Leitfaden

AWS CodeCommit

API-Version 2015-04-13

AWS CodeCommit: Leitfaden

Table of Contents

Was ist CodeCommit? 1
Einführung von CodeCommit 2
CodeCommit, Git und die Auswahl des richtigenAWS Dienstes für Ihre Bedürfnisse 3
Wie funktioniert CodeCommit ? 7
WieCodeCommit unterscheidet sich die Dateiversionierung in Amazon S3? 9
Erste Schritte mit CodeCommit 9
Wo kann ich mehr über Git erfahren? 10
Einrichtung 11
Ihre Anmeldeinformationen einsehen und verwalten 11
Einrichten mit Git-Anmeldeinformationen 12
Einrichten mit anderen Methoden 13
Kompatibilität für CodeCommit, Git und anderen Komponenten 15
Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden 16
Schritt 1: Erstkonfiguration für CodeCommit 16
Schritt 2: Installieren von Git 17
Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit 18
Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository 20
Nächste Schritte 22
Für HTTPS-Verbindungen mitgit-remote-codecommit 22
Schritt 0: Installieren Sie die Voraussetzungen fürgit-remote-codecommit 23
Schritt 1: Erstkonfiguration fürCodeCommit 24
Schritt 2: Installierengit-remote-codecommit 28
Schritt 3: Stellen Sie eine Verbindung zum herCodeCommitkonsole und klone das Repository 29
Nächste Schritte 30
Für Verbindungen von Entwicklungstools 31
Integration von AWS Cloud9 in AWS CodeCommit 34
Integrieren Sie Visual Studio mitAWS CodeCommit 39
Integration von Eclipse in AWS CodeCommit 40
Für SSH-Benutzer, die dieAWS CLI 48
Schritt 1: Verknüpfen Sie Ihren öffentlichen Schlüssel mit dem IAM-Benutzer 49
Schritt 2: Fügen Sie CodeCommit zu Ihrer SSH-Konfiguration hinzu 50
Nächste Schritte 50

Für SSH-Verbindungen unter Linux, macOS oder Unix ... 51
Schritt 1: Erstkonfiguration fürCodeCommit ... 51
Schritt 2: Installieren von Git ... 52
Schritt 3: Konfiguration von Anmeldeinformationen unter Linux, macOS oder Unix ... 53
Schritt 4: Connect derCodeCommit Konsole und Klonen des Repositorys ... 57
Nächste Schritte ... 59
Für SSH-Verbindungen unter Windows ... 59
Schritt 1: Anfängliche Konfiguration für CodeCommit ... 59
Schritt 2: Installieren Sie Git ... 61
Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein ... 61
Schritt 4: Connect Sie sich mit der CodeCommit-Konsole und klonen Sie das Repository ... 66
Nächste Schritte ... 67
Für HTTPS-Verbindungen unter Linux, MacOS oder Unix mit demAWS CLICredential-Helper ... 68
Schritt 1: Erstkonfiguration fürCodeCommit ... 68
Schritt 2: Installieren von Git ... 72
Schritt 3: Richten Sie den Credential Helper ein ... 73
Schritt 4: Stellen Sie eine Verbindung mit dem herCodeCommitkonsole und klone das Repository ... 75
Nächste Schritte ... 77
Für HTTPS-Verbindungen unter Windows mit demAWS CLICredential-Helper ... 77
Schritt 1: Erstkonfiguration fürCodeCommit ... 78
Schritt 2: Installieren von Git ... 82
Schritt 3: Richten Sie den Credential Helper ein ... 82
Schritt 4: Stellen Sie eine Verbindung mit dem herCodeCommitkonsole und klone das Repository ... 85
Nächste Schritte ... 86
Erste Schritte ... 87
Erste Schritte mit CodeCommit ... 87
Voraussetzungen ... 89
Schritt 1: Erstellen eines CodeCommit-Repositorys ... 89
Schritt 2: Fügen Sie Dateien zu Ihrem Repository hinzu ... 92
Schritt 3: Durchsuchen Sie den Inhalt Ihres Repositorys ... 94
Schritt 4: Einen Pull-Request erstellen und gemeinsam bearbeiten ... 99
Schritt 5: Bereinigen ... 106
Schritt 6: Nächste Schritte ... 107
Erste Schritte mit Git und CodeCommit ... 107

Schritt 1: Erstellen Sie ein Repository CodeCommit 109
Schritt 2: Erstellen Sie ein lokales Repo 110
Schritt 3: Erstelle deinen ersten Commit 112
Schritt 4: Push deinen ersten Commit 114
Schritt 5: Teile das CodeCommit Repository und führe einen weiteren Commit per Push und Pull durch 114
Schritt 6: Einen Branch erstellen und teilen 117
Schritt 7: Erstelle und teile ein Tag 119
Schritt 8: Zugriffsberechtigungen einrichten 120
Schritt 9: Bereinigen 124
Produkt- und Service-Integrationen 126
Integration mit anderen AWS Diensten 126
Integrationsbeispiele der Community 136
Blog-Posts 136
Codebeispiele 140
Arbeiten mit Repositorien 141
Erstellen eines -Repositorys 142
Erstellen Sie ein Repository (Konsole) 143
Erstellen Sie ein Repository ()AWS CLI 145
Herstellen einer Verbindung mit einem Repository 147
Voraussetzungen für die Verbindung mit einem CodeCommit Repository 148
Connect zum CodeCommit Repository her, indem Sie das Repository klonen 149
Ein lokales Repo mit dem CodeCommit Repository Connect 151
Teilen Sie ein Repository 152
Wählen Sie das Verbindungsprotokoll, das Sie mit Ihren Benutzern teilen möchten 153
Erstelle IAM-Richtlinien für dein Repository 155
Erstellen Sie eine IAM-Gruppe für Repository-Benutzer 156
Teilen Sie die Verbindungsinformationen mit Ihren Benutzern 157
Benachrichtigungen für Repository-Ereignisse konfigurieren 159
Verwenden von Repository-Benachrichtigungsregeln 161
Erstellen einer Benachrichtigungsregel 162
Benachrichtigungen ändern oder deaktivieren 165
Benachrichtigungen löschen 166
Markieren eines Repositorys 168
Hinzufügen eines Tags zu einem Repository 169
Anzeigen von Tags für ein Repository 171

Bearbeiten von Tags für ein Repository 173
Entfernen Sie ein Tag aus einem Repository 175
Verwalten von Auslösern für ein Repository 176
Erstellen Sie die Ressource und fügen Sie Berechtigungen hinzu für CodeCommit 177
Einen Auslöser für ein Amazon SNS SNS-Thema erstellen 178
Erstellen Sie einen Trigger für eine Lambda-Funktion 186
Erstellen Sie einen Trigger für eine bestehende Lambda-Funktion 191
Trigger für ein Repository bearbeiten 200
Testen Sie Trigger für ein Repository 202
Trigger aus einem Repository löschen 204
Ein Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben 207
Ordnen Sie dem CodeGuru Rezensenten ein Repository zu 209
Trennen Sie die Zuordnung eines Repositorys zu Reviewer CodeGuru 210
Repository-Details anzeigen 210
Repository-Details anzeigen (Konsole) 211
CodeCommit Repository-Details anzeigen (Git) 211
CodeCommit Repository-Details anzeigen (AWS CLI) 213
Ändern von Repository-Einstellungen 217
Repository-Einstellungen ändern (Konsole) 217
AWS CodeCommit Repository-Einstellungen ändern ()AWS CLI 219
Synchronisieren Sie Änderungen zwischen Repositorys 221
Senden Sie Commits an zwei Repositorys 223
Konfigurieren Sie den kontoübergreifenden Zugriff auf ein Repository mithilfe von Rollen 228
Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountA 229
Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountB 233
Kontoübergreifender Repository-Zugriff: Aktionen für den Repository-Benutzer in AccountB 235
Löscht ein Repository 241
Löschen Sie ein CodeCommit Repository (Konsole) 242
Löscht ein lokales Repo 243
Löscht ein CodeCommit Repository ()AWS CLI 243
Arbeiten mit Dateien 245
Durchsuchen Sie Dateien in einem Repository 246
Durchsuchen einesCodeCommitEndlager 247

Eine Datei erstellen oder hinzufügen ... 248
Erstellen oder Hochladen einer Datei (Konsole) ... 249
Hinzufügen einer Datei (AWS CLI) ... 251
Hinzufügen einer Datei (Git) ... 252
Bearbeiten Sie den Inhalt einer Datei ... 252
Bearbeiten von eingebundenen Dateien (Konsole) ... 253
Eine Datei bearbeiten oder löschen (AWS CLI) ... 254
Eine Datei bearbeiten (Git) ... 257
Verwenden von Pull-Anforderungen ... 258
Erstellen einer Pull-Anforderung ... 262
Erstellen einer Pull-Anforderung (-Konsole) ... 262
Erstellen Sie eine Pull-Anforderung (AWS CLI) ... 264
Erstellen einer Genehmigungsregel ... 266
Erstellen einer Pull-Anforderung ... 267
Erstellen einer Genehmigungsregel für eine Pull-Anforderung (AWS CLI) ... 270
Anzeigen von Pull-Anforderungen ... 271
Anzeigen von Pull-Anforderungen (Konsole) ... 272
Anzeigen von Pull-Anforderungen (AWS CLI) ... 273
Überprüfen einer Pull-Anforderung ... 277
Überprüfen Sie eine Pull-Anfrage (Konsole) ... 278
Überprüfen Sie Pull-Requests (AWS CLI) ... 283
Aktualisieren einer Pull-Anforderung ... 289
Aktualisieren einer Pull-Anforderung (-Konsole) ... 289
Aktualisieren von Pull-Anforderungen (AWS CLI) ... 290
Bearbeiten oder Löschen einer Genehmigungsregel ... 293
Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung ... 293
Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung (AWS CLI) .. 295
Überschreiben von Genehmigungsregeln für eine Pull-Anforderung ... 297
Genehmigungsregeln außer Kraft setzen (Konsole) ... 298
Überschreiben von Genehmigungsregeln (AWS CLI) ... 299
Zusammenführen einer Pull-Anforderung ... 300
Zusammenführen einer Pull-Anforderung (-Konsole) ... 301
Zusammenführen einer Pull-Anforderung (AWS CLI) ... 304
Lösen von Konflikten in einer Pull-Anforderung ... 311
Konflikte in einer Pull-Request (Konsole) lösen ... 312
Lösen von Konflikten in einer Pull-Anforderung (AWS CLI) ... 314

Schließen einer Pull-Anforderung 323
Schließen einer Pull-Anforderung (-Konsole) 323
Schließen einer Pull-Anforderung (AWS CLI) 324
Arbeiten mit Genehmigungsregelvorlagen 327
Erstellen einer Genehmigungsregelvorlage 329
So erstellen Sie eine Genehmigungsregelvorlage (Konsole) 329
So erstellen Sie eine Genehmigungsregelvorlage (AWS CLI) 334
Eine Genehmigungsregelvorlage einem Repository zuordnen. 336
Eine Genehmigungsregelvorlage zuordnen (Konsole) 336
Eine Genehmigungsregelvorlage zuordnen (AWS CLI) 337
Verwalten von Genehmigungsregelvorlagen 338
Vorlagen für Genehmigungsregeln verwalten (Konsole) 338
Vorlagen für Genehmigungsregeln verwalten (AWS CLI) 339
So trennen Sie eine Genehmigungsregelvorlage. 343
So trennen Sie eine Genehmigungsregelvorlage (Konsole) 344
Eine Genehmigungsregelvorlage aufheben (AWS CLI) 344
Eine Genehmigungsregelvorlage löschen 346
Eine Genehmigungsregelvorlage löschen (Konsole) 346
Eine Genehmigungsregelvorlage löschen (AWS CLI) 346
Mit Commits arbeiten 348
Erstellen Sie einen Commit 349
Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI 350
Einen Commit mit einem Git-Client erstellen 351
Erstelle einen Commit dem AWS CLI 355
Anzeigen von Commit-Details 358
Durchsuche Commits in einem Repository 359
Commit-Details anzeigen (AWS CLI) 363
Commit-Details anzeigen (Git) 369
Vergleichen von Commits 372
Einen Commit mit seinem übergeordneten Commit vergleichen 372
Vergleichen von zwei beliebigen Commit-Spezifizierern 375
Kommentar zu einem Commit 378
Kommentare zu einem Commit in einem Repository anzeigen 378
Kommentare zu einem Commit in einem Repository hinzufügen und darauf antworten 379
Kommentare ansehen, hinzufügen, aktualisieren und beantworten ()AWS CLI 384
Erstellen Sie ein Git-Tag 393

Verwende Git, um ein Tag zu erstellen ... 394
Tag-Details anzeigen ... 395
Tag-Details anzeigen (Konsole) ... 395
Git-Tag-Details anzeigen (Git) ... 396
Löschen eines Tags ... 398
Verwende Git, um ein Git-Tag zu löschen ... 398
Arbeiten mit Zweigen ... 400
Erstellen eines Zweigs ... 402
Erstelle einen Branch (Konsole) ... 402
Einen Zweig erstellen (Git) ... 403
Erstelle einen Zweig (AWS CLI) ... 405
Beschränken Sie Pushs und Merges auf Branches ... 406
Konfigurieren Sie eine IAM-Richtlinie, um Pushs und Merges auf einen Branch zu beschränken ... 407
Wenden Sie die IAM-Richtlinie auf eine IAM-Gruppe oder -Rolle an ... 409
Testen Sie die Richtlinie ... 410
Filialdetails anzeigen ... 410
Branch-Details anzeigen (Konsole) ... 411
Filialdetails anzeigen (Git) ... 412
Details zur Filiale anzeigen (AWS CLI) ... 413
Zweige vergleichen und zusammenführen ... 414
Vergleichen Sie einen Branch mit dem Standard-Branch ... 415
Vergleichen Sie zwei spezifische Branchen ... 415
Zwei Zweige zusammenführen ()AWS CLI ... 416
Ändern Sie die Filialeinstellungen ... 419
Ändern Sie den Standardzweig (Konsole) ... 419
Ändern Sie den Standardzweig ()AWS CLI ... 420
Löschen Sie einen Zweig ... 421
Lösche einen Branch (Konsole) ... 422
Lösche einen Zweig (AWS CLI) ... 422
Einen Zweig löschen (Git) ... 423
Arbeiten mit Benutzereinstellungen ... 425
Migration zu CodeCommit ... 426
Migrieren Sie ein Git-Repository zu AWS CodeCommit ... 426
Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit ... 427
Schritt 1: Erstellen Sie ein Repository CodeCommit ... 433

Schritt 2: Das Repository klonen und zum CodeCommit Repository pushen ... 436
Schritt 3: Dateien anzeigen in CodeCommit ... 437
Schritt 4: Teilen Sie das Repository CodeCommit ... 438
Migrieren Sie Inhalte zu CodeCommit ... 441
Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit ... 442
Schritt 1: Erstellen Sie ein Repository CodeCommit ... 448
Schritt 2: Migrieren Sie lokale Inhalte in das CodeCommit Repository ... 449
Schritt 3: Dateien anzeigen in CodeCommit ... 451
Schritt 4: Teilen Sie das Repository CodeCommit ... 451
Migrieren Sie ein Repository schrittweise ... 454
Schritt 0: Stellen Sie fest, ob eine inkrementelle Migration erfolgen soll ... 455
Schritt 1: Installieren Sie die erforderlichen Komponenten und fügen Sie das CodeCommit Repository als Remote-Repository hinzu ... 456
Schritt 2: Erstellen Sie das Skript, das für die inkrementelle Migration verwendet werden soll ... 458
Schritt 3: Führen Sie das Skript aus und migrieren Sie schrittweise zu CodeCommit ... 458
Anhang: Beispielskript `incremental-repo-migration.py` ... 460
Sicherheit ... 468
Datenschutz ... 468
AWS KMS und Verschlüsselung ... 469
Rotieren der Anmeldeinformationen ... 472
Identitäts- und Zugriffsverwaltung ... 477
Zielgruppe ... 478
Authentifizierung mit Identitäten ... 479
Verwalten des Zugriffs mit Richtlinien ... 482
Authentifizierung und Zugriffskontrolle ... 485
Featuresweise von AWS CodeCommit mit IAM ... 559
Ressourcenbasierte CodeCommit-Richtlinien ... 560
Autorisierung auf der Grundlage von Tags CodeCommit ... 560
CodeCommit IAM-Rollen ... 564
Beispiele für identitätsbasierte Richtlinien ... 565
Fehlerbehebung ... 568
Ausfallsicherheit ... 571
Sicherheit der Infrastruktur ... 571
Überwachung CodeCommit ... 573
Überwachung CodeCommit Veranstaltungen ... 574

referenceCreated-Ereignis ... 575
referenceUpdated-Ereignis ... 576
referenceDeleted-Ereignis ... 576
unreferencedMergeCommitErstelltes ... 577
commentOnCommitErstelltes ... 578
commentOnCommitErstelltes ... 579
commentOnPullRequestCreated Veranstaltung ... 580
commentOnPullRequestUpdated Veranstaltung ... 581
pullRequestCreated Veranstaltung ... 582
pullRequestSourceBranchUpdated Veranstaltung ... 583
pullRequestStatusGeändertes Ereignis ... 584
pullRequestMergeStatusUpdated Veranstaltung ... 585
approvalRuleTemplateErstelltes ... 586
approvalRuleTemplateErstelltes ... 586
approvalRuleTemplateErstelltes ... 587
approvalRuleTemplateAssociatedWithRepository Veranstaltung ... 588
approvalRuleTemplateDisassociatedWithRepository Veranstaltung ... 589
approvalRuleTemplateBatchAssociatedWithRepositories Veranstaltung ... 590
approvalRuleTemplateBatchDisassociatedFromRepositories Veranstaltung ... 591
pullRequestApprovalRuleCreated Veranstaltung ... 591
pullRequestApprovalRuleDeleted Veranstaltung ... 593
pullRequestApprovalRuleOverridden Veranstaltung ... 594
pullRequestApprovalStateChanged Veranstaltung ... 596
pullRequestApprovalRuleUpdated Veranstaltung ... 598
ReactionCreated ... 599
ReactionUpdat ... 600
Protokollierung von AWS CodeCommit-API-Aufrufen mit AWS CloudTrail ... 601
CodeCommit Informationen in CloudTrail ... 601
Grundlagen zu CodeCommit -Protokolldateieinträgen ... 603
AWS CloudFormation-Ressourcen ... 611
CodeCommit- und AWS CloudFormation-Vorlagen ... 611
Beispiele für --- ... 612
AWS CloudFormation,CodeCommit, und derAWS Cloud Development Kit (AWS CDK) ... 613
Weitere Informationen zu AWS CloudFormation ... 614
Fehlerbehebung ... 615
Fehlerbehebung bei Git-Anmeldeinformationen (HTTPS) ... 615

Git-Anmeldeinformationen fürAWS CodeCommit: Bei der Verbindung zu meinem CodeCommit -Repository über Terminal oder Befehlszeile wird immer eine Eingabeaufforderung für Anmeldeinformationen angezeigt 616
Git-Anmeldeinformationen fürAWS CodeCommit: Ich habe Git-Anmeldeinformationen eingerichtet, aber mein System nutzt diese nicht 616
Fehlerbehebung bei git-remote-codecommit 617
Ich sehe einen Fehler: git: 'remote-codecommit' ist kein Git-Befehl 617
Ich sehe einen Fehler: fatal: Remote-Helfer für 'codecommit' konnte nicht gefunden werden 618
Klonfehler: Ich kann kein CodeCommit----Repository von einer IDE klonen 618
Push oder Pull-Fehler: Ich kann keine Commits per Push oder Pull von einer IDE in ein CodeCommit---Repository verschieben 618
Fehlerbehebung bei SSH-Verbindungen 619
Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen, aber die Verbindung schlägt auf Linux-, macOS- oder Unix-Systemen fehl 619
Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen und SSH wurde erfolgreich getestet, aber die Verbindung schlägt auf Windows-Systemen fehl 621
Authentifizierungsproblem: Bei der Verbindung mit einem CodeCommit-Repository kann die Authentizität des Hosts nicht festgestellt werden 621
IAM-Fehler: „Ungültiges Format“ beim Versuch, IAM einen öffentlichen Schlüssel hinzuzufügen 629
Ich muss darauf zugreifenCodeCommitRepositorys in mehreren Amazon Web Services-Konten mit SSH-Anmeldeinformationen 629
Git unter Windows: Bash-Emulator oder Befehlszeilenfenster stürzt bei Verbindungsversuchen mit SSH ab 630
Das Public-Key-Format erfordert in einigen Linux-Distributionen eine Spezifikation 631
Zugriffsfehler: Der öffentliche SSH-Schlüssel wurde verweigert, wenn eine Verbindung zu einem hergestellt wurdeCodeCommitEndlager 631
Fehlerbehebung beim Hilfsprogramm für Anmeldeinformationen (HTTPS) 632
Ich erhalte eine Fehlermeldung beim Ausführen des`git config`Befehl zum Konfigurieren des Helfers für Anmeldeinformationen 633
Zurückgegebene Fehlermeldung "Befehl nicht gefunden" unter Windows bei Verwendung des Hilfsprogramms für Anmeldeinformationen 633
Wenn ich eine Verbindung mit einem CodeCommit-Repository verbinde, werde ich zur Eingabe eines Benutzernamens aufgefordert 634

