi di condizione di Secrets Manager. • Ottimizzazione delle prestazioni del modo in cui lo scheduler analizza i DAG • Fornisci ad Amazon MWAA l'autorizzazione ad accedere alle chiavi segrete di Secrets Manager • Configurazione delle chiavi di condizione nel ruolo di esecuzione di Amazon MWAA per Secrets Manager	22 novembre 2021

Nuovo codice di esempio	Aggiunto il seguente nuovo esempio di codice per modificare il fuso orario in cui i DAG vengono elaborati utilizzando un plug-in personalizzato e un nuovo argomento di risoluzione dei problemi per richiamare il comando CLI Apache `dags backfill` Airflow dall'interno di un operatore bash. • the section called “Modifica del fuso orario di un DAG” • Compila il comando CLI usando un operatore bash	1° novembre 2021
Correzioni	Sono stati corretti i problemi nell'esempio di codice dell'operatore Amazon ECS e sono state chiarite le autorizzazioni aggiuntive richieste nel ruolo di esecuzione di Amazon MWAA per consentire all'ambiente di accedere al gruppo di log di attività di Amazon ECS in Logs. CloudWatch • Autorizzazioni dell'operatore Amazon ECS.	26 ottobre 2021

Nuovo codice di esempio	È stato aggiunto un nuovo esempio di codice che interroga il database Aurora PostgreSQL alla ricerca di informazioni relative al DAG, esegue e scrive i risultati su file archiviati su Amazon S3. `CSV` • the section called “Esportazione di metadati ambientali in Amazon S3”.	1° ottobre 2021
Correzioni	Informazioni corrette su come Amazon MWAA sincronizza automaticamente oggetti nuovi e modificati dal bucket Amazon S3 di destinazione ai tuoi pianificatori e lavoratori. • Come funziona la cartella DAG.	1° ottobre 2021
Ora supportata	Amazon MWAA ora supporta pacchetti provider aggiuntivi per Apache Airflow 2.0+. Per ulteriori informazioni sui pacchetti supportati, consulta quanto segue: • the section called “Pacchetti provider per connessioni Apache Airflow v2.0.2”.	24 settembre 2021

Nuovi comandi e procedure	Sono state aggiunte ulteriori linee guida ed esempi di AWS CLI comandi per la creazione di un endpoint gateway Amazon S3 quando si utilizza un Amazon VPC senza accesso a Internet: • Creazione di una rete Amazon VPC senza accesso a Internet.	24 settembre 2021
Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un nuovo esempio di codice che utilizza un operatore Amazon Elastic Container Service inthe section called “Utilizzo di `ECSOperator` ”. • Sono stati aggiunti nuovi argomenti per la risoluzio ne dei problemi relativi alla configurazione dei plugin Apache Airflow in. the section called “Plug-in”	19 settembre 2021

Nuova regione supportata	Amazon MWAA è ora disponibile nelle seguenti regioni: • Asia Pacifico (Mumbai) - ap-south-1 • Asia Pacifico (Seoul) - ap-northeast-2 • Europa (Londra) - eu-west-2 • Europa (Parigi) - eu-west-3 • Canada (Centrale) - ca-central-1 • Sud America (San Paolo) - sa-east-1 Per ulteriori informazioni sulla disponibilità delle regioni e sugli endpoint di servizio, consulta quanto segue: • Endpoint e quote Amazon MWAA in. Riferimenti generali di AWS	31 agosto 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Sono state aggiornate le politiche di esempio per consentire ad Amazon MWAA di recuperare le impostazioni Amazon S3 a livello di account () in. `s3:GetAccountPublicAccessBlock` Ruolo di esecuzione di Amazon MWAA	27 agosto 2021
Correzioni	Sono state aggiunte le seguenti modifiche: • È stato corretto il AWS CloudFormation modello per utilizzare una regola in entrata autoreferenziata per il gruppo di sicurezza in. Crea la rete VPC • È stato corretto il AWS CloudFormation modello per utilizzare una regola in entrata autoreferenziata per il gruppo di sicurezza in. Tutorial di avvio rapido per Amazon Managed Workflows for Apache Airflow	27 agosto 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto il decorator e DAG all'elenco di ciò che è supportato per Apache Airflow v2.0.2. Versioni di Apache Airflow su Amazon Managed Workflows per Apache Airflow	20 agosto 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un caso `celery.sync_parallelism` d'uso aOttimizzazione delle prestazioni per Apache Airflow su Amazon MWAA. • Sono stati aggiunti gli endpoint del servizio alla pagina delle quote e il nome è stato modificato in. Endpoint e quote del servizio Amazon Managed Workflows per Apache Airflow • Prerequisiti di rete chiariti in base al feedback degli utenti all'indirizzo. Inizia a usare Amazon Managed Workflows per Apache Airflow • Comandi CLI Airflow Spostati `dags list-runs` e `dags next-execution` non supportati in. Riferimento ai comandi CLI di Apache Airflow	13 agosto 2021

Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un esempio bash per impostare, ottenere o eliminare una variabile Apache Airflow v2.0.2 in. Riferimento ai comandi CLI di Apache Airflow • Sono state aggiunte le dipendenze di Apache Airflow v2.0.2 e l'esempio di connessione Airflow a. Utilizzo di Amazon MWAA con Amazon RDS per Microsoft SQL Server	13 agosto 2021
Correzioni	Sono state aggiunte le seguenti modifiche: • È stato corretto l'esempio di codice Python basato sul feedback degli utenti all'indirizzo. Creazione di una connessione SSH utilizzando `SSHOperator`	13 agosto 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • `variables set`Spostato nei comandi CLI Airflow supportati in. Riferimento ai comandi CLI di Apache Airflow • È stato aggiunto il riepilogo di Cosa è cambiato nella v2.0.2 dalla pagina delle versioni di Airflow in base al Installazione delle dipendenze in Python feedback degli utenti. • È stato aggiunto il riepilogo delle modifiche nella v2.0.2 dalla pagina delle versioni di Airflow in base al feedback degli utenti. Riferimento ai comandi CLI di Apache Airflow • È stato aggiunto il riepilogo delle modifiche nella v2.0.2 dalla pagina delle versioni di Airflow in base al feedback degli utenti. Panoramica dei tipi di connessione • È stato aggiunto il riepilogo delle modifiche nella v2.0.2 dalla pagina delle versioni di Airflow in base al feedback degli utenti. Installazione di plugin personalizzati	6 agosto 2021

	• È stato aggiunto il riepilogo delle modifiche nella v2.0.2 dalla pagina delle versioni di Airflow in base al feedback degli utenti. Aggiunta o aggiornamento di DAG	
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto il codice di esempio di Apache Airflow v2.0.2 a. Utilizzo di un DAG per importare variabili nella CLI • È stato aggiunto il codice di esempio Apache Airflow v2.0.2 a. Invocare DAG con una funzione Lambda	6 agosto 2021
Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un argomento di risoluzione dei problemi per «Non riesco a visualizzare la mia connessione nell'interfaccia utente di Airflow» all'indir izzo. Amazon Managed Workflows for Apache Airflow • È stato aggiunto un elenco di Amazon VPC a cui Amazon MWAA supporta. Informazioni sulla rete su Amazon MWAA	29 luglio 2021

Correzioni	Sono state aggiunte le seguenti modifiche: • È stato corretto l'esempio di codice Python basato sul feedback degli utenti su cui stampare il token di accesso Web su. Creare un token di accesso al server web Apache Airflow • È stato corretto l'argomento relativo alla connessione Snowflake in base al feedback degli utenti che prevedeva l'utilizzo di un'unica quotazione per il parametro warehouse su. Amazon Managed Workflows for Apache Airflow	29 luglio 2021

Argomenti rimossi o spostati	Sono state aggiunte le seguenti modifiche: • È stata ristrutturata la pagina esistente per includere tutte le pagine di documentazione sul monitoraggio e sulle metriche. Monitoraggio e parametri per Amazon Managed Workflows for Apache Airflow • Spostato nel Metriche di ambiente Apache Airflow v2 in CloudWatch menu di navigazione del monitoraggio e delle metriche.	23 luglio 2021
Nuove guide	Sono state aggiunte le seguenti modifiche: • CreatoPacchetti del provider Apache Airflow installati in ambienti Amazon MWAA. • CreatoPanoramica del monitoraggio su Amazon MWAA. • CreatoVisualizzazione dei registri di controllo AWS CloudTrail. • CreatoVisualizzazione dei log di Airflow in Amazon CloudWatch.	23 luglio 2021