Git für macOS: Ich habe den Hilfsprogramm für Anmeldeinformationen erfolgreich konfiguriert, aber jetzt wird mir der Zugriff auf mein Repository verweigert (403) 635
Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert (403) 638
Fehlerbehebung bei Git-Clients 641
Git-Fehler: Fehler: RPC fehlgeschlagen; result=56, HTTP code = 200 fatal: Das Remote-Ende hing unerwartet auf 641
Git-Fehler: Zu viele Referenz-Update-Befehle 642
Git-Fehler: Push-Übertragung über HTTPS funktioniert in einigen Git-Versionen nicht 642
Git-Fehler: "gnutls_handshake() failed" 642
Git-Fehler: Git kann das CodeCommit--Repository nicht finden oder hat keine Zugriffsberechtigung für das Repository 643
Git unter Windows: Keine unterstützten Authentifizierungsmethoden verfügbar (publickey) .. 643
Behebung von Zugriffsfehlern 644
Zugriffsfehler: Ich werde zur Eingabe eines Benutzernamens und eines Kennworts aufgefordert, über Windows eine Verbindung mit einem CodeCommit-Repository herzustellen 644
Zugriffsfehler: Verweigerung des öffentlichen Schlüssels bei der Verbindung mit einem CodeCommit-Repository 645
Zugriffsfehler: Meldung „Quote überschritten“ oder „429" bei der Verbindung mit einem CodeCommit-Repository 645
Behebung von Konfigurationsfehlern 646
Konfigurationsfehler: Kann nicht konfiguriert werdenAWS CLIAnmeldeinformationen unter macOS 647
Behebung von Konsolenfehlern 647
Zugriffsfehler: Zugriff auf Verschlüsselungsschlüssel für CodeCommit-Repository verweigert (Konsole oder AWS CLI) 645
Verschlüsselungsfehler: Das Repository kann nicht entschlüsselt werden 648
Konsolenfehler: Der Code in einem CodeCommit Repository kann nicht von der Konsole aus durchsucht werden 648
Anzeigefehler: Eine Datei oder ein Vergleich zwischen Dateien kann nicht angezeigt werden 649
Fehlerbehebung bei Auslösern 649
Fehler auslösen: Ein Repository-Auslöser wird nicht zum erwarteten Zeitpunkt ausgeführt .. 649
Aktivieren des Debuggings 650
CodeCommit Referenz 652

Regionen und Git-Verbindungsendpunkte 652
Unterstützt AWS-Regionen für CodeCommit 653
Endpunkte für Git-Verbindungen 654
Server-Fingerabdrücke für CodeCommit 661
Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle 668
Verfügbarkeit 669
VPC-Endpoints erstellen für CodeCommit 670
Erstellen Sie eine VPC-Endpunktrichtlinie für CodeCommit 671
Kontingente 672
Befehlszeilenreferenz 680
Grundlegende Git-Befehle 686
Konfigurationsvariablen 687
Remote-Repositorys 687
Commits 689
Branches 691
Tags 693
Dokumentverlauf 694
Frühere Aktualisierungen 705
AWS-Glossar 713
..... dccxiv

Was ist AWS CodeCommit?

AWS CodeCommitist ein von Amazon Web Services gehosteter Service für die Versionskontrolle. Über diesen Service können Sie Komponenten (wie Dokumente, Quellcode und Binärdateien) privat in der Cloud speichern und verwalten. Weitere Informationen zu den CodeCommit-Preisen finden Sie unter Preise.

> **Note**
>
> CodeCommit ist im Rahmen vieler Compliance-Programmen zugelassen. Weitere Informationen zu AWS und Compliance-Bemühungen finden Sie unter Im Rahmen des Compliance-Programms zugelassene AWS-Services.
> Dies ist ein HIPAA-berechtigter Service. Weitere Informationen zu AWS, dem Health Insurance Portability and Accountability Act of 1996 (HIPAA) und die Verwendung von AWS-Services zur Verarbeitung, Speicherung und Übertragung geschützter Gesundheitsinformationen (PHI) finden Sie in der HIPAA-Übersicht.
> Informationen zu diesem Service und zu ISO 27001, einer Sicherheitsmanagementnorm, die bewährte Methoden für das Sicherheitsmanagement festlegt, finden Sie in der ISO 27001-Übersicht.
> Weitere Informationen zu diesem Service und dem Payment Card Industry Data Security Standard (PCI DSS) finden Sie unter Übersicht über PCI DSS.
> Informationen zu diesem Service und zum US-Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2, der die Sicherheitsanforderungen für kryptografische Module zum Schutz vertraulicher Informationen angibt, finden Sie unter Federal Information Processing Standard (FIPS) 140-2 – Übersicht und Endpunkte für Git-Verbindungen.

Themen

- Einführung von CodeCommit
- CodeCommit, Git und die Auswahl des richtigenAWS Dienstes für Ihre Bedürfnisse
- Wie funktioniert CodeCommit ?
- WieCodeCommit unterscheidet sich die Dateiversionierung in Amazon S3?
- Erste Schritte mit CodeCommit
- Wo kann ich mehr über Git erfahren?

Einführung von CodeCommit

CodeCommitist ein von sicheren, hochgradig skalierbaren, verwalteten Service für Quellüberwachung, der private Git-Repositorys hostet. CodeCommitmacht es überflüssig, dass Sie Ihr eigenes Quellcodeverwaltungssystem verwalten oder sich Gedanken über die Skalierung seiner Infrastruktur machen müssen. Mit CodeCommit können Sie beliebige Daten speichern – von Quellcode bis zu Binärdateien. Es unterstützt die Standardfunktionen von Git, sodass Sie Ihre vorhandenen Git-basierten Tools weiterhin problemlos nutzen können.

Mit CodeCommit haben Sie folgende Möglichkeiten:

- Profitieren Sie von einem vollständig verwalteten Service, der von gehostet wirdAWS. CodeCommitbietet eine hohe Serviceverfügbarkeit und Beständigkeit und macht den administrativen Aufwand für die Verwaltung Ihrer eigenen Hard- und Software überflüssig. Es ist nicht mehr nötig, Hardware bereitzustellen und zu skalieren oder Serversoftware zu installieren, zu konfigurieren und zu aktualisieren.
- Bewahren Sie Ihren Code sicher auf. CodeCommitRepositorys werden sowohl im Speicher als auch bei der Übertragung verschlüsselt.
- Arbeiten Sie gemeinsam an Code. CodeCommitRepositorys unterstützen Pull-Requests, bei denen Benutzer die Codeänderungen der anderen Benutzer überprüfen und kommentieren können, bevor sie sie zu Branches zusammenführen, Benachrichtigungen, die Benutzern automatisch E-Mails über Pull-Requests und Kommentare senden, und vieles mehr.
- Skalieren Sie Ihre Versionskontrollprojekte ganz einfach. CodeCommitRepositorys können an Ihre Entwicklungsanforderungen angepasst werden. Der Service kann Repositorys mit einer großen Anzahl an Dateien oder Branches, mit großen Dateien und mit umfangreichen Revisionsverläufen verarbeiten.
- Speichere alles, jederzeit. CodeCommithat keine Beschränkung der Größe der Repositorys oder der Dateitypen, der das Sie speichern.
- Integrieren Sie es mit anderen DienstenAWS und Diensten von Drittanbietern. CodeCommithält Ihre Repositorys in der Nähe Ihrer anderen Produktionsressourcen in derAWS Cloud, was dazu beiträgt, die Geschwindigkeit und Häufigkeit Ihres Entwicklungslebenszyklus zu erhöhen. Es ist in IAM integriert und kann mit anderenAWS Diensten und parallel zu anderen Repositorys verwendet werden. Weitere Informationen finden Sie unter Produkt- und Serviceintegrationen mit AWS CodeCommit.
- Migrieren Sie problemlos Dateien aus anderen Remote-Repositorys. Sie können aus jedem Git-basierten Repository Dateien in CodeCommit migrieren.

- Verwenden Sie die Git-Tools, die Sie bereits kennen. CodeCommitunterstützt Git-Befehle sowie eigeneAWS CLI Befehle und APIs.

CodeCommit, Git und die Auswahl des richtigenAWS Dienstes für Ihre Bedürfnisse

Als Git-basierter Service eignet sich CodeCommit optimal für die meisten Versionskontrollanforderungen. Es gibt keine willkürlichen Einschränkungen hinsichtlich Dateigröße, Dateityp und Repository-Größe. Es gibt jedoch inhärente Einschränkungen für Git, die insbesondere im Laufe der Zeit negative Auswirkungen auf die Leistung bestimmter Vorgänge haben können. Sie können potenzielle Leistungsminderungen des CodeCommit-Repositorys vermeiden, indem Sie es nach Möglichkeit nicht für Anwendungsfälle verwenden, in denen andere AWS-Services besser für die Aufgabe geeignet wären. Sie können die Git-Leistung auch für komplexe Repositorys optimieren. Hier finden Sie einige Anwendungsfälle, in denen Git und daher CodeCommit möglicherweise nicht die beste Lösung für Sie sind oder in denen Sie zur Optimierung von Git möglicherweise zusätzliche Schritte durchführen müssen.

Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services
Große Dateien mit häufigen Änderungen	Git verwendet zum Speichern von Unterschieden zwischen Dateiversionen Delta-Kod ierung. Beispiel: Wenn Sie ein paar Wörter in einem Dokument ändern, speichert Git nur diese geänderten Wörter. Bei Dateien oder Objekten über 5 MB mit vielen Änderungen muss Git möglicherweise eine große Kette von Delta-Unterschiede rekonstruieren. Dadurch kann eine zunehmende Menge an Datenverarbeitungs ressourcen sowohl auf Ihrem	Für die Versionierung von großen Dateien sollten Sie Amazon Simple Storage Service (Amazon S3). Weitere Informationen finden Sie unter Verwenden der Versionierung im Amazon Simple Storage Service-Benutzerhandbuch.

Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services
	lokalen Computer als auch in CodeCommit in Anspruch genommen werden, während diese Dateien im Laufe der Zeit an Umfang zunehmen.	
Datenbank	Git-Repositorys werden im Laufe immer größer. Da beim Versioning alle Änderungen verfolgt werden, nimmt die Größe des Repositorys mit jeder Änderung zu. Bei der Übermittlung von Daten mit Commit werden also selbst dann, wenn während der Übermittlung Daten gelöscht werden, Daten zu einem Repository hinzugefügt. Da im Laufe der Zeit mehr Daten verarbeitet und übertragen werden, wird Git immer langsamer. Dies wirkt sich insbesondere bei einem Datenbank-Anwendungsfall nachteilig aus. Git wurde nicht als Datenbank konzipiert.	Um eine Datenbank mit gleichbleibender Leistung unabhängig von der Größe zu erstellen und zu verwenden, sollten Sie Amazon DynamoDB in Betracht ziehen. Weitere Informationen finden Sie unter Amazon DynamoDB -Handbuch „Erste Schritte“.

Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services
Audit-Trails	In der Regel werden Audit-Trails über einen langen Zeitraum aufbewahrt und kontinuierlich von Systempro zessen in sehr häufigen Intervallen erstellt. Git ist auf das sichere Speichern von Quellcode ausgelegt, der von Gruppen von Entwicklern in einem Entwicklungszyklus erstellt wird. Bei sich schnell ändernden Repositorys, in denen programmgesteuerte Systemänderungen kontinuie rlich gespeichert werden, vermindert sich die Leistung im Laufe der Zeit.	Ziehen Sie Amazon Simple Storage Service (Amazon S3) an. Um dieAWS Aktivitäten zu überprüfen, sollten Sie je nach Anwendungsfall die Verwendung von AWS CloudTrailAWS Config, oder Amazon in Betracht ziehenClo udWatch.

Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services
Backups	Git wurde zum Versionin g des von Entwicklern geschriebenen Quellcode s konzipiert. Sie können als Sicherungsstrategie Commit-Übertragungen mit Push auf zwei Remote-Repositorys hochladen, darunter ein CodeCommit-Repository. Git ist jedoch nicht auf die Handhabung von Sicherung en Ihres Computer-Dateisyst ems, Datenbank-Dumps oder ähnlicher Sicherungsinhalte ausgelegt. Dies könnte Ihr System verlangsamen und den Zeitaufwand erhöhen, der zum Klonen und für die Push-Übertragung eines Repositor ys erforderlich ist.	Weitere Informationen zur Sicherung in der AWS-Cloud finden Sie unter Sicherung und Wiederherstellung.

Anwendungsfall	Beschreibung	Weitere zu berücksichtigende Services

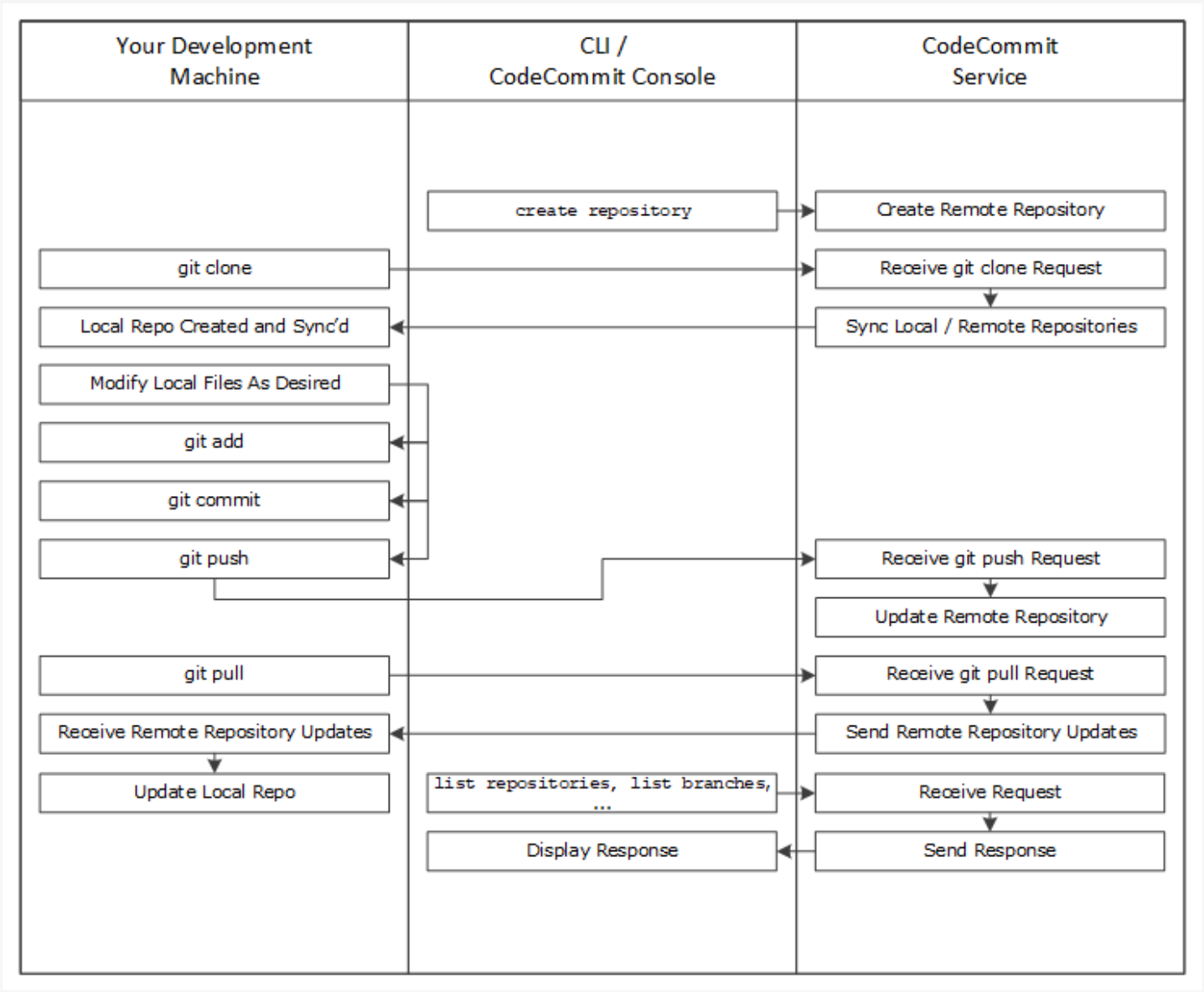
Große Anzahl von Verzweigungen oder Verweisen	Wenn ein Git-Client Repository-Daten per Push and Pull hochlädt oder abruft, muss der Remote-Server alle Verzweigungen und Verweise wie z. B. Tags senden, auch wenn Sie nur an einer Verzweigung interessiert sind. Wenn Tausende von Verzweigungen und Verweise vorhanden sind, kann die Verarbeitung und das Senden (Paketaushandlung) einige Zeit in Anspruch nehmen und eine scheinbar langsame Repository-Reaktion zur Folge haben. Je mehr Verzweigungen und Tags vorhanden sind, desto länger kann dieser Vorgang dauern. Es wird zur Verwendung von CodeCommit geraten. Löschen Sie jedoch alle Verzweigungen und Tags, die nicht mehr benötigt werden.	Mit den folgenden Befehle können Sie die Anzahl der Verweise in einem CodeCommit-Repository analysieren, um zu bestimmen, welche u. U. nicht mehr benötigt werden:
		• Linux, macOS oder Unix oder Bash-Emulator unter Windows:
		`git ls-remote | wc -l`
		• Powershell:
		`git ls-remote | Measure-Object -line`

Wie funktioniert CodeCommit ?

CodeCommitist Benutzern von Git-basierten Repositorys vertraut, aber selbst Unbekannte sollten den Übergang zu Git-basierten Repositorys alsCodeCommit relativ einfach empfinden. CodeCommitbietet eine Konsole für die einfache Erstellung von Repositorys und die Auflistung vorhandener Repositorys und Branches. Mit einigen einfachen Schritten können Benutzer Informationen über ein Repository erhalten, dieses auf ihrem Computer klonen, ein lokales Repository für die Änderungen erstellen und

diese per Push an das CodeCommit-Repository übertragen. Benutzer können die Befehlszeile des lokalen Computers oder einen GUI-basierten Editor nutzen.

In der folgenden Abbildung wird veranschaulicht, wie Sie mithilfe des Entwicklungscomputers, mit der AWS CLI oder der CodeCommit-Konsole oder dem CodeCommit-Service Repositorys erstellen und verwalten:



1. Verwenden Sie dieAWS CLI oder dieCodeCommit Konsole, um einCodeCommit Repository zu erstellen.

2. Führen Sie auf Ihrem Entwicklungscomputer über Git den Befehl git clone aus und geben Sie den Namen des CodeCommit-Repositorys an. Dadurch wird ein lokales Repo erstellt, das eine Verbindung zumCodeCommit Repository herstellt.
3. Verwenden Sie das lokale Repo auf Ihrem Entwicklungscomputer, um Dateien zu ändern (hinzuzufügen, zu bearbeiten und zu löschen), und führen Sie es dann aus,git add um die geänderten Dateien lokal bereitzustellen. Führen Sie den Befehl aus,git commit um die Dateien lokal zu übertragen, und führen Sie ihn dann aus,git push um die Dateien an dasCodeCommit Repository zu senden.
4. Laden Sie Änderungen von anderen Benutzern herunter. Führen Sie das Programm aus,git pull um die Dateien imCodeCommit Repository mit Ihrem lokalen Repo zu synchronisieren. So wird sichergestellt, dass Sie mit den neuesten Versionen dieser Dateien arbeiten.

Sie können Ihre Repositorys mit der AWS CLI oder der CodeCommit-Konsole verfolgen und verwalten.

WieCodeCommit unterscheidet sich die Dateiversionierung in Amazon S3?

CodeCommit ist für die Softwareentwicklung im Team optimiert. Es verwaltet Batches von Änderungen in mehreren Dateien, die parallel zu den von anderen Entwicklern vorgenommenen Änderungen erfolgen können. Die Amazon S3 S3-Versionierung unterstützt die Wiederherstellung früherer Dateiversionen, konzentriert sich jedoch nicht auf kollaborative Dateiverfolgungsfunktionen, die Softwareentwicklungsteams benötigen.

Erste Schritte mit CodeCommit

So sehen die ersten Schritte mit CodeCommit aus:

1. Führen Sie die Schritte unter Einrichtung aus, um die Entwicklungscomputer vorzubereiten.
2. Führen Sie die Schritte in einem oder mehreren Tutorials unter Erste Schritte aus.
3. Erstellen Sie Versionskontrollprojekte in CodeCommit oder migrieren Sie diese in CodeCommit.

Wo kann ich mehr über Git erfahren?

Sie sollten sich mit der Verwendung von Git vertraut machen, sofern noch nicht geschehen. Nachfolgend finden Sie einige hilfreiche Ressourcen:

- Pro Git ist eine Online-Version des Buchs Pro Git. Verfasst von Scott Chacon. Veröffentlicht von Apress.
- Git Immersion, einetry-it-yourself geführte Tour, die Sie durch die Grundlagen der Verwendung von Git führt. Veröffentlicht von Neo Innovation, Inc.
- Git Reference ist ein Online-Schnelleinstieg, der auch als ausführlicheres Git-Tutorial genutzt werden kann. Herausgegeben vomGitHub Team.
- Git Cheat Sheet bietet die grundlegende Git-Befehlssyntax. Herausgegeben vomGitHub Team.
- Git Pocket Guide. Verfasst von Richard E. Silverman. Veröffentlicht von O'Reilly Media, Inc.

Einrichten für AWS CodeCommit

Sie können sich bei derAWS Management Consoleundhochladen, hinzufügen oder bearbeiten Sie eine Dateizu einem Repository direkt von derAWS CodeCommitconsole. Dies ist eine schnelle Möglichkeit, um eine Änderung vorzunehmen. Wenn Sie jedoch mit mehreren Dateien, Branch-übergreifenden Dateien usw. arbeiten möchten, sollten Sie Ihren lokalen Computer so einrichten, dass er Repositorys unterstützt. Die einfachste Möglichkeit der Einrichtung von CodeCommit ist es, Git-HTTPS-Anmeldeinformationen für zu konfigurierenAWS CodeCommitaus. Diese HTTPS-Authentifizierungsmethode:

- Nutzt einen statischen Benutzernamen und ein statisches Passwort.
- Funktioniert mit allen von CodeCommit unterstützten Betriebssystemen.
- Ist auch mit Integrated Development Environments (IDEs, integrierte Entwicklungsumgebungen) und anderen Entwicklungstools kompatibel, die Git-Anmeldeinformationen unterstützen.

Sie können andere Methoden verwenden, wenn Sie dies wünschen, oder aus betrieblichen Gründen keine Git-Anmeldeinformationen verwenden können. Wenn Sie beispielsweise mit Verbundzugriff, temporären Anmeldeinformationen oder einem Webidentitätsanbieter auf CodeCommit-Repositorys zugreifen, können Sie keine Git-Anmeldeinformationen verwenden. Es wird empfohlen, den lokalen Computer mit dem `git-remote-codecommit`-Befehl einzurichten. Lesen Sie diese anderen Optionen aufmerksam durch, um zu entscheiden, welche alternative Methode am besten für Sie geeignet ist.

- Einrichten mit Git-Anmeldeinformationen
- Einrichten mit anderen Methoden
- Kompatibilität für CodeCommit, Git und anderen Komponenten

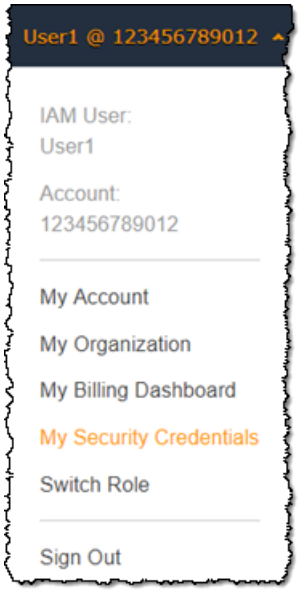
Weitere Informationen zur Verwendung von CodeCommit und Amazon Virtual Private Cloud finden Sie unterVerwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelleaus.

Ihre Anmeldeinformationen einsehen und verwalten

Sie können CodeCommit-Anmeldeinformationen über dasAWSkonsolen durchMeine Sicherheitsanmeldeinformationenaus.

> **Note**
>
> Diese Option steht Benutzern nicht zur Verfügung, die Verbundzugriff, temporäre Anmeldeinformationen oder einen Webidentitätsanbieter verwenden.

1. Melden Sie sich bei der AWS Management Console an, und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie auf der Navigationsleiste rechts oben Ihren Benutzernamen und dann My Security Credentials (Meine Sicherheitsanmeldeinformationen).



3. Wählen Sie das SymbolAWS CodeCommitReferenzenTabulator.

Einrichten mit Git-Anmeldeinformationen

Mit HTTPS-Verbindungen und Git-Anmeldeinformationen erzeugen Sie einen statischen Benutzernamen und ein statisches Passwort in IAM. Sie können dieselben Anmeldeinformationen dann auch mit Git und jedem Drittanbieter-Tool verwenden, das die Authentifizierung mit Git-Benutzername und -Passwort unterstützt. Diese Methode wird von den meisten IDEs und Entwicklungstools unterstützt. Es handelt sich dabei um die einfachste und einfachste Verbindungsmethode zur Verwendung mit CodeCommit.

- Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden: Befolgen Sie diese Anweisungen, um Verbindungen zwischen dem lokalen Computer und CodeCommit-Repositorys mithilfe von Git-Anmeldeinformationen einzurichten.
- Für Verbindungen von Entwicklungstools: Befolgen Sie diese Richtlinien, um Verbindungen zwischen der IDE oder anderen Entwicklungstools und CodeCommit-Repositorys mithilfe von Git-Anmeldeinformationen einzurichten. Zu den IDEs, die Git-Anmeldeinformationen unterstützen, zählen unter anderem Visual Studio, Eclipse, Xcode und IntelliJ.

Einrichten mit anderen Methoden

Sie können statt HTTPS das SSH-Protokoll verwenden, um eine Verbindung mit dem -CodeCommit-Repository herzustellen. Bei SSH-Verbindungen erstellen Sie öffentliche und private Schlüsseldateien auf Ihrem lokalen Rechner, die Git und CodeCommit für die SSH-Authentifizierung verwenden. Sie verknüpfen den öffentlichen Schlüssel mit dem -IAM-Benutzer. Den privaten Schlüssel speichern Sie auf dem lokalen Computer. Da SSH die manuelle Erstellung und Verwaltung von öffentlichen und privaten Schlüsseldateien erfordert, kann es einfacher sein, Git-Anmeldeinformationen mit CodeCommit zu verwenden.

Anders als bei Git-Anmeldeinformationen ist die Einrichtung von SSH-Verbindungen je nach Betriebssystem auf dem lokalen Computer unterschiedlich.

- Für SSH-Benutzer, die dieAWS CLI: Befolgen Sie diese verkürzten Anweisungen, wenn Sie bereits über ein Schlüsselpaar aus öffentlichem und privatem Schlüssel verfügen und mit SSH-Verbindungen auf dem lokalen Computer vertraut sind.
- Für SSH-Verbindungen unter Linux, macOS oder Unix: Befolgen Sie diese Anweisungen für eine schrittweise Anleitung zur Erstellung eines Schlüsselpaars aus öffentlichem und privatem Schlüssel und zur Einrichtung von Verbindungen unter Linux-, macOS- oder Unix -Betriebssystemen.
- Für SSH-Verbindungen unter Windows: Befolgen Sie diese Anweisungen für eine schrittweise Anleitung zur Erstellung eines Schlüsselpaars aus öffentlichem und privatem Schlüssel und zur Einrichtung von Verbindungen in Windows-Betriebssystemen.

Wenn Sie sich mit CodeCommit verbinden undAWSMit Verbundzugriff, einem Identitätsanbieter oder temporären Anmeldeinformationen oder wenn Sie keine IAM-Benutzer oder Git-Anmeldeinformationen für IAM-Benutzer konfigurieren möchten, können Sie Verbindungen mit CodeCommit-Repositorys auf zwei Arten einrichten:

- Installieren und verwenden Sie git-remote-codecommit (empfohlen).
- Installieren und verwenden Sie das Hilfsprogramm für Anmeldeinformationen, der in der AWS CLI enthalten ist.

Beide Methoden unterstützen den Zugriff auf CodeCommit-Repositorys, ohne dass ein IAM-Benutzer erforderlich ist. Das bedeutet, dass Sie mit Verbundzugriff und temporären Anmeldeinformationen eine Verbindung mit -Repositorys herstellen können. Das Dienstprogramm git-remote-codecommit ist der empfohlene Ansatz. Es erweitert Git und ist mit einer Vielzahl von Git-Versionen und Hilfsprogrammen für Anmeldeinformationen kompatibel. Allerdings unterstützen nicht alle IDEs das von `git-remote-codecommit` verwendete Klon-URL-Format. Möglicherweise müssen Sie Repositorys manuell auf Ihren lokalen Computer klonen, bevor Sie mit ihnen in der IDE arbeiten können.

- Folgen Sie den Anweisungen in Setup-Schritte für HTTPS-Verbindungen mit AWS CodeCommit Repositorys mit git-remote-codecommit installieren und einrichten git-remote-codecommit unter Windows, Linux, macOS oder Unix.

Der Hilfsprogramm für Anmeldeinformationen, der in der AWS CLI ermöglicht Git die Verwendung von HTTPS und einer kryptografisch signierten Version Ihrer IAM-Benutzeranmeldeinformationen oder Amazon EC2 EC2-Instancerolle, wenn Git sich mit authentifizieren muss AWSum mit CodeCommit-Repositorys zu interagieren. Einige Betriebssysteme und Git-Versionen verfügen über eigene Hilfsprogramme für Anmeldeinformationen, die mit dem in der AWS CLI enthaltenen Hilfsprogramm für Anmeldeinformationen in Konflikt stehen. Sie können bei CodeCommit zu Konnektivitätsproblemen führen.

- Für HTTPS-Verbindungen unter Linux, MacOS oder Unix mit dem AWS CLI Credential-Helper: Befolgen Sie diese Anweisungen für eine schrittweise Anleitung zur Installation und Einrichtung des Hilfsprogramms für Anmeldeinformationen auf Linux-, macOS- oder Unix-Systemen.
- Für HTTPS-Verbindungen unter Windows mit dem AWS CLI Credential-Helper: Befolgen Sie diese Anweisungen für eine schrittweise Anleitung zur Installation und Einrichtung des Hilfsprogramms für Anmeldeinformationen auf Windows-Systemen.

Wenn Sie eine Verbindung zu einem CodeCommit-Repository herstellen, das in einem anderen Amazon Web Services Services-Konto gehostet wird, können Sie mithilfe von Rollen, Richtlinien und dem Hilfsprogramm der für Anmeldeinformationen den Zugriff konfigurieren und Verbindungen einrichten AWS CLI aus.

- Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen: Befolgen Sie diese Anweisungen für eine schrittweise Anleitung zur Konfiguration eines kontoübergreifenden Zugriffs in einem anderen Amazon Web Services Services-Konto für Benutzer einer IAM-Gruppe in einem anderen Amazon Web Services Services-Konto.

Kompatibilität für CodeCommit, Git und anderen Komponenten

Wenn Sie mit CodeCommit arbeiten, verwenden Sie Git. Sie können aber auch andere Programme verwenden. Die folgende Tabelle enthält die neuesten Hinweise zur Versionskompatibilität. Als bewährte Methode empfehlen wir, die neuesten Versionen von Git und anderer Software zu verwenden.

Informationen zur Versionskompatibilität fürAWS CodeCommit

Komponente	Version
Git	CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt das Konfigurieren des Zweignamens für erste Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.
Curl	CodeCommit benötigt curl 7.33 und höher. Es gibt jedoch ein bekanntes Problem mit HTTPS und Curl-Update 7.41.0. Weitere Informationen finden Sie unter Fehlerbehebung .
Python (nur git-remote-codecommit)	git-remote-codecommit benötigt Version 3 und höher.
Pip (nur git-remote-codecommit)	git-remote-codecommit benötigt Version 9.0.3 und höher.
AWS CLI (nur git-remote-codecommit)	Wir empfehlen eine aktuelle Version vonAWS CLIVersion 2 für alle CodeCommit-Benutzer.git-remote-codecommiterfordertAWS CLIVersion 2 zur UnterstützungAWSSSO und Verbindungen,

Komponente	Version
	die temporäre Anmeldeinformationen erfordern, wie z. B. Verbundbenutzer.

Einrichtung für HTTPS-Benutzer mit Git-Anmeldeinformationen

Die einfachste Methode, Verbindungen zu AWS CodeCommit Repositorys einzurichten, besteht darin, Git-Anmeldeinformationen für CodeCommit in der IAM-Konsole zu konfigurieren und diese Anmeldeinformationen dann für HTTPS-Verbindungen zu verwenden. Sie können dieselben Anmeldeinformationen auch mit jedem Drittanbieter-Tool oder jeder integrierten Entwicklungsumgebung (IDE) verwenden, die die HTTPS-Authentifizierung mit einem statischen Benutzernamen und Passwort unterstützt. Beispiele finden Sie unter Für Verbindungen von Entwicklungstools.

> **Note**
>
> Wenn Sie Ihren lokalen Computer zuvor so konfiguriert haben, dass er den Credential Helper für verwendet CodeCommit, müssen Sie Ihre .gitconfig-Datei bearbeiten, um die Credential Helper-Informationen aus der Datei zu entfernen, bevor Sie Git-Anmeldeinformationen verwenden können. Wenn auf Ihrem lokalen Computer macOS ausgeführt wird, müssen Sie möglicherweise die zwischengespeicherten Anmeldeinformationen aus Keychain Access löschen.

Schritt 1: Erstkonfiguration für CodeCommit

Gehen Sie wie folgt vor, um ein Amazon Web Services Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und den Zugriff auf zu CodeCommit konfigurieren.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu http://aws.amazon.com gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere

Informationen finden Sie unter Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen.

> **Note**
>
> CodeCommit erfordert AWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter AWS KMS und Verschlüsselung.

3. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie AWSCodeCommitPowerUseraus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit.

   Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

   Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter Teilen Sie ein Repository und. Authentifizierung und Zugriffskontrolle für AWS CodeCommit

Wenn Sie AWS CLI Befehle mit verwenden möchten CodeCommit, installieren Sie den AWS CLI. Wir empfehlen Ihnen, ein Profil für die Verwendung von AWS CLI with zu erstellen CodeCommit. Weitere Informationen finden Sie unter Befehlszeilenreferenz Benannte Profile verwenden.

Schritt 2: Installieren von Git

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen

1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie Git Downloads.

> **Note**
>
> Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn Sie Probleme mit einer bestimmten Version von Git und haben CodeCommit, lesen Sie die Informationen unterFehlerbehebung.

Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit

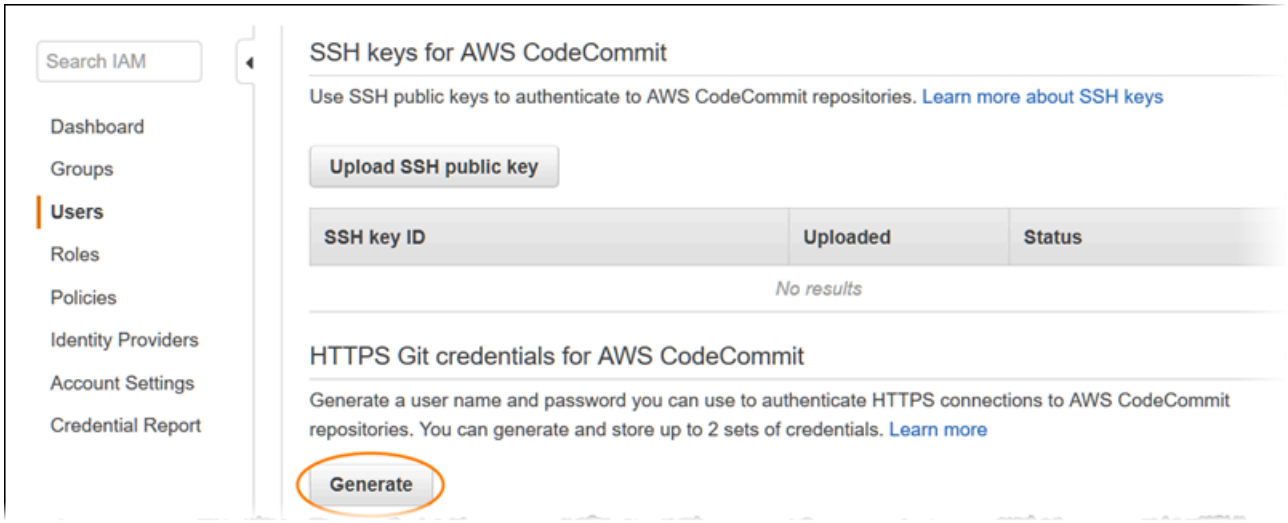
Nachdem Sie Git installiert haben, erstellen Sie Git-Anmeldeinformationen für Ihren IAM-Benutzer in IAM.

So richten Sie HTTPS-Git-Anmeldeinformationen ein für CodeCommit

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

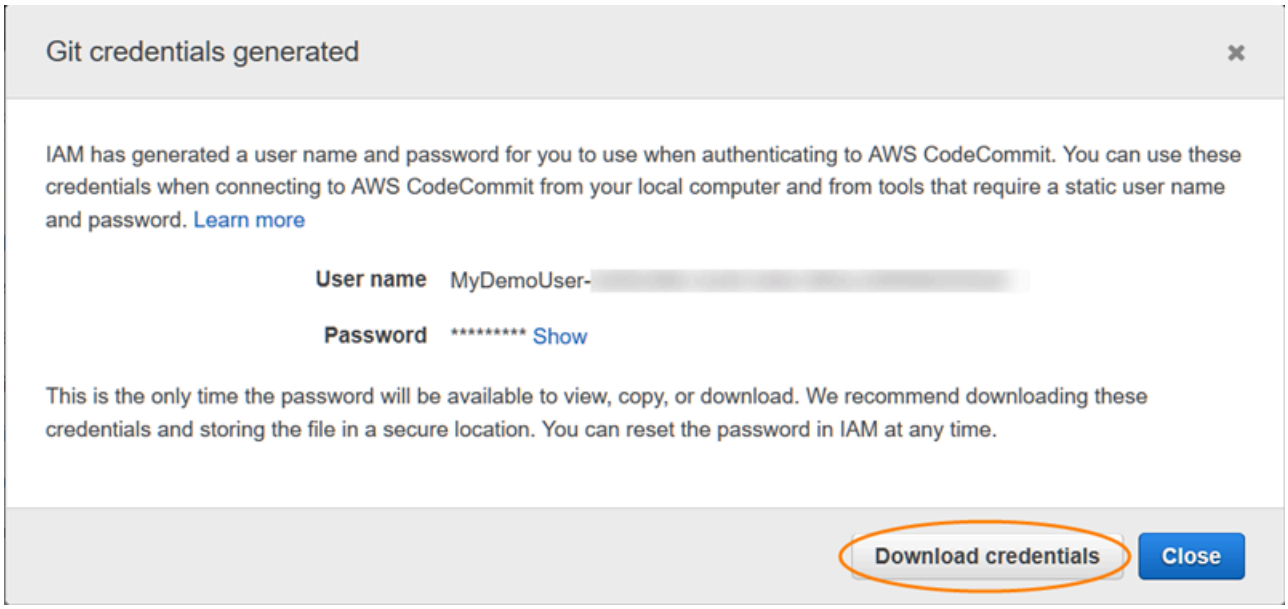
   Stellen Sie sicher, dass Sie sich als der IAM-Benutzer anmelden, der die Git-Anmeldeinformationen für Verbindungen zu CodeCommit erstellt und verwendet.
2. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer aus und wählen Sie aus der Benutzerliste Ihren IAM-Benutzer aus.

   > **Note**
   >
   > Sie können Ihre CodeCommit Anmeldeinformationen direkt unter Meine Sicherheitsanmeldedaten einsehen und verwalten. Weitere Informationen finden Sie unter Ihre Anmeldeinformationen einsehen und verwalten.
3. Wählen Sie auf der Seite mit den Benutzerdetails die Registerkarte Sicherheitsanmeldedaten und wählen Sie unter HTTPS-Git-Anmeldeinformationen für die AWS CodeCommit Option Generieren aus.



> **Note**
>
> Sie können nicht Ihren eigenen Benutzernamen oder das Passwort für Git-Anmeldeinformationen wählen. Weitere Informationen finden Sie unter Verwenden von Git-Anmeldeinformationen und HTTPS mit CodeCommit.

4. Kopieren Sie den Benutzernamen und das Passwort, die IAM für Sie generiert hat, indem Sie diese Informationen entweder anzeigen, kopieren und dann in eine sichere Datei auf Ihrem lokalen Computer einfügen oder indem Sie Anmeldeinformationen herunterladen wählen, um diese Informationen als CSV-Datei herunterzuladen. Sie benötigen diese Informationen, um eine Verbindung herzustellen. CodeCommit



Wählen Sie, nachdem Sie Ihre Anmeldeinformationen gespeichert haben, die Option Close aus.

> **Important**
>
> Dies ist Ihre einzige Möglichkeit, den Benutzernamen und das Passwort zu speichern. Wenn Sie sie nicht speichern, können Sie den Benutzernamen von der IAM-Konsole kopieren, aber Sie können das Passwort nicht nachschlagen. Sie müssen das Passwort zurücksetzen und es dann speichern.

Schritt 4: Connect zur CodeCommit Konsole her und klonen Sie das Repository

Wenn Ihnen bereits ein Administrator den Namen und die Verbindungsdetails für das CodeCommit Repository gesendet hat, können Sie diesen Schritt überspringen und das Repository direkt klonen.