Correzioni	Sono state aggiunte le seguenti modifiche: • È stato corretto l'esempio di codice Python basato sul feedback degli utenti per generare una stringa di connessione Airflow nella sequenza corretta e aggiunto il parametro port in. Configurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager • È stato aggiunto un passaggio per installare un pacchetto di decompressione locale in base al feedback degli utenti in. Creazione di un plugin personalizzato con Oracle	23 luglio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un argomento per gli operatori AWS DMS all'indirizzoDomande frequenti su Amazon MWAA. • È stato aggiunto un argomento di risoluzione dei problemi relativi a un errore di registro remoto in. Amazon Managed Workflows for Apache Airflow • `variables set`Spostato nei comandi CLI Airflow non supportati in. Riferimento ai comandi CLI di Apache Airflow	16 luglio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Sono stati aggiunti passaggi sequenziali per creare un file requirements.txt in base al feedback degli utenti all'indirizzoInstallazione delle dipendenze in Python. • Sono stati aggiunti passaggi sequenziali per creare un file plugins.zip in base al feedback degli utenti all'indirizzo. Installazione di plugin personalizzati • Sono stati aggiunti collegamenti incrociati nella guida per l'utente alla guida di riferimento delle API nella guida di riferimento delle API di Amazon Managed Workflows for Apache Airflow. • È stato aggiunto un argomento sul motivo per cui i plugin non vengono visualizzati nel menu Amministrazione > Plugins di Airflow 2.0 all'indirizzo. Domande frequenti su Amazon MWAA	9 luglio 2021

Nuove guide	Sono state aggiunte le seguenti modifiche: • CreatoEliminazione di file su Amazon S3.	9 luglio 2021
Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un elenco di valori supportati inUtilizzo di chiavi gestite dal cliente per la crittografia. • È stato aggiornato e chiarito l'esempio di URL di un repository privato basato sul feedback degli utenti in. Gestione delle dipendenze Python in requirements.txt	2 luglio 2021
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto il codice di esempio di Apache Airflow v1.10.12 per utilizzare una chiave privata AWS Secrets Manager per una connessione SSH all'indir izzo. Creazione di una connessione SSH utilizzan do `SSHOperator`	2 luglio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Aggiunte StartedTaskInstances e FinishedTaskInstances metriche a. Metriche di ambiente Apache Airflow v2 in CloudWatch	25 giugno 2021
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto il codice di esempio di Apache Airflow v2.0.2 all'indirizzo. Utilizzo di Amazon MWAA con Amazon EKS	25 giugno 2021
Nuove guide	Sono state aggiunte le seguenti modifiche: • CreatoOttimizzazione delle prestazioni per Apache Airflow su Amazon MWAA.	25 giugno 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Aggiunto `connections add` e `connections delete` ai comandi CLI Apache Airflow v2.0.2 supportati all'indirizzo. Riferimento ai comandi CLI di Apache Airflow • È stato aggiunto che l'ultima versione disponibile in AWS CloudFormation è Apache Airflow v2.0.2 all'indirizzo. Tutorial di avvio rapido per Amazon Managed Workflows for Apache Airflow • È stata aggiunta una domanda per l'archiviazione di dati temporanei su Apache Airflow Workers to. Domande frequenti su Amazon MWAA • È stato aggiunto un argomento per l'errore «Executor riporta l'istanza dell'attività %s completata». Amazon Managed Workflows for Apache Airflow • È stato aggiunto un argomento per il log a cui il server ha chiuso la connessione in modo imprevisto. Amazon	18 giugno 2021

Managed Workflows for Apache Airflow

- È stato aggiunto un esempio per eseguire comandi CLI su un tunnel SSH verso un bastion host to. Creazione di un token CLI di Apache Airflow
- È stato aggiunto un argomento per i nomi utente generati casualmente a. Amazon Managed Workflows for Apache Airflow
- È stato aggiunto un argomento per un errore 503 durante l'esecuzione di un DAG nella CLI a. Amazon Managed Workflows for Apache Airflow
- È stato aggiunto un argomento per i plug-in personalizzati in Apache Airflow v2.0.2 che richiedono un'opzione di configurazione Airflow attiva `core.lazy_load_plugins : False` per caricare i plug-in all'inizio di ogni processo Airflow per sovrascrivere l'impostazione predefinita della versione. Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA

- È stato aggiunto il passaggio relativo alle opzioni di configurazione Airflow per il codice di esempio dei plugin Apache Airflow v2.0.2 all'indirizzo. Creazione di un plugin personalizzato con Apache Hive e Hadoop
- È stato aggiunto il passaggio relativo alle opzioni di configurazione Airflow per il codice di esempio dei plugin Apache Airflow v2.0.2 all'indirizzo. Creazione di un plugin personalizzato che genera variabili di ambiente di runtime
- È stato aggiunto il passaggio relativo alle opzioni di configurazione Airflow per il codice di esempio dei plugin Apache Airflow v2.0.2 all'indir izzo. Creazione di un plugin personalizzato per Apache AirflowPythonVirtu alenvOperator
- È stato aggiunto il passaggio relativo alle opzioni di configurazione Airflow per il codice di esempio dei plugin Apache Airflow v2.0.2 all'indirizzo.

	Creazione di un plugin personalizzato con Oracle	
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un codice di esempio per una connessione Apache Airflow Snowflake all'indirizzo. Utilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow Snowflake	18 giugno 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stata aggiunta una guida alla crittografia lato server a. Creare are are are are are are are are are are are are are are are are are are are are are are • È stato aggiunto il backend segreto per Apache Airflow v2.0.2 a. Configurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager • È stata aggiunta una domanda per le richieste di aumento della quota di Apache Airflow Workers a. Domande frequenti su Amazon MWAA • È stata aggiunta una domanda per le metriche utilizzate per determina re se scalare Apache Airflow Workers. Domande frequenti su Amazon MWAA • È stata aggiunta una domanda per la creazione di metriche personalizzate in. CloudWatch Domande frequenti su Amazon MWAA • Sono stati aggiunti passaggi per abilitare gli indirizzi IP	2 giugno 2021

	privati per un endpoint di interfaccia VPC Amazon S3 per un VPC con routing privato in ingresso. Creazione degli endpoint del servizio VPC richiesti in un Amazon VPC con routing privato • È stata aggiunta un'opzione per configurare un tunnel SSH utilizzando il port forwarding locale in. Tutorial: configurazione dell'accesso alla rete privata utilizzando un host Linux Bastion	
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un codice di esempio per un DAG che esegue query sul database di metadati PostgreSQL di Amazon Aurora e pubblica metriche personalizzate su Amazon all'indirizzo. CloudWatch Utilizzo di un DAG per scrivere metriche personalizzate inCloudWatch	2 giugno 2021

Nuove guide	Sono state aggiunte le seguenti modifiche: • Ha creato una guida su come utilizzare i modelli di connessione in modo intercambiabile nell'interfaccia utente di Apache Airflow in. Panoramica dei tipi di connessione	2 giugno 2021
Correzioni	Sono state aggiunte le seguenti modifiche: • Aggiunti gli endpoint VPC Apache Airflow al AWS CloudFormation modello nell'opzione tre: creazione di una rete VPC senza accesso a Internet. Crea la rete VPC	2 giugno 2021

Lancio di Apache Airflow v2.0.2	Lancio della disponibilità generale di Apache Airflow v2.0.2. • Creato. Versioni di Apache Airflow su Amazon Managed Workflows per Apache Airflow • CreatoMetriche di ambiente Apache Airflow v2 in CloudWatch. • Aggiunti collegamenti specifici per la versione di Apache Airflow v2.0.2 a. Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA • Sono state aggiunte linee guida specifiche per la versione di Apache Airflow v2.0.2 a. Installazione delle dipendenze in Python • È stata aggiunta una guida specifica per la versione di Apache Airflow v2.0.2 a. Gestione delle dipendenze Python in requirements.txt • Aggiunti i plugin di esempio di Apache Airflow v2.0.2 a. Installazione di plugin personalizzati • È stato aggiunto il codice di esempio Apache Airflow v2.0.2 a. Pulizia del database Aurora PostgreSQ	26 maggio 2021