Um eine Verbindung zu einem CodeCommit Repository herzustellen

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositorys sind spezifisch für ein. AWS-Region Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.

   - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang von enthalten ist. AWS CLI
   - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
   - Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

> **Note**
>
> Wenn Sie statt einer Liste von Repositorys eine Willkommensseite sehen, sind Ihrem AWS Konto in dem Land, in dem Sie angemeldet sind, keine Repositorys zugeordnet. AWS-Region Informationen zur Erstellung eines Repositorys finden Sie unter the section called “Erstellen eines -Repositorys”, oder befolgen Sie die Schritte im Tutorial Erste Schritte mit Git und CodeCommit.

4. Öffnen Sie ein Terminal- oder Befehlszeilenfenster oder eine Git-Shell. Führen Sie den git clone-Befehl mit der HTTPS-Klon-URL aus, die Sie kopiert haben, um das Repository zu klonen. Um beispielsweise ein Repository mit dem Namen *MyDemoRepo*eines lokalen Repositorys mit dem Namen *my-demo-repo*der Region USA Ost (Ohio) zu klonen:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-
demo-repo
```

Wenn Sie zum ersten Mal die Verbindung herstellen, werden Sie aufgefordert, den Benutzernamen und das Passwort für das Repository einzugeben. Abhängig von der Konfiguration Ihres lokalen Computers stammt diese Aufforderung entweder von einem Anmeldeinformationsverwaltungssystem für das Betriebssystem, einem Credential Manager-Hilfsprogramm für Ihre Version von Git (z. B. dem in Git für Windows enthaltenen Git Credential Manager), Ihrer IDE oder von Git selbst. Geben Sie den Benutzernamen und das Passwort ein, die für Git-Anmeldeinformationen in IAM generiert wurden (die, die Sie in erstellt habenSchritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit). Je nach Betriebssystem und anderer Software werden diese Daten möglicherweise für Sie in einem Speicher für Anmeldeinformationen oder einem Dienstprogramm zur Verwaltung von Anmeldeinformationen gespeichert. In diesem Fall sollten Sie nicht erneut dazu aufgefordert werden, es sei denn, Sie ändern das Passwort, deaktivieren die Git-Anmeldeinformationen oder löschen die Git-Anmeldeinformationen in IAM.

Wenn auf dem lokalen Computer kein Speicher für Anmeldeinformationen und kein Dienstprogramm zur Verwaltung von Anmeldeinformationen konfiguriert ist, können Sie einen Speicher bzw. ein Dienstprogramm installieren. Weitere Informationen über Git und darüber, wie es Anmeldeinformationen verwaltet, finden Sie in der Git-Dokumentation unter Credential Storage.

Weitere Informationen finden Sie unter Connect zum CodeCommit Repository her, indem Sie das Repository klonen und Erstellen Sie einen Commit.

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Schritten unter, um mit der Nutzung Erste Schritte mit CodeCommit zu beginnen. CodeCommit

Informationen zum Erstellen und Pushen Ihres ersten Commit finden Sie unterErstellen Sie einen Commit in AWS CodeCommit. Wenn Sie mit Git noch nicht vertraut sind, finden Sie unter Wo kann ich mehr über Git erfahren? und Erste Schritte mit Git und AWS CodeCommit genauere Informationen.

Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit

Wenn Sie mit einem Root-Konto, einem Verbundzugriff oder temporären Anmeldeinformationen eine Verbindung mit CodeCommit herstellen möchten, sollten Sie Zugriff mittels git-remote-codecommit einrichten. Dieses Dienstprogramm bietet eine einfache Methode zum Push- und Pull von Code aus CodeCommit-Repositorys durch die Erweiterung von Git. Dies ist die empfohlene Methode zur Unterstützung von Verbindungen, die mit Verbundzugriff, Identitätsanbietern und temporären Anmeldeinformationen hergestellt werden. Um einer föderierten Identität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wenn eine Verbundidentität authentifiziert wird, wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center-Benutzerhandbuch.

Sie können auch Folgendes verwendengit-remote-codecommitmit einem IAM-Benutzer. Im Gegensatz zu anderen HTTPS-Verbindungsmethoden erfordert git-remote-codecommit das Einrichten von Git-Anmeldeinformationen für den Benutzer nicht.

> **Note**
>
> Einige IDEs unterstützen das von `git-remote-codecommit` verwendete Klon-URL-Format nicht. Möglicherweise müssen Sie Repositorys manuell auf Ihren lokalen Computer klonen, bevor Sie mit ihnen in der bevorzugten IDE arbeiten können. Weitere Informationen finden Sie unter Fehlerbehebung bei git-remote-codecommit und AWS CodeCommit.

Diese Verfahren wurden unter der Annahme geschrieben, dass Sie ein Amazon Web Services-Konto haben und mindestens ein Repository erstellt haben inCodeCommit, und verwenden Sie einen IAM-Benutzer mit einer verwalteten Richtlinie, wenn Sie eine Verbindung zu herstellenCodeCommitRepositorien. Informationen zum Konfigurieren des Zugriffs für verbundener Benutzer und andere rotierende Anmeldeinformationstypen finden Sie unter Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen.

Themen

- Schritt 0: Installieren Sie die Voraussetzungen fürgit-remote-codecommit
- Schritt 1: Erstkonfiguration fürCodeCommit
- Schritt 2: Installierengit-remote-codecommit
- Schritt 3: Stellen Sie eine Verbindung zum herCodeCommitkonsole und klone das Repository
- Nächste Schritte

Schritt 0: Installieren Sie die Voraussetzungen fürgit-remote-codecommit

Bevor Sie git-remote-codecommit verwenden können, müssen Sie einige Voraussetzungen auf Ihrem lokalen Computer installieren. Dazu zählen:

- Python (Version 3 oder höher) und dessen Paketmanager „pip“, wenn sie nicht bereits installiert sind. Um die neueste Version von Python herunterzuladen und zu installieren, besuchen Sie die Python-Website.
- Git

> **Note**
>
> Wenn Sie Python unter Windows installieren, stellen Sie sicher, dass Sie die Option auswählen, mit der sich Python zum Pfad hinzufügen lässt.

git-remote-codecommit benötigt pip Version 9.0.3 oder höher. Um Ihre pip-Version zu überprüfen, öffnen Sie ein Terminal oder eine Befehlszeile und führen Sie den folgenden Befehl aus:

```
pip --version
```

Sie können die folgenden beiden Befehle ausführen, um Ihre pip-Version auf die neueste Version zu aktualisieren:

```
curl -O https://bootstrap.pypa.io/get-pip.py
python3 get-pip.py --user
```

Um mit Dateien, Commits und anderen Informationen zu arbeiten inCodeCommitRepositorys müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt die Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wieGit lädt herunter.

> **Note**
>
> Git ist eine sich weiterentwickelnde, regelmäßig aktualisierte Plattform. Es kommt vor, dass sich die Änderung eines Features darauf auswirkt, wie Git mit CodeCommit zusammen funktioniert. Wenn Sie Probleme mit einer bestimmten Version von Git und CodeCommit haben, lesen Sie die Informationen in Fehlerbehebung.

Schritt 1: Erstkonfiguration fürCodeCommit

Gehen Sie wie folgt vor, um einen IAM-Benutzer zu erstellen, ihn mit den entsprechenden Richtlinien zu konfigurieren, einen Zugriffsschlüssel und einen geheimen Schlüssel zu erhalten und den zu installieren und zu konfigurierenAWS CLI.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurierenCodeCommit

1. Erstellen Sie ein Amazon Web Services-Konto, indem Sie zuhttp://aws.amazon.comund wählenMelde dich an.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unterEinen IAM-Benutzer in Ihrem Amazon Web Services-Konto erstellen.

   > **Note**
   >
   > CodeCommit erfordert AWS Key Management Service Wenn Sie einen bestehenden IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die das ausdrücklich verbietenAWS KMSAktionen erforderlich vonCodeCommit. Weitere Informationen finden Sie unter AWS KMS und Verschlüsselung.

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
4. Wählen Sie in der IAM-Konsole im NavigationsbereichBenutzer, und wählen Sie dann den IAM-Benutzer aus, für den Sie konfigurieren möchtenCodeCommitZugriff.
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie in der Richtlinienliste den Eintrag AWSCodeCommitPowerUser oder eine andere verwaltete Richtlinie für den CodeCommit-Zugriff aus. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit.

   Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen SieWeiter: Überprüfenum die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

   Weitere Informationen über die verwalteten Richtlinien von CodeCommit und die Freigabe von Repositorys für andere Gruppen und Benutzer finden Sie unter Teilen Sie ein Repository und Authentifizierung und Zugriffskontrolle für AWS CodeCommit.

So installieren und konfigurieren Sie die AWS CLI

1. Laden Sie die AWS CLI auf den lokalen Computer herunter und installieren Sie. Dies ist eine Voraussetzung für das Arbeiten mit CodeCommit über die Befehlszeile. Wir empfehlen die InstallationAWS CLIVersion 2. Es ist die neueste Hauptversion vonAWS CLIund unterstützt die neuesten Funktionen. Es ist die einzige Version vonAWS CLIdas unterstützt die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mitgit-remote-codecommit.

   Weitere Informationen finden Sie unter Erste Schritte mit der AWS-Befehlszeilenschnittstelle.

   > **Note**
   >
   > CodeCommitfunktioniert nur mitAWS CLIVersionen 1.7.38 und höher. Als bewährte Methode installieren oder aktualisieren Sie die AWS CLI auf die neueste verfügbare Version. Führen Sie den Befehl aws --version aus, um zu überprüfen, welche Version der AWS CLI installiert ist.
   > Informationen zum Upgraden von einer älteren auf die aktuelle Version der AWS CLI finden Sie unter Installieren der AWS Command Line Interface.

2. Führen Sie diesen Befehl aus, um zu überprüfen, obCodeCommitBefehle fürAWS CLIsind installiert.

   ```
   aws codecommit help
   ```

   Dieser Befehl gibt eine Liste vonCodeCommitBefehle.

3. Konfigurieren Sie dieAWS CLImit einem Profil unter Verwendung desconfigureBefehl, wie folgt:.

   ```
   aws configure
   ```

   Wenn Sie dazu aufgefordert werden, geben SieAWSZugriffsschlüssel undAWSgeheimer Zugriffsschlüssel des IAM-Benutzers, mit dem verwendet werden sollCodeCommit. Stellen Sie außerdem sicher, dass Sie Folgendes angebenAWS-Regionwo das Repository existiert, z. B.`us-east-2`. Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

   ```
   AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press
    Enter
   ```

```
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and
 then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then
 press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen für die AWS CLI finden Sie unter:

- Benannte Profile
- Verwenden einer IAM-Rolle in derAWS CLI
- Befehl „Set“
- Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen herzustellenAWS-Region, müssen Sie das neu konfigurierenAWS CLImit dem Standardnamen der Region. Zu den Standardregionsnamen für CodeCommit zählen:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1

- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3
- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und AWS-Region finden Sie unter Regionen und Git-Verbindungsendpunkte. Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unterWie erhalte ich Anmeldeinformationen?undVerwaltung von Zugriffsschlüsseln für IAM-Benutzer. Weitere Informationen zur AWS CLI und zu Profilen finden Sie unter Benannte Profile.

Schritt 2: Installierengit-remote-codecommit

Führen Sie zur Installation von git-remote-codecommit die folgenden Schritte aus.

So installieren Sie git-remote-codecommit

1. Führen Sie am Terminal oder in der Befehlszeile den folgenden Befehl aus:

```
pip install git-remote-codecommit
```

> **Note**
>
> Je nach Betriebssystem und Konfiguration müssen Sie diesen Befehl möglicherweise mit erhöhten Rechten ausführen, z. B. mit sudo, oder Sie müssen den Parameter --

user verwenden, um in einem Verzeichnis zu installieren, für das keine besonderen Rechte erforderlich sind, wie z. B. Ihr aktuelles Benutzerkonto. Zum Beispiel auf einem Computer, auf dem Linux, macOS oder Unix ausgeführt wird:

```
sudo pip install git-remote-codecommit
```

Auf einem Computer mit Windows:

```
pip install --user git-remote-codecommit
```

2. Überwachen Sie den Installationsvorgang, bis Sie eine Erfolgsmeldung sehen.

Schritt 3: Stellen Sie eine Verbindung zum herCodeCommitkonsole und klone das Repository

Wenn Ihnen ein Administrator bereits die Klon-URL zur Verwendung mit git-remote-codecommit für das CodeCommit-Repository gesendet hat, können Sie die Verbindung zur Konsole überspringen und das Repository direkt klonen.

So stellen Sie eine Verbindung mit einem CodeCommit-Repository her

1. Öffne dasCodeCommitKonsole beihttps://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der RegionsauswahlAWS-Regionwo das Repository erstellt wurde. Repositorien sind spezifisch für einAWS-Region. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.

   - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang vonAWS CLI.
   - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.

- Kopieren Sie die SSH-URL, wenn Sie mit Ihrem IAM-Benutzer ein öffentliches/privates SSH-Schlüsselpaar verwenden.

> **Note**
>
> Wenn Sie eine sehenWillkommenSeite statt einer Liste von Repositorien, es gibt keine Repositorys, die mit Ihrem verknüpft sindAWSKonto imAWS-Regionwo Sie angemeldet sind. Informationen zur Erstellung eines Repositorys finden Sie unter the section called "Erstellen eines -Repositorys", oder befolgen Sie die Schritte im Tutorial Erste Schritte mit Git und CodeCommit.

4. Klonen Sie am Terminal oder an der Eingabeaufforderung das Repository mit dem git clone-Befehl. Verwenden Sie das HTTPSgit-remote-codecommitDie URL, die Sie kopiert haben, und der Name desAWS CLIProfil, wenn Sie ein benanntes Profil erstellt haben. Wenn Sie kein Profil angeben, nimmt der Befehl das Standardprofil an. Dadurch wird das lokale Repository in einem Unterverzeichnis des Verzeichnisses erstellt, in dem Sie den Befehl ausgeführt haben. Im folgenden Beispiel wird ein Repository mit dem Namen *MyDemoRepo* in ein lokales Repository mit dem Namen *my-demo-repo* geklont:

```
git clone codecommit://MyDemoRepo my-demo-repo
```

Um dasselbe Repository mit einem Profil namens zu klonen*CodeCommitProfile*:

```
git clone codecommit://CodeCommitProfile@MyDemoRepo my-demo-repo
```

Um ein Repository in einem anderen zu klonenAWS-Regionals das in Ihrem Profil konfigurierte, fügen Sie das hinzuAWS-RegionName. Beispiele:

```
git clone codecommit::ap-northeast-1://MyDemoRepo my-demo-repo
```

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Schritten inErste Schritte mit CodeCommit um mit der Nutzung zu beginnenCodeCommit.

Wie du deinen ersten Commit erstellst und pushst, erfährst du unterErstellen Sie einen Commit in AWS CodeCommit. Wenn Sie mit Git noch nicht vertraut sind, finden Sie unter Wo kann ich mehr über Git erfahren? und Erste Schritte mit Git und AWS CodeCommit genauere Informationen.

Verbindungen von Entwicklungstools mithilfe von Git-Anmeldeinformationen einrichten

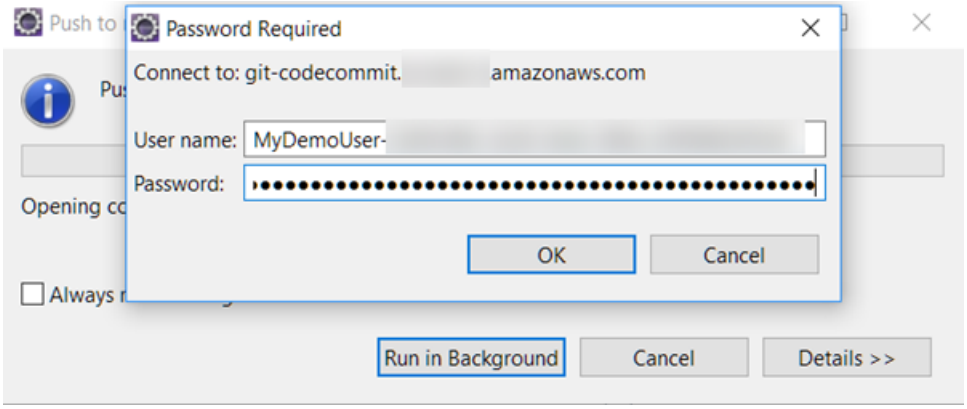
Nachdem Sie die Git-Anmeldeinformationen für AWS CodeCommit in der IAM-Konsole konfiguriert haben, können Sie diese Anmeldeinformationen mit jedem Entwicklungstool verwenden, das Git-Anmeldeinformationen unterstützt. Sie können beispielsweise den Zugriff auf Ihr CodeCommit Repository in Visual Studio AWS Cloud9, Eclipse, Xcode, IntelliJ oder einer beliebigen integrierten Entwicklungsumgebung (IDE) konfigurieren, die Git-Anmeldeinformationen integriert. Wenn Sie den Zugriff konfiguriert haben, können Sie den Code bearbeiten, die Änderungen per Commit übergeben und direkt per Push von der IDE oder einem anderen Entwicklungstool übertragen.

> **Note**
>
> Wenn Sie über Verbundzugriff, temporäre Anmeldeinformationen oder einen Web-Identitätsanbieter auf CodeCommit Repositorys zugreifen, können Sie keine Git-Anmeldeinformationen verwenden. Es wird empfohlen, den lokalen Computer mit dem `git-remote-codecommit`-Befehl einzurichten. Allerdings sind nicht alle IDEs vollständig mit Git-Remote-Helfern wie git-remote-codecommit kompatibel. Bei Problemen finden Sie weitere Informationen unter Fehlerbehebung bei git-remote-codecommit und AWS CodeCommit.

Themen

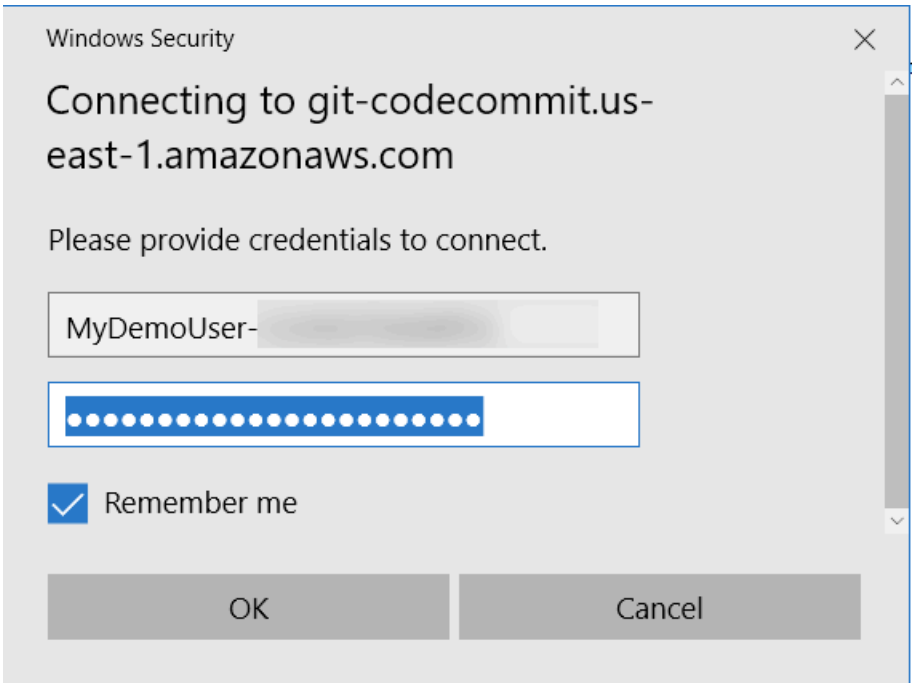
- Integration von AWS Cloud9 in AWS CodeCommit
- Integrieren Sie Visual Studio mitAWS CodeCommit
- Integration von Eclipse in AWS CodeCommit

Wenn Sie von Ihrer IDE oder Ihrem Entwicklungstool nach dem Benutzernamen und dem Passwort gefragt werden, die für die Verbindung mit dem CodeCommit Repository verwendet wurden, geben Sie die Git-Anmeldeinformationen für den Benutzernamen und das Passwort ein, die Sie in IAM erstellt haben. Wenn Sie beispielsweise zur Eingabe eines Benutzernamens und eines Passworts in Eclipse aufgefordert werden, geben Sie Ihre Git-Anmeldeinformationen wie folgt an:



Weitere Informationen zu AWS-Regionen und Endpunkten für finden Sie CodeCommit unterRegionen und Git-Verbindungsendpunkte.

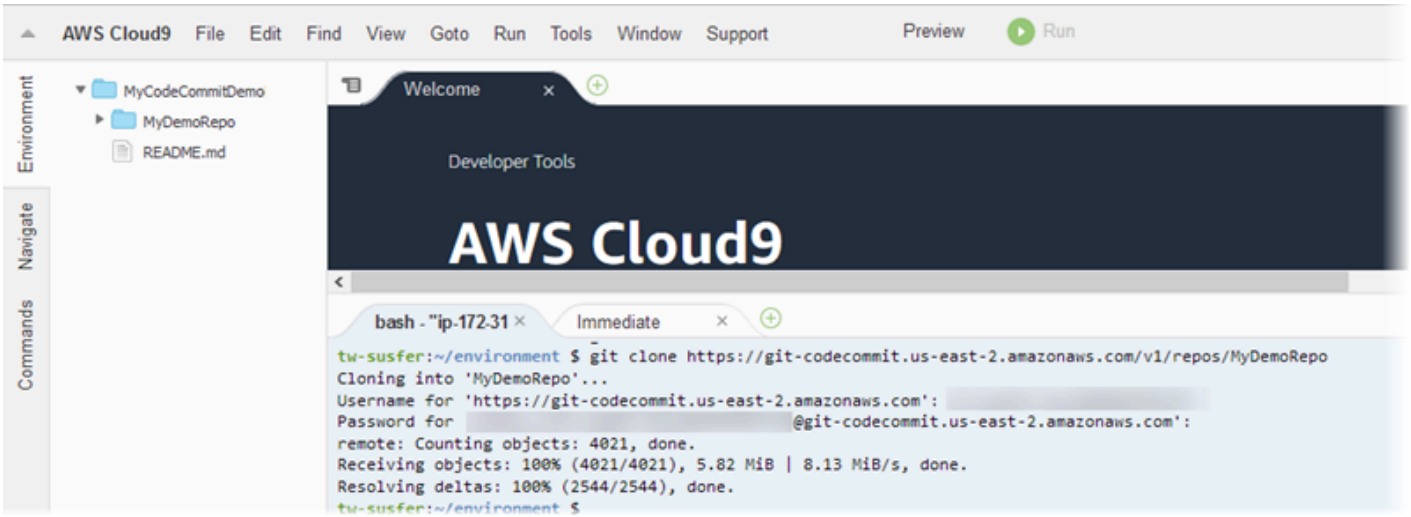
Möglicherweise werden Sie auch vom Betriebssystem aufgefordert, Ihren Benutzernamen und das Passwort zu speichern. Beispielsweise geben Sie in Windows Ihre Git-Anmeldeinformationen wie folgt ein:



Weitere Informationen zur Konfiguration der Git-Anmeldeinformationen für ein bestimmtes Softwareprogramm oder Entwicklungstool finden Sie in der Dokumentation zum jeweiligen Produkt.

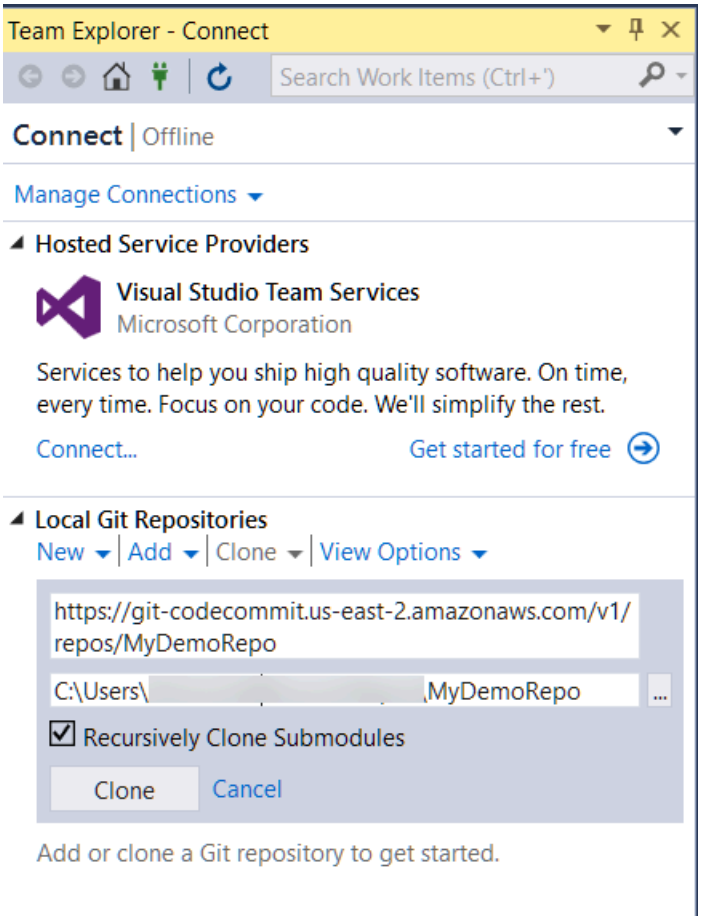
Die folgende Liste enthält nicht alle IDEs. Die Links dienen ausschließlich dazu, Ihnen zu helfen, mehr über diese Tools zu erfahren. AWS ist nicht verantwortlich für den Inhalt dieser Themen.

- AWS Cloud9



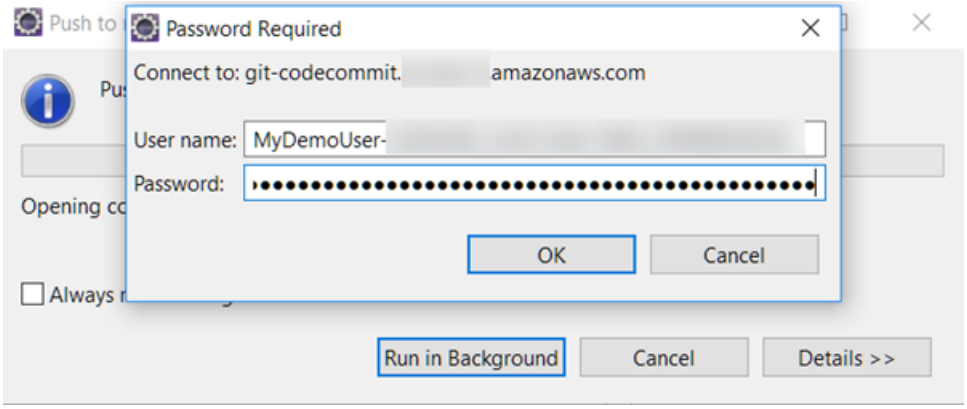
- Visual Studio

  Installieren Sie alternativ die AWS Toolkit for Visual Studio. Weitere Informationen finden Sie unter Integrieren Sie Visual Studio mitAWS CodeCommit.



- EGit mit Eclipse

Installieren Sie alternativ die AWS Toolkit for Eclipse. Weitere Informationen finden Sie unter Integration von Eclipse in AWS CodeCommit.



- XCode

Integration von AWS Cloud9 in AWS CodeCommit

Sie können verwendenAWS Cloud9So machen Sie Codeänderungen in einem CodeCommit -Repository.AWS Cloud9enthält eine Sammlung von Tools, mit denen Sie Code schreiben sowie Software erstellen, ausführen, testen, debuggen und veröffentlichen können. Sie können vorhandene Repositorys klonen, Repositorys erstellen, Commits durchführen und Codeänderungen per Push an ein Repository übertragen und mehr – und das alles direkt von Ihrer AWS Cloud9 EC2-Entwicklungsumgebung aus. DieAWS Cloud9Die EC2-Entwicklungsumgebung ist in der Regel mit derAWS CLI, einer Amazon EC2 EC2-Rolle und Git. In den meisten Fällen können Sie daher einige einfache Befehle ausführen und die Interaktion mit Ihrem Repository beginnen.

Um zu verwendenAWS Cloud9Bei CodeCommit benötigen Sie Folgendes:

- Eine AWS Cloud9 EC2-Entwicklungsumgebung, die in Amazon Linux ausgeführt wird
- Die in einem Webbrowser geöffnete AWS Cloud9-IDE
- Ein IAM-Benutzer mit einem der CodeCommit verwaltete Richtlinien und eine derAWS Cloud9verwaltete Richtlinien, die darauf angewendet wurden.

  Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit und Verstehen und Abrufen Ihrer Sicherheitsanmeldeinformationen.

> **Note**
>
> In diesem Thema wird das Einrichten der Integration mit beschrieben CodeCommit undAWS Cloud9mit allgemeinem Zugriff aus dem Internet. Sie können den Zugriff auf einrichten CodeCommit undAWS Cloud9in einer isolierten Umgebung, aber das erfordert zusätzliche Schritte. Weitere Informationen finden Sie unter:
>
> - Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle
> - Zugriff auf Amazon EC2 EC2-Instances ohne Eindringen mitAWS Systems Manager
> - Arbeiten mit gemeinsamen Umgebungen
> - Freigeben Ihrer VPC für andere Konten
> - Blog-Beitrag: Den Netzwerkzugriff auf IhreAWS Cloud9Umgebungen

Themen

- Schritt 1: Erstellen einesAWS Cloud9Entwicklungsumgebung
- Schritt 2: Konfigurieren derAWS CLIHilfsprogramm für Anmeldeinformationen auf IhremAWS Cloud9EC2-Entwicklungsumgebung
- Schritt 3: Clone a CodeCommit Repository in IhremAWS Cloud9EC2-Entwicklungsumgebung
- Nächste Schritte

Schritt 1: Erstellen einesAWS Cloud9Entwicklungsumgebung

AWS Cloud9hostet Ihre Entwicklungsumgebung auf einer Amazon EC2 EC2-Instance. Dies ist die einfachste Integration, da Sie dasAWSverwaltete temporäre Anmeldeinformationen für die Instanz zur Verbindung mit Ihrem CodeCommit -Repository. Wenn Sie stattdessen Ihren eigenen Server verwenden möchten, finden Sie weitere Informationen unterAWS Cloud9-Benutzerhandbuchaus.

Erstellen Sie eine AWS Cloud9-Umgebung wie folgt:

1. Anmeldung beiAWSAls IAM-Benutzer, den Sie konfiguriert haben und öffnenAWS Cloud9console.
2. Wählen Sie in der AWS Cloud9-Konsole Create environment (Umgebung erstellen) aus.
3. In :Schritt 1: Benennen der Umgebung, geben Sie einen Namen und eine optionale Beschreibung für die Umgebung ein und wählen Sie dannNächster Schrittaus.

4. In :Schritt 2: Konfigurieren der Einstellungenkonfigurieren Sie Ihre Umgebung wie folgt:

   - Wählen Sie unter Environment type die Option Create a new instance for environment (EC2) aus.
   - Klicken Sie unter Instance type auf den entsprechenden Instance-Typ für Ihre Entwicklungsumgebung. Wenn Sie sich nur mit dem Service vertraut machen möchten, wählen Sie die Standardoption t2.micro aus. Wenn Sie diese Umgebung für die Entwicklung verwenden möchten, wählen Sie einen größeren Instance-Typ aus.
   - Akzeptieren Sie die anderen Standardeinstellungen, es sei denn, Sie benötigen andere Einstellungen (z. B. wenn Ihre Organisation eine bestimmte VPC verwendet oder für Ihr Amazon Web Services Services-Konto keine VPCs konfiguriert sind).Nächster Schrittaus.

5. In :Schritt 3: Prüfen, überprüfen Sie Ihre Einstellungen. Wählen Sie Previous step aus, wenn Sie Änderungen vornehmen möchten. Wählen Sie andernfalls Create environment aus.

   Das Erstellen einer Umgebung und das erstmalige Herstellen einer Verbindung dauert einige Minuten. Wenn dieser Vorgang ungewöhnlich lange dauert, finden Sie weitere Informationen unter Fehlerbehebung im AWS Cloud9 Benutzerhandbuch.

6. Nachdem Sie eine Verbindung zu Ihrer Umgebung hergestellt haben, prüfen Sie, ob Git bereits installiert und eine unterstützte Version ist, indem Sie den Git-Befehl git --version im Terminal-Fenster ausführen.

   Wenn Git nicht installiert oder keine unterstützte Version ist, installieren Sie eine unterstützte Version. CodeCommit unterstützt die Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt das Konfigurieren des Zweignamens für erste Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git. Für die Installation von Git empfehlen wir Websites wieGit-Downloadsaus.

   > **Tip**
   >
   > Je nach Betriebssystem Ihrer Umgebung können Sie mit dem Befehl yum und der Option sudo Updates, einschließlich Git, installieren. Eine administrative Befehlssequenz ähnelt beispielsweise den folgenden drei Befehlen:
   >
   > ```
   > sudo yum -y update
   > sudo yum -y install git
   > git --version
   > ```

7. Konfigurieren Sie einen Benutzernamen und eine E-Mail-Adresse, die mit Ihren Git-Commits verknüpft werden, indem Sie den Befehl git config ausführen. Zum Beispiel:

```
git config --global user.name "Mary Major"
git config --global user.email mary.major@example.com
```

Schritt 2: Konfigurieren derAWS CLIHilfsprogramm für Anmeldeinformationen auf IhremAWS Cloud9EC2-Entwicklungsumgebung

Nachdem du eine erstellt hastAWS Cloud9-Umgebung können Sie dieAWS CLIHilfsprogramm für Anmeldeinformationen zum Verwalten der Anmeldeinformationen für Verbindungen mit Ihrem CodeCommit -Repository. DieAWS Cloud9Entwicklungsumgebung kommt mitAWSverwaltete, temporäre Anmeldeinformationen, die mit Ihrem IAM-Benutzer verknüpft sind. Diese Anmeldeinformationen verwenden Sie mit dem Hilfsprogramm für AWS CLI-Anmeldeinformationen.

1. Öffnen Sie das Terminal-Fenster und führen Sie den folgenden Befehl aus, um zu überprüfen, dass die AWS CLI installiert ist:

```
aws --version
```

   Im Erfolgsfall gibt dieser Befehl die derzeit installierte Version der AWS CLI zurück. Informationen zum Upgraden von einer älteren auf die aktuelle Version der AWS CLI finden Sie unter Installieren der AWS Command Line Interface.

2. Führen Sie am Terminal die folgenden Befehle zum Konfigurieren des Hilfsprogramms für AWS CLI-Anmeldeinformationen für HTTPS-Verbindungen aus:

```
git config --global credential.helper '!aws codecommit credential-helper $@'
git config --global credential.UseHttpPath true
```

> **Tip**
>
> Das Hilfsprogramm für Anmeldeinformationen verwendet die Amazon EC2 EC2-Instance-Rolle für Ihre Entwicklungsumgebung. Wenn Sie die Entwicklungsumgebung für die Verbindung mit Repositorys verwenden möchten, die nicht in CodeCommit gehostet werden, konfigurieren Sie entweder SSH-Verbindungen mit diesen

Repositorys oder konfigurieren Sie ein lokales `.gitconfig`-Datei, um ein alternatives Anmeldeinformationsverwaltungssystem zu verwenden, wenn Sie eine Verbindung zu diesen anderen Repositorys herstellen. Weitere Informationen finden Sie unter Git Tools - Credential Storage auf der Git-Website.

Schritt 3: Clone a CodeCommit Repository in IhremAWS Cloud9EC2-Entwicklungsumgebung

Nachdem Sie dasAWS CLIHilfsprogramm für Anmeldeinformationen können Sie Ihr CodeCommit-Repository klonen. Anschließend können Sie den Code bearbeiten.

1. Führen Sie am Terminal den Befehl git clone unter Angabe der HTTPS-Klon-URL des zu klonenden Repositorys aus. Wenn Sie beispielsweise ein -Repository mit dem Namen klonen möchten MyDemoRepo In der Region USA Ost (Ohio) würden Sie Folgendes eingeben:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
```

> **Tip**
>
> Sie finden die Klon-URL für Ihr -Repository im CodeCommit konsole durch AuswahlClone URLaus.

2. Wenn das Klonen abgeschlossen ist, erweitern Sie den Ordner für das Repository in der seitlichen Navigationsleiste und wählen Sie die Datei aus, die Sie bearbeiten möchten. Alternativ können Sie File (Datei) und New File (Neue Datei) auswählen, um eine Datei zu erstellen.
3. Wenn Sie die Bearbeitung oder Erstellung der Dateien abgeschlossen haben, wechseln Sie im Terminal-Fenster zu dem Verzeichnis mit Ihrem geklonten Repository und führen Sie einen Commit durch. Übertragen Sie Ihre Änderungen dann per Push. Wenn Sie beispielsweise eine neue Datei mit dem Namen *`MyFile.py`* hinzugefügt haben:

```
cd MyDemoRepo
git commit -a MyFile.py
git commit -m "Added a new file with some code improvements"
git push
```

Nächste Schritte

Weitere Informationen finden Sie im .AWS Cloud9-BenutzerhandbuchundCodeCommit-Beispiel fürAWS Cloud9aus. Weitere Informationen zur Verwendung von Git mit CodeCommit finden Sie unterErste Schritte mit Git und AWS CodeCommitaus.

Integrieren Sie Visual Studio mitAWS CodeCommit

Sie können Visual Studio verwenden, um Codeänderungen in einem CodeCommit Repository vorzunehmen. Das enthältAWS Toolkit for Visual Studio jetzt Funktionen, die das Arbeiten mit Visual Studio CodeCommit einfacher und komfortabler machen.. Das Toolkit for Visual Studio Studio-Integration ist für die Verwendung mit Git-Anmeldeinformationen und einem IAM-Benutzer konzipiert. Sie können vorhandene Repositorys klonen, Repositorys erstellen, Code-Änderungen bestätigen und speichern, und vieles mehr.

> **Important**
>
> Das Toolkit for Visual Studio ist nur für die Installation auf Windows-Betriebssystemen verfügbar. Informationen zur Arbeit mit Visual Studio Code finden Sie unter AWS Toolkit for Visual Studio Code.

Wenn Sie das Toolkit for Visual Studio schon einmal verwendet haben, sind Sie wahrscheinlich bereits mit der Einrichtung vonAWS Anmeldeinformationsprofilen vertraut, die einen Zugriffsschlüssel und einen geheimen Schlüssel enthalten. Anmeldeinformationsprofile werden im Toolkit for Visual Studio verwendet, um Aufrufe vonAWS Service-APIs zu ermöglichen (z. B. an Amazon S3, um Buckets aufzulisten oder CodeCommit um Repositorys aufzulisten). Um Code abzurufen und in ein CodeCommit Repository zu übertragen, benötigen Sie außerdem Git-Anmeldeinformationen. Wenn Sie keine Git-Anmeldeinformationen haben, kann das Toolkit for Visual Studio diese Anmeldeinformationen für Sie generieren und anwenden. Dadurch können Sie viel Zeit sparen.

Um Visual Studio mit verwenden zu können CodeCommit, benötigen Sie Folgendes:

- Ein IAM-Benutzer mit einem gültigen Satz von Anmeldeinformationen (ein Zugriffsschlüssel und ein geheimer Schlüssel), die für ihn konfiguriert sind. Dieser IAM-Benutzer sollte außerdem über Folgendes verfügen:

  Eine der CodeCommit verwalteten Richtlinien und dieIAMSelfManageServiceSpecificCredentials verwaltete Richtlinie wurden darauf angewendet.

ODER

Wenn der IAM-Benutzer bereits Git-Anmeldeinformationen konfiguriert hat, eine der CodeCommit verwalteten Richtlinien oder entsprechende Berechtigungen.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit und Verstehen und Abrufen Ihrer Sicherheitsanmeldeinformationen.

- Die auf dem ComputerAWS Toolkit for Visual Studio installiert sind, auf dem Sie Visual Studio installiert haben. Weitere Informationen finden Sie unter Einrichten von AWS Toolkit for Visual Studio.

Weitere Informationen zur VerwendungAWS Toolkit for Visual Studio mit CodeCommit finden Sie unter VerwendenAWS CodeCommit mit Visual Studio Team Explorer im Toolkit for Visual Studio Studio-Benutzerhandbuch.

Integration von Eclipse in AWS CodeCommit

Sie können Eclipse verwenden, um Codeänderungen in einem CodeCommit Repository vorzunehmen. Das Toolkit for Eclipse-Integration ist für die Verwendung mit Git-Anmeldeinformationen und einem IAM-Benutzer konzipiert. Sie können vorhandene Repositorys klonen, Repositorys erstellen, Code-Änderungen bestätigen und speichern, und vieles mehr.

Um Toolkit for Eclipse mit verwenden zu können CodeCommit, benötigen Sie Folgendes:

- Eclipse muss auf Ihrem lokalen Computer installiert sein.
- Ein IAM-Benutzer mit einem gültigen Satz von Anmeldeinformationen (ein Zugriffsschlüssel und ein geheimer Schlüssel), die für ihn konfiguriert sind. Dieser IAM-Benutzer sollte außerdem über Folgendes verfügen:

  Eine der CodeCommit verwalteten Richtlinien und dieIAMSelfManageServiceSpecificCredentials verwaltete Richtlinie wurden darauf angewendet.

  ODER

  Wenn der IAM-Benutzer bereits Git-Anmeldeinformationen konfiguriert hat, eine der CodeCommit verwalteten Richtlinien oder entsprechende Berechtigungen.

Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit und Verstehen und Abrufen Ihrer Sicherheitsanmeldeinformationen.

- Ein aktiver Satz von Git-Anmeldeinformationen, die für den Benutzer in IAM konfiguriert wurden. Weitere Informationen finden Sie unter Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit.

Themen

- Schritt 1: Besorgen eines Zugriffsschlüssels und eines geheimen Schlüssels für Ihren IAM-Benutzer
- Schritt 2: InstallierenAWS Toolkit for Eclipse und verbinden CodeCommit
- Klonen Sie ein CodeCommit Repository aus Eclipse
- Erstellen Sie ein CodeCommit Repository aus Eclipse
- Verwenden von CodeCommit Repositorys

Schritt 1: Besorgen eines Zugriffsschlüssels und eines geheimen Schlüssels für Ihren IAM-Benutzer

Wenn Sie auf dem Computer, auf dem Eclipse installiert ist, noch kein Anmeldeinformationsprofil eingerichtet haben, können Sie eines mit demaws configure BefehlAWS CLI und konfigurieren. Alternativ können Sie dieses Verfahren zum Erstellen und Herunterladen Ihrer Anmeldeinformationen verwenden. Stellen Sie sie dem Toolkit for Eclipse zur Verfügung, wenn Sie dazu aufgefordert werden.

Benutzer benötigen programmgesteuerten Zugriff, wenn sie außerhalb der AWS Management Console mit AWS interagieren möchten. Die Vorgehensweise, um programmgesteuerten Zugriff zu gewähren, hängt davon ab, welcher Benutzertyp auf AWS zugreift:

Wählen Sie eine der folgenden Optionen aus, um den Benutzern programmgesteuerten Zugriff zu gewähren.

Welcher Benutzer benötigt programmgesteuerten Zugriff?	Bis	Von
Mitarbeiteridentität (Benutzer, die in IAM Identity Center verwaltet werden)	Verwenden Sie temporäre Anmeldeinformationen, um programmatische Anfragen an die AWS CLI, AWS SDKs oder AWS APIs zu signieren.	Folgen Sie den Anweisungen für die Schnittstelle, die Sie verwenden möchten: • Weitere Informationen zur AWS CLI finden Sie unter Konfigurieren der AWS CLI zur Verwendung von AWS IAM Identity Center im AWS Command Line Interface-Benutzerhandbuch. • Informationen zu AWS SDKs, Tools und AWS APIs finden Sie unter IAM Identity Center-Authentifizierung im Referenzhandbuch für AWS-SDKs und -Tools.
IAM	Verwenden Sie temporäre Anmeldeinformationen, um programmatische Anfragen an die AWS CLI, AWS SDKs oder AWS APIs zu signieren.	Folgen Sie den Anweisungen unter Verwenden temporärer Anmeldeinformationen mit AWS-Ressourcen im IAM-Benutzerhandbuch.
IAM	(Nicht empfohlen) Verwenden Sie langfristige Anmeldeinformationen, um programmgesteuerte Anforderungen an die AWS CLI, AWS SDKs oder AWS APIs zu signieren.	Folgen Sie den Anweisungen für die Schnittstelle, die Sie verwenden möchten: • Informationen zur AWS CLI finden Sie unter Authentifizierung mit IAM-Benutzeranmeldeinformationen im AWS Command

Welcher Benutzer benötigt programmgesteuerten Zugriff?	Bis	Von
		Line Interface-Benutzer handbuch. • Informationen zu den AWS-SDKs und -Tools finden Sie unter Authentifizierung mit langfristigen Anmeldeinformationen im Referenzhandbuch für AWS-SDKs und -Tools. • Informationen zu AWS-APIs siehe Verwalten von Zugriffsschlüsseln für IAM-Benutzer im IAM-Benutzerhandbuch.

Schritt 2: InstallierenAWS Toolkit for Eclipse und verbinden CodeCommit

Das Toolkit for Eclipse ist ein Softwarepaket, das Sie zu Eclipse hinzufügen können. Nachdem Sie es installiert und mit IhremAWS Anmeldeinformationsprofil konfiguriert haben, können Sie vomAWS Explorer in Eclipse CodeCommit aus eine Verbindung herstellen.

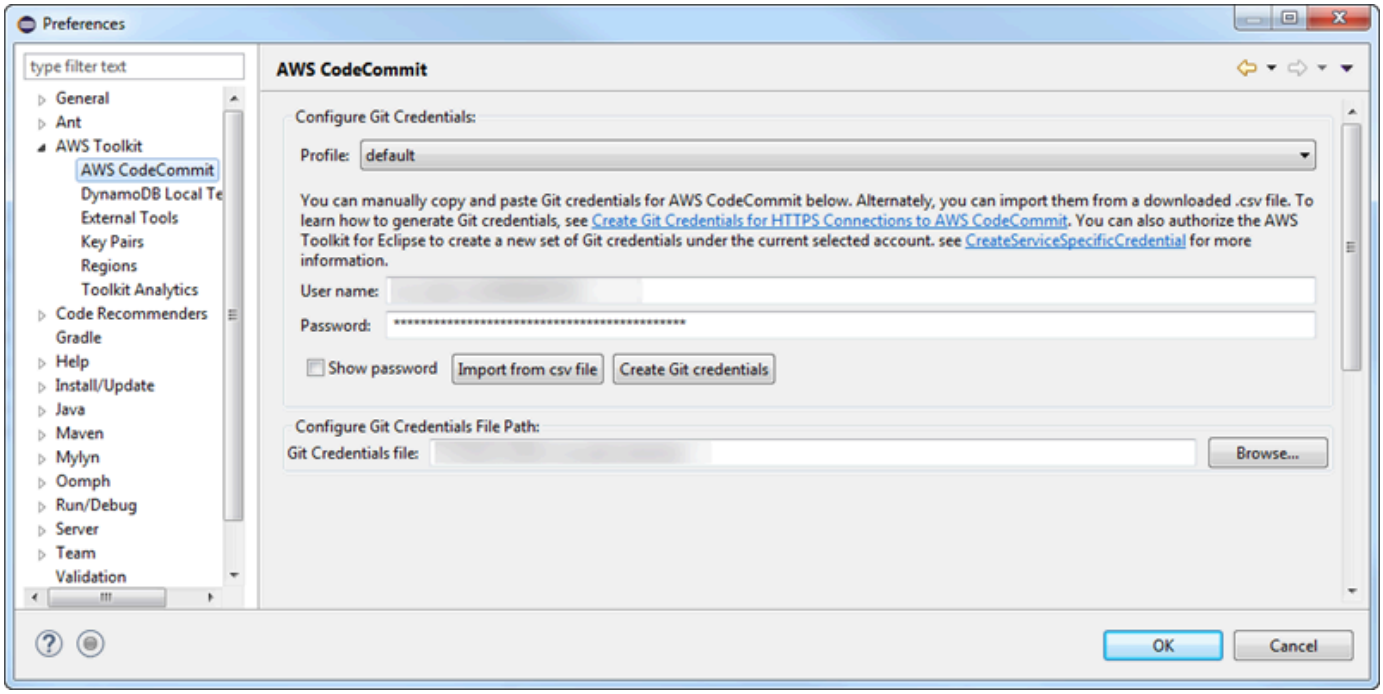
Um das Toolkit for Eclipse mit demAWS CodeCommit Modul zu installieren und den Zugriff auf Ihr Projekt-Repository zu konfigurieren

1. Installieren Sie Toolkit for Eclipse auf Ihrem lokalen Computer, falls Sie noch keine unterstützte Version installiert haben. Wenn Sie Ihre Version von Toolkit for Eclipse aktualisieren müssen, folgen Sie den Anweisungen unter Toolkit einrichten.
2. Folgen Sie in Eclipse entweder dem Einrichtungsassistenten oder öffnen Sie Preferences (Präferenzen) aus dem Eclipse-Menüsystem (der genaue Speicherort variiert je nach Version und Betriebssystem) und wählen Sie AWS Toolkit aus.
3. Führen Sie eine der folgenden Aktionen aus:

   • Wenn Sie der ersten Ausführung folgen, geben Sie IhreAWS Sicherheitsdaten ein, wenn Sie dazu aufgefordert werden, Ihr Anmeldeinformationsprofil einzurichten.

- Wenn Sie die Konfiguration in Preferences (Präferenzen) vornehmen und auf Ihrem Computer bereits ein Anmeldeinformationsprofil konfiguriert ist, wählen Sie es unter Default Profile (Standardprofil) aus.
- Wenn Sie die Konfiguration in Preferences vornehmen und das gewünschte Profil nicht angezeigt wird oder die Liste leer ist, wählen Sie Add profile aus. Geben Sie unter Profildetails einen Namen für das Profil und die Anmeldeinformationen für den IAM-Benutzer (Zugriffsschlüssel und geheimer Schlüssel) ein, oder geben Sie alternativ den Speicherort der Anmeldeinformationsdatei ein.
- Wenn Sie die Konfiguration in Preferences (Präferenzen) vornehmen und noch kein Profil konfiguriert ist, verwenden Sie die Links zur Registrierung für ein Konto oder zur Verwaltung Ihrer vorhandenen AWS-Sicherheitsanmeldeinformationen.

4. Erweitern Sie in Eclipse das AWSToolkit-Menü und wählen Sie AWS CodeCommit. Wählen Sie Ihr Anmeldeinformationenprofil aus und geben Sie dann den Benutzernamen und das Passwort für Ihre Git-Anmeldeinformationen ein oder importieren Sie sie aus der CSV-Datei. Wählen Sie Übernehmen und anschließend OK aus.



Nachdem Sie über ein Profil angemeldet haben, wird der AWS CodeCommit-Verbindungsbereich im Team Explorer anzeigt, wo Sie die Optionen zum Klonen, zum Erstellen oder zum Abmelden erhalten. Wenn Sie Clone wählen, wird ein vorhandenes CodeCommit Repository auf Ihren lokalen Computer geklont, sodass Sie mit der Arbeit am Code beginnen können. Die ist die am häufigsten verwendete Option.

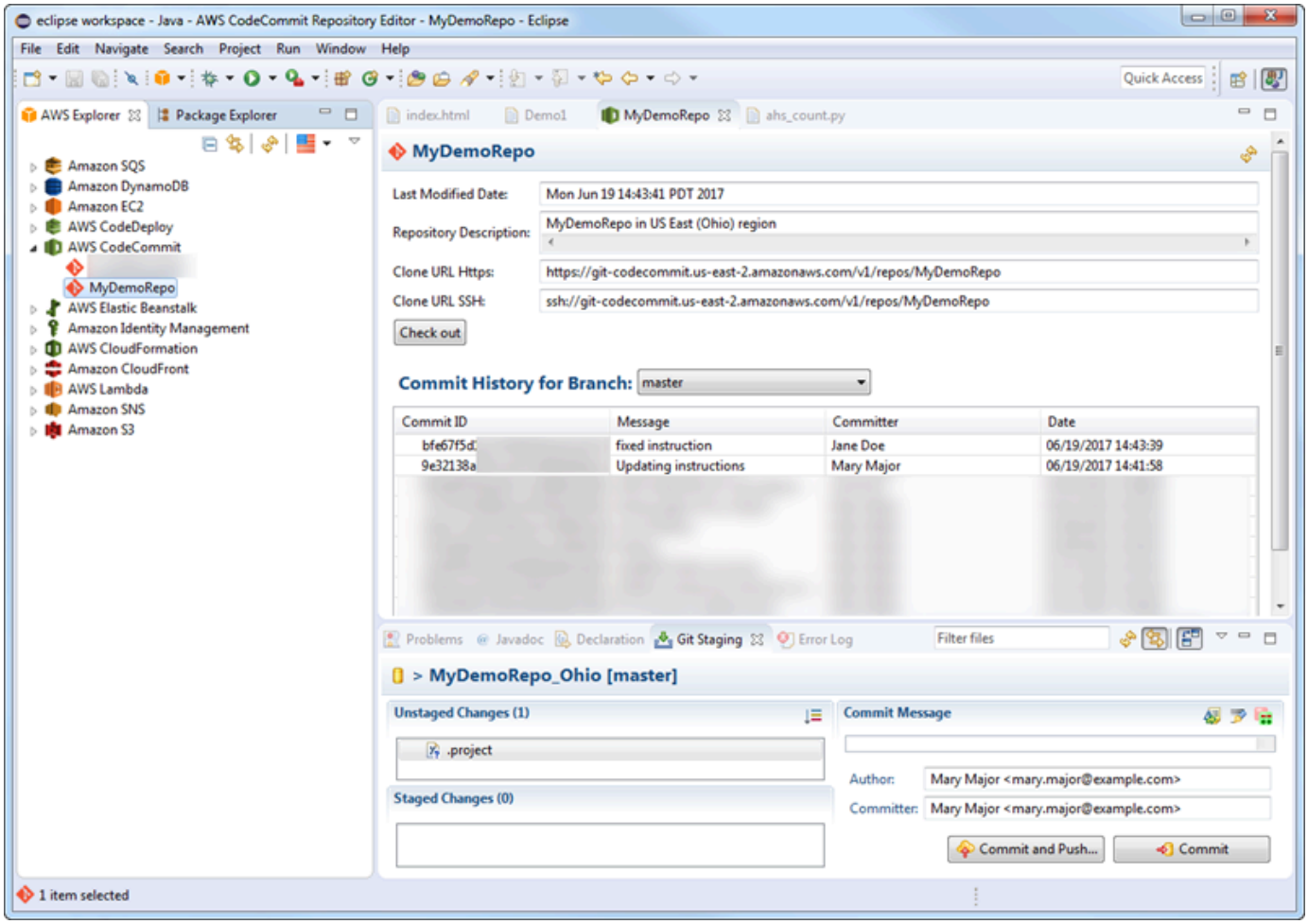
Wenn Sie keine Repositorys haben oder ein Repository erstellen möchten, wählen Sie Create aus.

Klonen Sie ein CodeCommit Repository aus Eclipse

Nachdem Sie Ihre Anmeldeinformationen konfiguriert haben, können Sie ein Repository in ein lokales Repository auf Ihren Computer klonen, indem Sie es in Eclipse auschecken. Anschließend können Sie den Code bearbeiten.

1. Öffnen Sie in Eclipse den AWSExplorer. Weitere Informationen zum Aufrufen der Option finden Sie unter Zugriff auf AWS Explorer. Erweitern Sie AWS CodeCommit und wählen Sie das CodeCommit -Repository aus, in dem Sie arbeiten möchten. Sie können den Commit-Verlauf und weitere Details des Repositorys anzeigen, um zu ermitteln, ob es sich um das Repository und den Branch handelt, die Sie klonen möchten.

   > **Note**
   >
   > Wenn Sie Ihr Repository nicht sehen, wählen Sie das Flaggensymbol, um dasAWS-Regionen Menü zu öffnen, und wählen Sie denAWS-Region Ort aus, an dem das Repository erstellt wurde.



2. Wählen Sie Check out und befolgen Sie die Anweisungen zum Klonen des Repositorys auf Ihren lokalen Computer.
3. Wenn Sie das Klonen des Projekts abgeschlossen haben, können Sie mit dem Bearbeiten Ihres Codes in Eclipse beginnen und Ihre Änderungen vornehmen, einen Commit durchführen und sie auf das Repository Ihres Projekts in CodeCommit übertragen.

Erstellen Sie ein CodeCommit Repository aus Eclipse

Mit dem Toolkit for Eclipse können Sie CodeCommit Repositorys aus Eclipse erstellen. Beim Erstellen des Repositorys klonen Sie dieses außerdem in ein lokales Repository auf Ihrem Computer, sodass Sie sofort mit der Arbeit beginnen können.

1. Klicken Sie imAWS Explorer mit der rechten Maustaste AWS CodeCommitund wählen Sie dann Repository erstellen.

> **Note**
>
> Repositorys sind regionsspezifisch. Bevor Sie das Repository erstellen, stellen Sie sicher, dass Sie das richtige ausgewählt habenAWS-Region. Sie können das nicht auswählen,AWS-Region nachdem Sie den Repository-Erstellungsprozess gestartet haben.

2. Geben Sie im Feld Repository Name (Repository-Name) einen Namen für dieses Repository ein. Repository-Namen müssen innerhalb eines Amazon Web Services-Kontos eindeutig sein. Es gibt Einschränkungen im Hinblick auf Zeichen und Länge. Weitere Informationen finden Sie unter Kontingente. Optional können Sie in das Feld Repository Description (Repository-Beschreibung) eine optionale Beschreibung für dieses Repository eingeben. Auf diese Weise wissen andere Benutzer, wofür dieses Repository vorgesehen ist, und es kann von anderen Repositorys in der Region unterschieden werden. Wählen Sie OK.
3. Erweitern Sie imAWS Explorer und wählen Sie dann das CodeCommit Repository aus AWS CodeCommit, das Sie gerade erstellt haben. Sie sehen, dass dieses Repository nicht über einen Commit-Verlauf verfügt. Wählen Sie Check out und befolgen Sie die Anweisungen zum Klonen des Repositorys auf Ihren lokalen Computer.

Verwenden von CodeCommit Repositorys

Nachdem Sie eine Verbindung hergestellt haben CodeCommit, können Sie imAWS Explorer eine Liste der Repositorys sehen, die mit Ihrem Konto verknüpft sind.AWS-Region Die Region können Sie im Flag-Menü ändern.

> **Note**
>
> CodeCommit ist möglicherweise nicht in allen vom Toolkit for EclipseAWS-Regionen unterstützten Versionen verfügbar.

In Toolkit for Eclipse können Sie den Inhalt dieser Repositorys in den Ansichten Navigation und Package Explorer durchsuchen. Um eine Datei zu öffnen, wählen Sie sie aus der Liste aus.

Git-Operationen in Toolkit for Eclipse für CodeCommit Repositorys funktionieren genauso wie für jedes andere Git-basierte Repository. Sie können Änderungen am Code vornehmen, Dateien hinzufügen und lokale Commits erstellen. Wenn du bereit bist, sie zu teilen, verwendest du die

Git Staging-Option, um deine Commits ins CodeCommit Repository zu übertragen. Wenn Sie noch keine Autoren- und Committer-Angaben in einem Git-Profil konfiguriert haben, können Sie dies tun, bevor Sie einen Commit für Ihre Änderungen durchführen und diese übertragen. Da Ihre Git-Anmeldeinformationen für Ihren IAM-Benutzer bereits lokal gespeichert und mit Ihrem verbundenenAWS Anmeldeinformationsprofil verknüpft sind, werden Sie nicht erneut aufgefordert, sie einzugeben, wenn Sie zu wechseln CodeCommit.

Weitere Informationen zum Verwenden von Toolkit for Eclipse finden Sie im AWS Toolkit for EclipseGetting Started Guide.

Setup für SSH-Benutzer, die dieAWS CLI

Wenn Sie SSH-Verbindungen für Ihr Repository verwenden möchten, können Sie eine Verbindung zu AWS CodeCommit herstellen, ohne die AWS CLI zu installieren. DieAWS CLIenthält Befehle, die bei der Verwendung und Verwaltung von CodeCommit-Repositorys nützlich sein werden. Sie ist jedoch für die erstmalige Einrichtung nicht erforderlich.

In diesem Thema wird Folgendes vorausgesetzt:

- Sie haben einen IAM-Benutzer mit den für CodeCommit und derIAMUserSSHKeysverwaltete Richtlinie oder gleichwertige Berechtigungen, die zum Hochladen von Schlüsseln erforderlich sind. Weitere Informationen finden Sie unter Verwendung identitätsbasierter Richtlinien (IAM-Richtlinien) für CodeCommit .
- Sie haben bereits ein Schlüsselpaar aus einem privaten und einem öffentlichen Schlüssel bzw. wissen, wie ein solches erstellt wird. Wir empfehlen dringend, eine sichere Pass-Phrase für den SSH-Schlüssel zu verwenden.
- Sie sind mit SSH, Ihrem Git-Client und den entsprechenden Konfigurationsdateien vertraut.
- Wenn Sie Windows verwenden, haben Sie ein Befehlszeilen-Dienstprogramm wie Git Bash installiert, das die Bash-Shell emuliert.

Wenn Sie weitere Anleitungen benötigen, folgen Sie den Anweisungen unter Für SSH-Verbindungen unter Linux, macOS oder Unix oder Für SSH-Verbindungen unter Windows.

Themen

- Schritt 1: Verknüpfen Sie Ihren öffentlichen Schlüssel mit dem IAM-Benutzer
- Schritt 2: Fügen Sie CodeCommit zu Ihrer SSH-Konfiguration hinzu
- Nächste Schritte

Schritt 1: Verknüpfen Sie Ihren öffentlichen Schlüssel mit dem IAM-Benutzer

1. Melden Sie sich bei der AWS Management Console an, und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich der IAM-KonsoleBenutzerWählen Sie in der Liste der Benutzer Ihren IAM-Benutzer aus.
3. Wählen Sie auf der Registerkarte Security Credentials die Option Upload SSH public key aus.
4. Kopieren Sie die Inhalte des öffentlichen SSH-Schlüssels in das Feld und wählen Sie dann Upload SSH Key aus.

> **Tip**
>
> Das Schlüsselpaar aus einem privaten und einem öffentlichen Schlüssel muss vom Schlüsseltyp SSH-2 RSA sein, im OpenSSH-Format vorliegen und 2048 Bits enthalten. Der Schlüssel sieht wie folgt aus:
>
> ```
> ssh-rsa EXAMPLE-
> AfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJB
> gNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb2
> 5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhc
> NMTEwNDI1MjA0NTIxWhcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAw
> DgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDAS=EXAMPLE user-
> name@ip-192-0-2-137
> ```
>
> IAM; akzeptiert nur öffentliche Schlüssel im OpenSSH-Format. Wenn Sie den öffentlichen Schlüssel in einem anderen Format bereitstellen, wird eine Fehlermeldung angezeigt, dass das Schlüsselformat ungültig ist.

5. Kopieren Sie die SSH-Schlüssel-ID (beispielsweise *APKAEIBAERJR2EXAMPLE*) und schließen Sie die Konsole.

Schritt 2: Fügen Sie CodeCommit zu Ihrer SSH-Konfiguration hinzu

1. Bearbeiten Sie im Terminal (Linux, macOS oder Unix) oder im Bash-Emulator (Windows) die SSH-Konfigurationsdatei, indem Sie eingebencat>> ~/.ssh/config:

```
Host git-codecommit.*.amazonaws.com
User Your-SSH-Key-ID, such as APKAEIBAERJR2EXAMPLE
IdentityFile Your-Private-Key-File, such as ~/.ssh/codecommit_rsa or ~/.ssh/id_rsa
```

> **Tip**
>
> Wenn Sie mehr als eine SSH-Konfiguration haben, stellen Sie sicher, dass Sie die Leerzeilen vor und nach dem Inhalt einfügen. Speichern Sie die Datei, indem Sie gleichzeitig die Tasten `Ctrl` und d drücken.

2. Führen Sie den folgenden Befehl aus, um Ihre SSH-Konfiguration zu testen:

```
ssh git-codecommit.us-east-2.amazonaws.com
```

Geben Sie die Pass-Phrase für die SSH-Schlüsseldatei ein, wenn Sie dazu aufgefordert werden. Wenn alles ordnungsgemäß konfiguriert ist, sollten Sie folgende Erfolgsmeldung sehen:

```
You have successfully authenticated over SSH. You can use Git to interact with
 CodeCommit.
```

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Führen Sie die Schritte unter ausErste Schritte mit CodeCommit um CodeCommit zu verwenden.

Um die Verbindung zu einem bestehenden Repository herzustellen, folgen Sie den Schritten unter Herstellen einer Verbindung mit einem Repository. Um ein Repository zu erstellen, folgen Sie den Schritten unter Erstellen eines -Repositorys.

Schritte zur Einrichtung von SSH-Verbindungen zuAWS CodeCommit Repositorys unter Linux, macOS oder Unix

Bevor SieCodeCommit erstmals eine Verbindung mit herstellen können, müssen Sie zunächst einige Schritte zur Konfiguration abschließen. Nachdem Sie Ihren Computer und IhrAWS Profil eingerichtet haben, können Sie eine Verbindung zu einemCodeCommit Repository herstellen und dieses Repository auf Ihren Computer klonen (auch bekannt als Erstellen eines lokalen Repositorys). Wenn Sie mit Git noch nicht vertraut sind, finden Sie unter Wo kann ich mehr über Git erfahren? genauere Informationen.

Themen

- Schritt 1: Erstkonfiguration fürCodeCommit
- Schritt 2: Installieren von Git
- Schritt 3: Konfiguration von Anmeldeinformationen unter Linux, macOS oder Unix
- Schritt 4: Connect derCodeCommit Konsole und Klonen des Repositorys
- Nächste Schritte

Schritt 1: Erstkonfiguration fürCodeCommit

Folgen Sie diesen Schritten, um ein Amazon Web Services Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und den Zugriff auf zu zu konfigurierenCodeCommit.

So erstellen und konfigurieren Sie einen IAM-Benutzer für den ZugriffCodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu http://aws.amazon.com gehen und „Anmelden“ wählen.
2. Erstellen Sie einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto, oder verwenden Sie einen vorhandenen. Vergewissern Sie sich, dass Sie eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel haben, die mit diesem IAM-Benutzer verknüpft sind. Weitere Informationen finden Sie unter Erstellen eines IAM-Benutzers in Ihrem Amazon Web Services Services-Konto.

   > **Note**
   >
   > CodeCommit erfordert AWS Key Management Service Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien

zugeordnet sind, die ausdrücklich die von gefordertenAWS KMS Aktionen verweigernCodeCommit. Weitere Informationen finden Sie unter AWS KMS und Verschlüsselung.

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
4. Wählen Sie im Navigationsbereich Users Users und anschließend den Option Users und anschließend denCodeCommit Zugriff konfigurieren möchten.
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie in der Richtlinienliste den Eintrag AWSCodeCommitPowerUser oder eine andere verwaltete Richtlinie für den CodeCommit-Zugriff aus. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit.

   Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, klicken Sie auf Weiter: Überprüfen, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

   Weitere Informationen über die verwalteten Richtlinien von CodeCommit und die Freigabe von Repositorys für andere Gruppen und Benutzer finden Sie unter Teilen Sie ein Repository und Authentifizierung und Zugriffskontrolle für AWS CodeCommit.

> **Note**
>
> Installieren Sie die AWS CLI, wenn Sie AWS CLI-Befehle mit CodeCommit verwenden möchten. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

Schritt 2: Installieren von Git

Um mit Dateien, Commits und anderen Informationen inCodeCommit Repositorys zu arbeiten, musst du Git auf deinem lokalen Computer installieren. CodeCommit unterstützt die Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für erste Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie Git Downloads.

> **Note**
>
> Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Es kommt vor, dass sich die Änderung eines Features darauf auswirkt, wie Git mit CodeCommit zusammen funktioniert. Wenn Sie Probleme mit einer bestimmten Version von Git und CodeCommit haben, lesen Sie die Informationen in Fehlerbehebung.

Schritt 3: Konfiguration von Anmeldeinformationen unter Linux, macOS oder Unix

SSH und Linux, macOS oder Unix: Richte die öffentlichen und privaten Schlüssel für Git ein undCodeCommit

Um die öffentlichen und privaten Schlüssel für Git einzurichten undCodeCommit

1. Führen Sie im Terminalfenster Ihres lokalen Computers den Befehl ssh-keygen aus und folgen Sie den Anweisungen, um die Datei im Verzeichnis ".ssh" für Ihr Profil zu speichern.

   > **Note**
   >
   > Fragen Sie den Systemadministrator nach dem gewünschten Speicherort der Schlüsseldateien und nach dem Muster, nach dem die Dateien benannt werden sollten.

   Beispiel:

   ```
   $ ssh-keygen

   Generating public/private rsa key pair.
   Enter file in which to save the key (/home/user-name/.ssh/id_rsa): Type /home/
   your-user-name/.ssh/ and a file name here, for example /home/your-user-name/.ssh/
   codecommit_rsa

   Enter passphrase (empty for no passphrase): <Type a passphrase, and then press
    Enter>
   Enter same passphrase again: <Type the passphrase again, and then press Enter>

   Your identification has been saved in /home/user-name/.ssh/codecommit_rsa.
   ```

```
Your public key has been saved in /home/user-name/.ssh/codecommit_rsa.pub.
The key fingerprint is:
45:63:d5:99:0e:99:73:50:5e:d4:b3:2d:86:4a:2c:14 user-name@client-name
The key's randomart image is:
+--[ RSA 2048]----+
|        E.+.o*.++|
|        .o .=.=o.|
|       . ..  *. +|
|        ..o . +..|
|        So . . . |
|          .      |
|                 |
|                 |
|                 |
+-----------------+
```

Folgendes wird generiert:

- Die Datei *codecommit_rsa* – das ist die Datei mit dem privaten Schlüssel.
- Die Datei *codecommit_rsa*.pub – das ist die Datei mit dem öffentlichen Schlüssel.

> **Tip**
>
> ssh-keygenGeneriert standardmäßig einen 2048-Bit-Schlüssel. Sie können die Parameter -t und -b verwenden, um den Typ und die Länge des Schlüssels anzugeben. Wenn Sie einen 4096-Bit-Schlüssel im RSA-Format benötigen, geben Sie dies an, indem Sie den Befehl mit den folgenden Parametern ausführen:
>
> ```
> ssh-keygen -t rsa -b 4096
> ```
>
> Weitere Informationen zu den Formaten und Längen, die für SSH-Schlüssel erforderlich sind, finden Sie unter Using IAM withCodeCommit.

2. Führen Sie den folgenden Befehl aus, um den Wert der Datei mit dem öffentlichen Schlüssel anzuzeigen (*codecommit_rsa*.pub):

```
cat ~/.ssh/codecommit_rsa.pub
```

Kopieren Sie diesen Wert. Er sieht in etwa wie folgt aus:

```
ssh-rsa EXAMPLE-AfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJB
gNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb2
5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhc
NMTEwNDI1MjA0NTIxWhcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAw
DgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDAS=EXAMPLE user-name@ip-192-0-2-137
```

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

> **Note**
>
> Sie können Ihre CodeCommit-Anmeldeinformationen unter My Security Credentials (Meine Sicherheitsanmeldeinformationen) direkt anzeigen und verwalten. Weitere Informationen finden Sie unter Ihre Anmeldeinformationen einsehen und verwalten.

4. Wählen Sie im Navigationsbereich Users Users Users und anschließend im Navigationsbereich Users Users und anschließend die Option.
5. Wählen Sie auf der Detailseite des Benutzers auf der Registerkarte Security Credentials die Option Upload SSH public key aus.
6. Kopieren Sie den Inhalt des öffentlichen SSH-Schlüssels in das Feld und wählen Sie dann Upload SSH public key aus.
7. Kopieren oder speichern Sie die Informationen unter SSH Key ID (z. B. *APKAEIBAERJR2EXAMPLE*).

> **Note**
>
> Wenn Sie mehr als eine SSH-Schlüssel-IDs hochgeladen haben, werden die Schlüssel alphabetisch nach Schlüssel-ID und nicht nach dem Datum des Hochladens aufgeführt. Stellen Sie sicher, dass Sie die Schlüssel-ID kopiert haben, die dem richtigen Upload-Datum zugeordnet ist.

8. Erstellen Sie mit einem Texteditor auf dem lokalen Computer im Verzeichnis "~/.ssh" eine Konfigurationsdatei. Fügen Sie der Datei dann die folgenden Zeilen hinzu, wobei *User* die SSH-Schlüssel-ID ist, die Sie zuvor kopiert haben:

```
Host git-codecommit.*.amazonaws.com
  User APKAEIBAERJR2EXAMPLE
  IdentityFile ~/.ssh/codecommit_rsa
```

> **Note**
>
> Wenn Sie der Datei mit dem privaten Schlüssel einen anderen Namen als *codecommit_rsa* gegeben haben, verwenden Sie diesen hier.
> Sie können den SSH-Zugriff auf Repositorys in mehreren Amazon Web Services Services-Konten einrichten. Weitere Informationen finden Sie unterFehlerbehebung bei SSH-Verbindungen zuAWS CodeCommit.

Speichern Sie die Datei unter dem Namen `config`.

9. Führen Sie im Terminalfenster den folgenden Befehl aus, um die Berechtigungen für die Konfigurationsdatei zu ändern:

```
chmod 600 config
```

10. Führen Sie den folgenden Befehl aus, um Ihre SSH-Konfiguration zu testen:

```
ssh git-codecommit.us-east-2.amazonaws.com
```

Sie werden gebeten, die Verbindung zu bestätigen, da sie noch nicht in Ihrer bekannten Hosts-Datei enthaltengit-codecommit.us-east-2.amazonaws.com ist. DerCodeCommit Serverfingerabdruck wird im Rahmen der Überprüfung angezeigt (a9:6d:03:ed:08:42:21:be:06:e1:e0:2a:d1:75:31:5efür MD5 oder3lBlW2g5xn/NA2Ck6dyeJIrQOWvn7n8UEs56fG6ZIzQ für SHA256).

> **Note**
>
> CodeCommitServerfingerabdrücke sind für jeden einzigartigAWS-Region. Informationen zum Anzeigen der Serverfingerabdrücke für eineAWS-Region finden Sie unterServer-Fingerabdrücke für CodeCommit.

Nachdem Sie die Verbindung bestätigt haben, sollte jeweils eine Bestätigung angezeigt werden, dass der Server der Datei mit den bekannten Hosts hinzugefügt und die Verbindung hergestellt wurde. Wenn keine Erfolgsmeldung angezeigt wird, überprüfen Sie, ob Sie die`config` Datei im Verzeichnis ~/.ssh des IAM-Benutzers gespeichert haben, für den Sie den Zugriff konfiguriert habenCodeCommit, und ob Sie die richtige private Schlüsseldatei angegeben haben.

Informationen zur Behandlung von Problemen erhalten Sie, wenn Sie den`ssh` Befehl mit dem`-v` Parameter ausführen. Beispiel:

```
ssh -v git-codecommit.us-east-2.amazonaws.com
```

Weitere Informationen zum Beheben von Verbindungsproblemen finden Sie unter Fehlerbehebung bei SSH-Verbindungen zuAWS CodeCommit.

Schritt 4: Connect derCodeCommit Konsole und Klonen des Repositorys

Wenn Sie den Namen und die Verbindungsdetails für das CodeCommit-Repository bereits von einem Administrator erhalten haben, können Sie diesen Schritt überspringen und das Repository direkt klonen.

So stellen Sie eine Verbindung mit einem CodeCommit-Repository her

1. Öffnen Sie dieCodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl den Ort aus,AWS-Region an dem das Repository erstellt wurde. Repositorys sind spezifisch für einAWS-Region. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie

beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.

- Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen für Ihren IAM-Benutzer oder den in der enthaltenen Anmeldeinformationshelfer verwendenAWS CLI.
- Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
- Kopieren Sie die SSH-URL, wenn Sie mit Ihrem IAM-Benutzer ein öffentliches/privates SSH-Schlüsselpaar verwenden.

> **Note**
>
> Wenn Sie eine Willkommensseite statt einer Liste von Repositorys sehen, sind IhremAWS Konto in dem, inAWS-Region dem Sie angemeldet sind, keine Repositorys zugeordnet. Informationen zur Erstellung eines Repositorys finden Sie unter the section called “Erstellen eines -Repositorys”, oder befolgen Sie die Schritte im Tutorial Erste Schritte mit Git und CodeCommit.

4. Öffnen Sie ein -Terminalfenster. Führen Sie im Verzeichnis "/tmp" den git clone-Befehl mit der SSH-URL aus, die Sie kopiert haben, um das Repository zu klonen. Um beispielsweise ein Repository mit dem Namen eines lokalen Repositorys mit dem Namen *my-demo-repo*der Region USA Ost (Ohio) *MyDemoRepo*zu klonen, gehen Sie wie folgt vor:

```
git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

> **Note**
>
> Wenn der Verbindungstest erfolgreich war, der Klonbefehl jedoch nicht ausgeführt wurde, besitzen Sie vielleicht nicht die erforderlichen Zugriffsrechte für die Konfigurationsdatei oder eine andere Einstellung verursacht einen Konflikt mit der Konfigurationsdatei. Versuchen Sie erneut, die Verbindung herzustellen. Geben Sie dabei die SSH-Schlüssel-ID als Teil des Befehls an. Beispiel:
>
> ```
> git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
> ```

Weitere Informationen finden Sie unter Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen, aber die Verbindung schlägt auf Linux-, macOS- oder Unix-Systemen fehl.

Weitere Informationen über die Verbindungserstellung zu Repositorys finden Sie unter Connect zum CodeCommit Repository her, indem Sie das Repository klonen.

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Schritten unterErste Schritte mit CodeCommit , um mit der Verwendung zu beginnenCodeCommit.

Einrichtungsschritte für SSH-Verbindungen zuAWS CodeCommitRepositorys unter Windows

Bevor Sie eine Verbindung herstellen könnenAWS CodeCommitZum ersten Mal müssen Sie einige Schritte zur ersten Konfiguration ausführen. Nachdem Sie Ihren Computer eingerichtet haben undAWS-Profil können Sie eine Verbindung mit einem CodeCommit-Repository herstellen und dieses Repository auf Ihren Computer klonen (ein lokales Repository erstellen). Wenn Sie mit Git noch nicht vertraut sind, finden Sie unter Wo kann ich mehr über Git erfahren? genauere Informationen.

Themen

- Schritt 1: Anfängliche Konfiguration für CodeCommit
- Schritt 2: Installieren Sie Git
- Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein
- Schritt 4: Connect Sie sich mit der CodeCommit-Konsole und klonen Sie das Repository
- Nächste Schritte

Schritt 1: Anfängliche Konfiguration für CodeCommit

Führen Sie die folgenden Schritte aus, um ein Amazon Web Services Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und den Zugriff auf CodeCommit zu konfigurieren.

So erstellen und konfigurieren Sie einen IAM-Benutzer für den -Zugriff auf CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie aufhttp://aws.amazon.comund wählenRegistrierenaus.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen Benutzer in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel haben, die mit diesem IAM-Benutzer verknüpft sind. Weitere Informationen finden Sie unterErstellen eines IAM-Benutzers in Ihrem Amazon Web Services Services-Kontoaus.

   > **Note**
   >
   > CodeCommit benötigtAWS Key Management Serviceaus. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, müssen Sie sicherstellen, dass mit diesem keine Richtlinien verknüpft sind, die ausdrücklich dieAWS KMSvon CodeCommit erforderliche Aktionen. Weitere Informationen finden Sie unter AWS KMS und Verschlüsselung .

3. Melden Sie sich bei der AWS Management Console an, und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
4. Wählen Sie im Navigationsbereich der IAM-KonsoleBenutzer, und wählen Sie dann den IAM-Benutzer aus, den Sie für CodeCommit-Zugriff konfigurieren möchten.
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie in der Liste der RichtlinienAWSCodeCommitPowerUseroder eine andere verwaltete Richtlinie für CodeCommit-Zugriff. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit .

   Nachdem Sie die Richtlinie zugewiesen haben, wählen SieWeiter: Prüfenum die Liste der Richtlinien anzuzeigen, um den IAM-Benutzer zuzuweisen. Ist die Liste korrekt, wählen Sie Add permissions aus.

   Weitere Informationen zu verwalteten CodeCommit-Richtlinien und die Freigabe des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unterTeilen Sie ein RepositoryundAuthentifizierung und Zugriffskontrolle für AWS CodeCommitaus.

> **Note**
>
> Wenn Sie verwenden möchtenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLIaus. Weitere Informationen finden Sie unter Befehlszeilenreferenz .

Schritt 2: Installieren Sie Git

Um mit Dateien, Commits und anderen Informationen in CodeCommit-Repositorys arbeiten zu können, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt die Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt das Konfigurieren des Zweignamens für erste Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Für die Installation von Git empfehlen wir Websites wieGit-Downloadsaus.

> **Note**
>
> Git ist eine stetig weiterentwickelte und regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Funktionsänderung auf die Art und Weise auswirken, wie sie mit CodeCommit funktioniert. Wenn Probleme mit einer bestimmten Version von Git und CodeCommit auftreten, lesen Sie die Informationen unterFehlerbehebungaus.

Wenn die installierte Git-Version keinen Bash-Emulator, z. B. Git Bash, umfasst, installieren Sie einen. Sie verwenden diesen Emulator beim Erstellen von SSH-Verbindungen anstelle der Windows-Befehlszeile.

Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein

So richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein(Windows)

1. Öffnen Sie den Bash-Emulator.

   > **Note**
   >
   > Eventuell müssen Sie den Emulator mit Administratorberechtigungen ausführen.

Führen Sie im Emulator den Befehl ssh-keygen aus und befolgen Sie die Anweisungen, um die Datei im Verzeichnis ".ssh" für Ihr Profil zu speichern.

Beispiel:

```
$ ssh-keygen

Generating public/private rsa key pair.
Enter file in which to save the key (/drive/Users/user-name/.ssh/id_rsa): Type a
 file name here, for example /c/Users/user-name/.ssh/codecommit_rsa

Enter passphrase (empty for no passphrase): <Type a passphrase, and then press
 Enter>
Enter same passphrase again: <Type the passphrase again, and then press Enter>

Your identification has been saved in drive/Users/user-name/.ssh/codecommit_rsa.
Your public key has been saved in drive/Users/user-name/.ssh/codecommit_rsa.pub.
The key fingerprint is:
45:63:d5:99:0e:99:73:50:5e:d4:b3:2d:86:4a:2c:14 user-name@client-name
The key's randomart image is:
+--[ RSA 2048]----+
|        E.+.o*.++|
|        .o .=.=o.|
|       . ..  *. +|
|        ..o . +..|
|        So . . . |
|          .      |
|                 |
|                 |
|                 |
+-----------------+
```

Folgendes wird generiert:

- Die Datei *codecommit_rsa* – das ist die Datei mit dem privaten Schlüssel.
- Die Datei *codecommit_rsa*.pub – das ist die Datei mit dem öffentlichen Schlüssel.