L in un ambiente Amazon MWAA

- È stato aggiunto il codice di esempio Apache Airflow v2.0.2 a. Utilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow
- È stato aggiunto il codice di esempio Apache Airflow v2.0.2 a. Creazione di un plugin personalizzato per Apache AirflowPythonVirtualenvOperator
- Aggiunti i comandi Apache Airflow v2.0.2 a. Riferimento ai comandi CLI di Apache Airflow
- Aggiunti gli script Apache Airflow v2.0.2 a. Creazione di un token CLI di Apache Airflow
- È stata aggiunta una nota secondo cui Amazon MWAA utilizza la versione più recente di Apache Airflow per impostazione predefinita. Crea un ambiente Amazon MWAA

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Sono state aggiunte indicazioni per la risoluzione dei problemi relativi alle attività di Airflow bloccate o non in esecuzione. Amazon Managed Workflows for Apache Airflow	14 maggio 2021
Correzioni	Sono state aggiunte le seguenti modifiche: • Abbiamo aggiornato il codice dei plugin di esempio per utilizzare l'ultima versione di Java inCreazione di un plugin personalizzato con Apache Hive e Hadoop. In precedenza, lo era`os.environ["JAVA_HOME"]="/usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.272.b10-1.amzn2.0.1.x86_64"` .	12 maggio 2021
Argomenti rimossi o spostati	Sono state aggiunte le seguenti modifiche: • Gli argomenti sono stati spostati in Amazon Managed Workflows for Apache Airflow nuove pagine per categoria.	10 maggio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stata aggiunta una panoramica dei bucket Amazon S3 a. Utilizzo dei DAG su Amazon MWAA	10 maggio 2021
Argomenti rimossi o spostati	Sono state aggiunte le seguenti modifiche: • È stato Accesso ad Apache Airflow spostato nella barra di navigazione di primo livello e ha aggiunto pagine per Creare un token di accesso al server web Apache AirflowCreazione di un token CLI di Apache Airflow, eRiferimento ai comandi CLI di Apache Airflow.	7 maggio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Sono stati aggiunti collegamenti specifici per la versione alla guida di riferimento di Apache Airflow per tutti i comandi CLI Airflow supportati e non supportati in. Riferimento ai comandi CLI di Apache Airflow • Sono stati aggiunti collegamenti specifici per la versione alla guida di riferimento di Apache Airflow per tutte le opzioni di configurazione in. Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA • È stata aggiunta l'utilità Amazon MWAA CLI a. Gestione delle dipendenze Python in requirements.txt	7 maggio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Sono stati aggiunti esempi semplici e annidati su come strutturare un plugins.zip inInstallazione di plugin personalizzati. • È stata aggiunta l'utilità CLI Amazon MWAA alle pagineAggiunta o aggiornamento di DAG, Installazione di plugin personalizzati e. Installazione delle dipendenze in Python • Sezioni di contenuto ristrutturate in una panoramica, caricate su Amazon S3 e installate su Amazon MWAA in base al feedback Installazione di plugin personalizzati degli utenti in e nelle pagine. Installazione delle dipendenze in Python • È stato aggiunto un caso d'uso di esempio per creare e collegare gli endpoint VPC richiesti a un Amazon VPC esistente senza accesso a Internet. Informazioni sulla rete su Amazon MWAA	30 aprile 2021

Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un codice di esempio che utilizza una chiave segreta in Secrets Manager per una variabile Apache Airflow in. Utilizzo di una chiave segreta inAWS Secrets Managerper una variabile Apache Airflow	30 aprile 2021
Nuove guide	Sono state aggiunte le seguenti modifiche: • CreatoCreazione degli endpoint del servizio VPC richiesti in un Amazon VPC con routing privato.	30 aprile 2021
Correzioni	Sono state aggiunte le seguenti modifiche: • Ops! Abbiamo aggiornato `core.default_ui_timezone` a `webserver.default_ui_timezone` inUtilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA.	30 aprile 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Aggiunti i passaggi Windows (PuTTY) per il tunnel SSH a. Tutorial: configurazione dell'accesso alla rete privata utilizzando un host Linux Bastion • È stato aggiunto un argomento per `apache-airflow-providers-amazon` , compatibile solo con Apache Airflow 2.0 to. Amazon Managed Workflows for Apache Airflow	23 Aprile 2021
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un codice di esempio che utilizza una chiave segreta in Secrets Manager per una connessione Apache Airflow in. Utilizzo di una chiave segreta inAWS Secrets Managerper una connessione Apache Airflow	23 Aprile 2021

Nuove guide	Sono state aggiunte le seguenti modifiche: • CreatoInformazioni sulla rete su Amazon MWAA. • CreatoSicurezza nel tuo VPC su Amazon MWAA. • CreatoGestione dell'accesso agli endpoint Amazon VPC specifici del servizio su Amazon MWAA.	23 Aprile 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un nuovo AWS CloudFormation modello per creare una rete Amazon VPC senza accesso a Internet. Crea la rete VPC • È stato aggiunto un nuovo tutorial per creare un AWS Client VPN inTutorial: Configurazione dell'accesso alla rete privata utilizzando unAWS Client VPN. • Il nome della pagina di accesso alla rete è stato modificato in modalità di accesso ad Apache Airflow in base al feedback degli utenti e ha semplificato i documenti. Modalità di accesso Apache Airflow • Documenti semplificati per includere solo le informazioni introduttive e i modelli di Amazon VPC basati sul feedback degli utenti. Crea la rete VPC • È stata aggiunta una soluzione alternativa per BigQuery l'operatore a. Amazon Managed Workflows for Apache Airflow	16 aprile 2021

	• È stata aggiunta una procedura consigliata per il file dei vincoli di Apache Airflow v1.10.12 a. Installazione delle dipendenze in Python	
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un codice di esempio per creare un plug-in personalizzato utilizzando Oracle inCreazione di un plugin personalizzato con Oracle. • È stato aggiunto un codice di esempio per creare un plug-in personalizzato che genera variabili di ambiente di runtime inCreazione di un plugin personalizzato che genera variabili di ambiente di runtime. •	16 aprile 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un argomento per il requisito della regola di autorefer enziazione su un gruppo di sicurezza VPC a. Domande frequenti su Amazon MWAA • Aggiunti limiti di directory e dimensioni dei plugin personalizzati a. Installaz ione di plugin personalizzati • È stata aggiunta la directory dei requisiti e i limiti di dimensione a. Installazione delle dipendenze in Python • Sono state chiarite le opzioni di configurazione di Apache Airflow per `foo.user` e in. `foo.pass` Gestione delle dipendenze Python in requirements.txt • È stata aggiunta una panoramica delle opzioni di configurazione a. Utilizzo delle opzioni di configura zione di Apache Airflow su Amazon MWAA	9 aprile 2021

Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un codice di esempio per creare un plug-in personalizzato utilizzando PythonVirtualenvOperator in Creazione di un plugin personalizzato per Apache AirflowPythonVirtualenvOperator. • È stato aggiunto un codice di esempio per creare un plug-in personalizzato con Apache Hive e Hadoop. Creazione di un plugin personalizzato con Apache Hive e Hadoop	9 aprile 2021
Correzioni	Sono state aggiunte le seguenti modifiche: • Ops! Abbiamo aggiornato il formato per un requirements.txt e aggiunto un esempio compatibile con Apache Airflow v1.10.12 in versione. Installazione delle dipendenze in Python	31 marzo 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stata aggiunta una soluzione alternativa per rimuovere un file requirements.txt o plugins.zip aDomande frequenti su Amazon MWAA. • È stata aggiunta una soluzione alternativa bash per SSH in un ambiente a. Domande frequenti su Amazon MWAA • È stato aggiunto un argomento di errore a CloudTrail ResourceAlreadyExistsException . Amazon Managed Workflows for Apache Airflow	26 marzo 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto l'elenco dei AWS servizi utilizzati perRuolo di esecuzione di Amazon MWAA. • È stato aggiunto l'elenco dei AWS servizi utilizzati perRuolo collegato ai servizi per Amazon MWAA. • È stata aggiunta una domanda per la versione Python 3.7 per Amazon MWAA a. Domande frequenti su Amazon MWAA • È stata aggiunta una domanda per. PythonVir tualenvOperator Domande frequenti su Amazon MWAA • È stato aggiunto lo script di risoluzione dei problemi come passaggi successivi per tutti gli argomenti relativi al VPC e alla configura zione dell'ambiente all'indir izzo. Amazon Managed Workflows for Apache Airflow • Sono stati chiariti i documenti secondo cui un bastione di Linux deve trovarsi nella stessa regione di un ambiente. Tutorial: configurazione dell'accesso	19 marzo 2021

	alla rete privata utilizzando un host Linux Bastion	
Nuove guide	Sono state aggiunte le seguenti modifiche: • Creata la guida alle connessioni Apache Airflow per AWS Secrets Manager at. Configurazione di una connessione Apache Airflow utilizzando un segreto AWS Secrets Manager • Tutorial di avvio rapido creato utilizzando un AWS CloudFormation modello per creare l'infrastruttura Amazon VPC, il bucket Amazon S3 e l'ambiente Amazon MWAA su. Tutorial di avvio rapido per Amazon Managed Workflows for Apache Airflow	19 marzo 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto l'argomento per la risoluzione dei problemi relativi alla creazione di bucket Amazon S3. Amazon Managed Workflows for Apache Airflow • Sono stati aggiunti passaggi per creare e allegare una policy JSON. Ruolo di esecuzione di Amazon MWAA	12 marzo 2021
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un codice di esempio per aggiungere una configurazione quando si attiva un DAG a. Accesso ad Apache Airflow	12 marzo 2021
Nuove guide	Sono state aggiunte le seguenti modifiche: • Guida alle migliori pratiche creata all'indirizzoGestione delle dipendenze Python in requirements.txt.	12 marzo 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto l'argomento per la risoluzione dei problemi di BigQuery Google/GCP/ a. Amazon Managed Workflows for Apache Airflow • È stato aggiunto l'argomento per la risoluzione dei problemi di Cython a. Amazon Managed Workflows for Apache Airflow • È stato aggiunto l'argomento per la risoluzione dei problemi di MySQL a. Amazon Managed Workflows for Apache Airflow • È stato aggiunto l'argomento di risoluzione degli errori del server web 5xx a. Amazon Managed Workflows for Apache Airflow	5 marzo 2021

Ora supportato	Sono state aggiunte le seguenti modifiche: • In precedenza, non `backend_kwargs` era supportato per AWS Secrets Manager ed era necessaria una soluzione alternativa per sovrascrivere la chiamata alla funzione Secrets Manager. Ora `backend_kwargs` è supportata. Vedi l'argomento AWS Secrets Manager sulla risoluzione dei problemi in Amazon Managed Workflows for Apache Airflow.	4 marzo 2021
Correzioni	Sono state aggiunte le seguenti modifiche: • Ops! Abbiamo aggiornato la dimensione di ogni classe di ambiente per riflettere i GB effettivi in Configurazione della classe di ambiente Amazon MWAA GB.	4 marzo 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto l'accesso alla rete privata utilizzan do una policy di endpoint VPC a. Modalità di accesso Apache Airflow • Sono stati aggiunti ulteriori controlli per la risoluzione dei problemi relativi alla creazione di un ambiente. Amazon Managed Workflows for Apache Airflow • Sono stati aggiunti passaggi per visualizzare i log `requirements.txt` perInstallazione delle dipendenze in Python.	26 febbraio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Aggiunto il caso d'uso Apache Hive a. Installazione delle dipendenze in Python • È stato chiarito nei documenti che le dipendenz e richieste per un pacchetto Apache Airflow devono essere incluse nel file all'indirizzo. `requireme nts.txt` Installazione delle dipendenze in Python • È stato aggiunto l'argomen to di risoluzione dei problemi di Updating requireme nts.txt a. Amazon Managed Workflows for Apache Airflow	25 febbraio 2021
Nuovi tutorial	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un tutorial sulla rete privata aTutorial: configurazione dell'accesso alla rete privata utilizzando un host Linux Bastion.	22 febbraio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Sono state aggiunte configurazioni di rete pubbliche e private aModalità di accesso Apache Airflow. • Sono stati aggiunti casi d'uso e scenari utente per gruppi di sviluppo aRuolo di esecuzione di Amazon MWAA.	22 febbraio 2021
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • Sono stati aggiunti script Python di esempio per il token di accesso Web e il token CLI a. Accesso ad Apache Airflow • È stato aggiunto un codice di esempio per attivare DAG in un altro ambiente. Esempi di codice per Amazon Managed Workflows for Apache Airflow • È stato aggiunto un codice di esempio per attivare DAG utilizzando una funzione Lambda. Invocare DAG con una funzione Lambda	22 febbraio 2021

Nuovi comandi e procedure	Sono state aggiunte le seguenti modifiche: • Sono state aggiunte procedure dettagliate a tutti gli script inAccesso ad Apache Airflow.	22 febbraio 2021
Nuovo codice di esempio	Sono state aggiunte le seguenti modifiche: • Esempio curl aggiornato per il token di accesso web all'indirizzoAccesso ad Apache Airflow. • È stato aggiunto un codice di esempio a cui connettersi a un Amazon RDS Microsoft SQL Server. Utilizzo di Amazon MWAA con Amazon RDS per Microsoft SQL Server	17 febbraio 2021

Nuovi comandi e procedure	Sono state aggiunte le seguenti modifiche: • Aggiunti AWS CLI comandi alle Utilizzo dei DAG su Amazon MWAA pagine. • Apache Airflow non supporta i DAG serializzati nei comandi CLI. Poiché la CLI viene eseguita sul server Web, che non dispone di plug-in o requisiti per motivi di sicurezza, qualsiasi ambiente MWAA con plugins.zip o requirements.txt non supporterà questi comandi. Apache Airflow `list_dags` e i comandi sono stati spostati in comandi non supportati in. `backfill` Accesso ad Apache Airflow	17 febbraio 2021
GitHub avviare	I documenti della guida per l'utente sono ora open source su GitHub. Scegli «Modifica questa pagina su GitHub» su qualsiasi pagina.	17 febbraio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stata aggiunta una domanda per il caso d'uso Step Functions v. Amazon MWAA a. Amazon Managed Workflows for Apache Airflow • È stata aggiunta la politica di accesso CLI a. Accesso a un ambiente Amazon MWAA • Ha chiarito i documenti in cui è possibile specifica re qualsiasi opzione di configurazione Apache Airflow supportata in. Utilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA • Ha chiarito i documenti che se un container Fargate in una zona di disponibilità si guasta, MWAA passa all'altro contenitore in una zona di disponibilità diversa all'indirizzo. Crea la rete VPC	12 febbraio 2021
Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • Aggiunto Configurazione della classe di ambiente Amazon MWAA.	5 febbraio 2021

Argomenti rimossi o spostati	Sono state aggiunte le seguenti modifiche: • È stato rimosso il requisito per cui il nome del bucket Amazon S3 deve iniziare da. `airflow-` Inizia a usare Amazon Managed Workflows per Apache Airflow • Spostato Accesso a un ambiente Amazon MWAA e Ruolo di esecuzione di Amazon MWAA ripreso. Gestione dell'accesso a un ambiente Amazon MWAA	4 febbraio 2021
Amazon MWAA CloudFormation	Aggiorna i parametri per creare un ambiente in Amazon MWAA CloudFormation. • Rimuovi. SubnetList • Rimuovi TagList. • Aggiungi NetworkConfiguration. • Aggiungi TagMap. • Aggiungi esempi di richieste di creazione di ambiente.	4 febbraio 2021

Nuovi argomenti e casi d'uso	Sono state aggiunte le seguenti modifiche: • È stato aggiunto un esempio di configurazione e-mail aUtilizzo delle opzioni di configurazione di Apache Airflow su Amazon MWAA. • È stato aggiunto un argomento di PostgresH ook risoluzione dei problemi aAmazon Managed Workflows for Apache Airflow. • È stato aggiunto un argomento di AWS Secrets Manager risoluzione dei problemi aAmazon Managed Workflows for Apache Airflow. • È stato aggiunto un caso d'uso ad alte prestazio ni aConfigurazione della scalabilità automatica dei lavoratori Amazon MWAA.	29 gennaio 2021
Lancio di Amazon MWAA	Lancio della disponibilità generale di Amazon Managed Workflows per Apache Airflow. • Guida per l'utente e documentazione • AWS CloudFormation documentazione	24 novembre 2020

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.