> **Tip**
>
> Der Standardwert für istssh-keygengeneriert einen 2048-Bit-Schlüssel. Sie können die Parameter -t und -b verwenden, um den Typ und die Länge des Schlüssels anzugeben. Wenn Sie einen 4096-Bit-Schlüssel im RSA-Format wünschen, geben Sie dies an, indem Sie den Befehl mit den folgenden Parametern ausführen:
>
> ```
> ssh-keygen -t rsa -b 4096
> ```
>
> Weitere Informationen zu den Formaten und Längen für SSH-Schlüssel finden Sie unterVerwenden von IAM mit CodeCommitaus.

2. Führen Sie die folgenden Befehle aus, um den Wert der Datei mit dem öffentlichen Schlüssel anzuzeigen (*codecommit_rsa*.pub):

```
cd .ssh
notepad codecommit_rsa.pub
```

Kopieren Sie den Inhalt der Datei und schließen Sie Notepad, ohne zu speichern. Der Inhalt der Datei ähnelt dem folgenden Beispiel:

```
ssh-rsa EXAMPLE-AfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMCVVMxCzAJB
gNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb2
5zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhc
NMTEwNDI1MjA0NTIxWhcNMTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAw
DgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDAS=EXAMPLE user-name@computer-name
```

3. Melden Sie sich bei der AWS Management Console an, und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

> **Note**
>
> Sie können Ihre CodeCommit-Anmeldeinformationen inMeine Sicherheitsanmeldeinformationenaus. Weitere Informationen finden Sie unter Ihre Anmeldeinformationen einsehen und verwalten .

4. Wählen Sie im Navigationsbereich der IAM-KonsoleBenutzerund wählen Sie in der Liste der Benutzer Ihren IAM-Benutzer aus.
5. Wählen Sie auf der Detailseite des Benutzers auf der Registerkarte Security Credentials die Option Upload SSH public key aus.
6. Kopieren Sie den Inhalt des öffentlichen SSH-Schlüssels in das Feld und wählen Sie dann Upload SSH public key aus.
7. Kopieren oder speichern Sie die Informationen unter SSH Key ID (z. B. *APKAEIBAERJR2EXAMPLE*).

> **Note**
>
> Wenn Sie mehr als eine SSH-Schlüssel-IDs hochgeladen haben, werden die Schlüssel alphabetisch nach Schlüssel-ID und nicht nach dem Datum des Hochladens aufgeführt. Stellen Sie sicher, dass Sie die Schlüssel-ID kopiert haben, die dem richtigen Upload-Datum zugeordnet ist.

8. Führen Sie im Bash-Emulator die folgenden Befehle aus, um im Verzeichnis "~/.ssh" eine Konfigurationsdatei zu erstellen bzw. zu bearbeiten, falls bereits eine solche Datei vorhanden ist:

```
notepad ~/.ssh/config
```

9. Fügen Sie der Datei die folgenden Zeilen hinzu, wobei *User* die SSH-Schlüssel-ID ist, die Sie zuvor kopiert haben, und der Wert für *IdentityFile* der Pfad und der Name der privaten Schlüsseldatei ist:

```
Host git-codecommit.*.amazonaws.com
  User APKAEIBAERJR2EXAMPLE
  IdentityFile ~/.ssh/codecommit_rsa
```

> **Note**
>
> Wenn Sie der Datei mit dem privaten Schlüssel einen anderen Namen als *codecommit_rsa* gegeben haben, verwenden Sie diesen hier.
> Sie können SSH-Zugriff auf Repositorys in mehreren Amazon Web Services Services-Konten einrichten, Weitere Informationen finden Sie unterFehlerbehebung bei SSH-Verbindungen zuAWS CodeCommitaus.

Speichern Sie die Datei als "config" (nicht "config.txt") und schließen Sie Notepad.

> **Important**
>
> Der Name der Datei muss `config` ohne Dateierweiterung lauten. Andernfalls schlagen die SSH-Verbindungen fehl.

10. Führen Sie den folgenden Befehl aus, um Ihre SSH-Konfiguration zu testen:

```
ssh git-codecommit.us-east-2.amazonaws.com
```

Sie werden aufgefordert, die Verbindung zu bestätigen, weil`git-codecommit.us-east-2.amazonaws.com`ist noch nicht in Ihrer bekannten Hosts-Datei enthalten.
Der Fingerabdruck des CodeCommit-Servers wird im Zuge der Überprüfung (`a9:6d:03:ed:08:42:21:be:06:e1:e0:2a:d1:75:31:5e`für MD5 oder`3lBlW2g5xn/NA2Ck6dyeJIrQOWvn7n8UEs56fG6ZIzQ`für SHA256).

> **Note**
>
> CodeCommit-Server-Fingerabdrücke sind für jeden einzigartigAWS-Regionaus. So zeigen Sie die Server-Fingerabdrücke für einAWS-Regionfinden Sie unter,Server-Fingerabdrücke für CodeCommitaus.

Nachdem Sie die Verbindung bestätigt haben, sollte jeweils eine Bestätigung angezeigt werden, dass der Server der Datei mit den bekannten Hosts hinzugefügt und die Verbindung hergestellt wurde. Wenn Ihnen keine Erfolgsmeldung angezeigt wird, müssen Sie überprüfen, ob Sie das`config`-Datei im Verzeichnis ~/.ssh des IAM-Benutzers, den Sie für den Zugriff

auf CodeCommit konfiguriert haben, dass der`config`hat keine Dateierweiterung (z. B. darf sie nicht config.txt heißen) und dass Sie die richtige private Schlüsseldatei angegeben haben (*`codecommit_rsa`*, nicht*`codecommit_rsa`*.pub).

Um Probleme zu beheben, führen Sie die`ssh`Befehl mit dem`-v`-Parameter. Beispiel:

```
ssh -v git-codecommit.us-east-2.amazonaws.com
```

Weitere Informationen zum Beheben von Verbindungsproblemen finden Sie unter Fehlerbehebung bei SSH-Verbindungen zuAWS CodeCommit.

Schritt 4: Connect Sie sich mit der CodeCommit-Konsole und klonen Sie das Repository

Wenn Sie den Namen und die Verbindungsdetails für das CodeCommit-Repository bereits von einem Administrator erhalten haben, können Sie diesen Schritt überspringen und das Repository klonen direkt.

So stellen Sie eine Verbindung mit einem CodeCommit-Repository her

1. Öffnen Sie die CodeCommit-Konsole unterhttps://console.aws.amazon.com/codesuite/codecommit/homeaus.
2. Wählen Sie in der Regionenauswahl die OptionAWS-Regionwo das Repository erstellt wurde. Die Repositorys sind spezifisch für eineAWS-Regionaus. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte .
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.

   - Kopieren Sie die HTTPS-URL, wenn Sie Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Hilfsprogramm für Anmeldeinformationen verwenden, die in derAWS CLIaus.
   - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
   - Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

> **Note**
>
> Wenn du eine siehstWillkommenSeite anstelle einer Liste mit Repositorys sind keine Repositorys mit Ihrem verknüpftAWSKonto imAWS-Regionwo Sie angemeldet sind. Informationen zur Erstellung eines Repositorys finden Sie unter the section called "Erstellen eines -Repositorys", oder befolgen Sie die Schritte im Tutorial Erste Schritte mit Git und CodeCommit.

4. Führen Sie im Bash-Emulator den git clone-Befehl mit der SSH-URL aus, die Sie kopiert haben, um das Repository zu klonen. Durch diesen Befehl wird in einem Unterverzeichnis des Verzeichnisses, in dem Sie den Befehl ausgeführt haben, das lokale Repository erstellt. Beispiel zum Klonen eines Repositorys namens*MyDemoRepo*zu einem lokalen Repo namens*my-demo-repo*In der Region USA Ost (Ohio):

```
git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-
repo
```

Alternativ können Sie eine Eingabeaufforderung öffnen und mithilfe der URL und der SSH-Schlüssel-ID für den öffentlichen Schlüssel, den Sie in IAM hochgeladen haben, dengit clonebefehl. Dadurch wird das lokale Repository in einem Unterverzeichnis des Verzeichnisses erstellt, in dem Sie den Befehl ausgeführt haben. Beispiel zum Klonen eines Repositorys namens*MyDemoRepo*zu einem lokalen Repo namens*my-demo-repo*:

```
git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/repos/
MyDemoRepo my-demo-repo
```

Weitere Informationen finden Sie unter Connect zum CodeCommit Repository her, indem Sie das Repository klonen und Erstellen Sie einen Commit.

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Führen Sie die Schritte unter ausErste Schritte mit CodeCommit um CodeCommit zu verwenden.

Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitRepositorys auf Linux, macOS oder Unix mit demAWS CLIHelfer für Anmeldeinformationen

Bevor Sie AWS CodeCommit verwenden können, müssen Sie die Schritte für die erste Konfiguration ausführen. Für die meisten Benutzer lässt sich das am einfachsten mit den Schritten unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden umsetzen. Wenn Sie jedoch über ein Root-Konto, über einen Verbundzugriff oder mit temporären Anmeldeinformationen eine Verbindung mit CodeCommit herstellen möchten, müssen Sie das AWS CLI-Hilfsprogramm für Anmeldeinformationen nutzen.

> **Note**
>
> Obwohl das Hilfsprogramm für Anmeldeinformationen eine unterstützte Methode für die Verbindung mit CodeCommit über den Verbundzugriff, über einen Identitätsanbieter oder mit temporären Anmeldeinformationen ist, empfiehlt sich die Installation und Verwendung des git-remote-codecommit-Dienstprogramms. Weitere Informationen finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit.

Themen

- Schritt 1: Erstkonfiguration fürCodeCommit
- Schritt 2: Installieren von Git
- Schritt 3: Richten Sie den Credential Helper ein
- Schritt 4: Stellen Sie eine Verbindung mit dem herCodeCommitkonsole und klone das Repository
- Nächste Schritte

Schritt 1: Erstkonfiguration fürCodeCommit

Gehen Sie wie folgt vor, um ein Amazon Web Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und zu konfigurieren und denAWS CLI.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurierenCodeCommit

1. Erstellen Sie ein Amazon Web Services-Konto, indem Sie zuhttp://aws.amazon.comund wählenMelde dich an.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unterEinen IAM-Benutzer in Ihrem Amazon Web Services-Konto erstellen.

   > **Note**
   >
   > CodeCommit erfordert AWS Key Management Service Wenn Sie einen bestehenden IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die das ausdrücklich verbietenAWS KMSAktionen erforderlich vonCodeCommit. Weitere Informationen finden Sie unter AWS KMS und Verschlüsselung.

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
4. Wählen Sie in der IAM-Konsole im NavigationsbereichBenutzer, und wählen Sie dann den IAM-Benutzer aus, für den Sie konfigurieren möchtenCodeCommitZugriff.
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie in der Richtlinienliste den Eintrag AWSCodeCommitPowerUser oder eine andere verwaltete Richtlinie für den CodeCommit-Zugriff aus. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit.

   Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen SieWeiter: Überprüfenum die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

   Weitere Informationen über die verwalteten Richtlinien von CodeCommit und die Freigabe von Repositorys für andere Gruppen und Benutzer finden Sie unter Teilen Sie ein Repository und Authentifizierung und Zugriffskontrolle für AWS CodeCommit.

So installieren und konfigurieren Sie die AWS CLI

1. Laden Sie die AWS CLI auf den lokalen Computer herunter und installieren Sie. Dies ist eine Voraussetzung für das Arbeiten mit CodeCommit über die Befehlszeile. Wir empfehlen die InstallationAWS CLIVersion 2. Es ist die neueste Hauptversion vonAWS CLIund unterstützt die neuesten Funktionen. Es ist die einzige Version vonAWS CLIdas unterstützt die Verwendung eines Root-Kontos, Verbundzugriffs oder temporärer Anmeldeinformationen mitgit-remote-codecommit.

   Weitere Informationen finden Sie unter Erste Schritte mit der AWS-Befehlszeilenschnittstelle.

   > **Note**
   >
   > CodeCommitfunktioniert nur mitAWS CLIVersionen 1.7.38 und höher. Als bewährte Methode installieren oder aktualisieren Sie die AWS CLI auf die neueste verfügbare Version. Führen Sie den Befehl aws --version aus, um zu überprüfen, welche Version der AWS CLI installiert ist.
   > Informationen zum Upgraden von einer älteren auf die aktuelle Version der AWS CLI finden Sie unter Installieren der AWS Command Line Interface.

2. Führen Sie diesen Befehl aus, um zu überprüfen, obCodeCommitBefehle für denAWS CLIsind installiert.

   ```
   aws codecommit help
   ```

   Dieser Befehl gibt eine Liste vonCodeCommitBefehle.

3. Konfigurieren Sie dieAWS CLImit einem Profil unter Verwendung desconfigureBefehl, wie folgt:.

   ```
   aws configure
   ```

   Wenn Sie dazu aufgefordert werden, geben Sie denAWSZugriffsschlüssel undAWSgeheimer Zugriffsschlüssel des IAM-Benutzers, mit dem verwendet werden sollCodeCommit. Stellen Sie außerdem sicher, dass Sie Folgendes angebenAWS-Regionwo das Repository existiert, z. B.`us-east-2`. Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

   ```
   AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press
    Enter
   ```

```
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and
 then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then
 press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen für die AWS CLI finden Sie unter:

- Benannte Profile
- Verwenden Sie eine IAM-Rolle in derAWS CLI
- Befehl „Set“
- Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen herzustellenAWS-Region, müssen Sie das neu konfigurierenAWS CLImit dem Standardnamen der Region. Zu den Standardregionsnamen für CodeCommit zählen:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1

- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3
- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und AWS-Region finden Sie unter Regionen und Git-Verbindungsendpunkte. Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unterWie erhalte ich Zugangsdaten?undVerwaltung von Zugriffsschlüsseln für IAM-Benutzer. Weitere Informationen zur AWS CLI und zu Profilen finden Sie unter Benannte Profile.

Schritt 2: Installieren von Git

So arbeiten Sie mit Dateien, Commits und anderen Informationen inCodeCommitRepositorys müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt die Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wieGit-Downloads.

> **Note**
>
> Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Es kommt vor, dass sich die Änderung eines Features darauf auswirkt, wie Git mit CodeCommit zusammen funktioniert.

Wenn Sie Probleme mit einer bestimmten Version von Git und CodeCommit haben, lesen Sie die Informationen in Fehlerbehebung.

Schritt 3: Richten Sie den Credential Helper ein

1. Führen Sie im Terminalfenster den Git-Befehl git config aus. Geben Sie an, dass das Git-Hilfsprogramm für Anmeldeinformationen mit dem AWS-Anmeldeinformationsprofil verwendet werden soll, damit das Git-Hilfsprogramm für Anmeldeinformationen den Pfad an die Repositorys senden kann:

```
git config --global credential.helper '!aws codecommit credential-helper $@'
git config --global credential.UseHttpPath true
```

> **Tip**
>
> Der Credential Helper verwendet den StandardAWSCredential-Profil oder die Amazon EC2-Instance-Rolle. Sie können das zu verwendende Profil vorgeben, z. B. `CodeCommitProfile`, wenn Sie ein AWS-Anmeldeinformationsprofil erstellt haben, das mit CodeCommit verwendet werden soll:
>
> ```
> git config --global credential.helper '!aws --profile CodeCommitProfile
>  codecommit credential-helper $@'
> ```
>
> Falls im Profilnamen Leerzeichen enthalten sind, muss der Name in Anführungszeichen (") gesetzt werden.
> Sie können Profile anstatt global auch pro Repository konfigurieren, indem Sie `--local` anstelle von `--global` verwenden.

Das Git-Hilfsprogramm für Anmeldeinformationen schreibt folgenden Wert in `~/.gitconfig`:

```
[credential]
    helper = !aws --profile CodeCommitProfile codecommit credential-helper $@
    UseHttpPath = true
```

> **Important**
>
> Wenn Sie einen anderen IAM-Benutzer auf demselben lokalen Computer verwenden möchten fürCodeCommit, müssen Sie das ausführengit configBefehl erneut und geben Sie einen anderen anAWSProfil mit Anmeldeinformationen.

2. Führen Sie den Befehl git config --global --edit aus und überprüfen Sie so, ob der oben genannte Wert in `~/.gitconfig` geschrieben wurde. War der Schreibvorgang erfolgreich, wird der Wert angezeigt (zusätzlich zu den Werten, die ggf. bereits in der globalen Git-Konfigurationsdatei vorhanden sind). Zum Beenden geben Sie **`:q`** ein und drücken die Eingabetaste.

   Falls nach der Konfiguration des Hilfsprogramms für Anmeldeinformationen Probleme auftreten sollten, finden Sie weitere Informationen unter Fehlerbehebung.

   > **Important**
   >
   > Wenn Sie macOS verwenden, gehen Sie wie folgt vor, um sicherzustellen, dass der Credential Helper korrekt konfiguriert ist.

3. Wenn Sie macOS verwenden, verwenden Sie HTTPSstelle eine Verbindung zu einem herCodeCommitEndlager. Wenn Sie zum ersten Mal eine HTTPS-Verbindung zu einem CodeCommit-Repository hergestellt haben, ist nach etwa 15 Minuten kein Zugriff mehr möglich. Die Standard-Git-Version auf macOS verwendet das Keychain Access-Hilfsprogramm zum Speichern von Anmeldeinformationen. Aus Sicherheitsgründen wird für den Zugriff auf das CodeCommit-Repository ein temporäres Passwort generiert, daher sind die im Schlüsselbund hinterlegten Anmeldeinformationen nach ca. 15 Minuten nicht mehr gültig. Damit die abgelaufenen Anmeldeinformationen nicht mehr verwendet werden, führen Sie einen der folgenden Schritte aus:

   - Sie installieren eine Git-Version, in der nicht standardmäßig der Schlüsselbund verwendet wird.
   - Sie konfigurieren das Schlüsselbund-Dienstprogramm so, dass keine Anmeldeinformationen für CodeCommit-Repositorys bereitgestellt werden.

   1. Öffnen Sie das Schlüsselbund-Dienstprogramm. (Sie finden das Schlüsselbund-Dienstprogramm mit dem Finder.)

2. Suchen Sie nach `git-codecommit.us-east-2.amazonaws.com`. Markieren Sie die Zeile, öffnen Sie das Kontextmenü oder klicken Sie mit der rechten Maustaste und wählen Sie dann Informationen abrufen aus.
3. Wählen Sie die Registerkarte Access Control aus.
4. Wählen Sie unter Confirm before allowing access den Eintrag `git-credential-osxkeychain` aus und klicken Sie auf das Minuszeichen, um den Eintrag aus der Liste zu entfernen.

> **Note**
>
> Nachdem Sie `git-credential-osxkeychain` aus der Liste entfernt haben, wird bei jeder Ausführung eines Git-Befehls eine Popup-Meldung eingeblendet. Klicken Sie zum Fortfahren auf Deny. Falls Sie diese Dialogfelder umgehen möchten, finden Sie hier andere Optionen:
>
> - Stellen Sie die Verbindung zu CodeCommit mit SSH anstelle von HTTPS her. Weitere Informationen finden Sie unter Für SSH-Verbindungen unter Linux, macOS oder Unix.
> - Wählen Sie im Schlüsselbund-Dienstprogramm auf der Registerkarte Access Control für `git-codecommit.us-east-2.amazonaws.com` die Option Allow all applications to access this item (access to this item is not restricted) aus. Auf diese Weise werden keine Popups angezeigt, aber die Anmeldeinformationen laufen ab (nach durchschnittlich etwa 15 Minuten) und die Fehlermeldung 403 wird angezeigt. In diesem Fall müssen Sie das Schlüsselbundobjekt löschen, um die Funktionalität wiederherzustellen.
> - Weitere Informationen finden Sie unter Git für macOS: Ich habe den Hilfsprogramm für Anmeldeinformationen erfolgreich konfiguriert, aber jetzt wird mir der Zugriff auf mein Repository verweigert (403).

Schritt 4: Stellen Sie eine Verbindung mit dem herCodeCommitkonsole und klone das Repository

Wenn Sie den Namen und die Verbindungsdetails für das CodeCommit-Repository bereits von einem Administrator erhalten haben, können Sie diesen Schritt überspringen und das Repository direkt klonen.

So stellen Sie eine Verbindung mit einem CodeCommit-Repository her

1. Öffne dasCodeCommitKonsole beihttps://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl dieAWS-Regionwo das Repository erstellt wurde. Repositorien sind spezifisch für einAWS-Region. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.

   - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang vonAWS CLI.
   - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
   - Kopieren Sie die SSH-URL, wenn Sie mit Ihrem IAM-Benutzer ein öffentliches/privates SSH-Schlüsselpaar verwenden.

   > **Note**
   >
   > Wenn Sie eine sehenWillkommenSeite statt einer Liste von Repositorien, es gibt keine Repositorys, die mit Ihrem verknüpft sindAWSKonto imAWS-Regionwo Sie angemeldet sind. Informationen zur Erstellung eines Repositorys finden Sie unter the section called “Erstellen eines -Repositorys”, oder befolgen Sie die Schritte im Tutorial Erste Schritte mit Git und CodeCommit.

4. Öffnen Sie ein Terminal und führen Sie den git clone-Befehl mit der kopierten HTTPS-URL aus. Zum Beispiel, um ein Repository mit dem Namen zu klonen*MyDemoRepo*zu einem lokalen Repo mit dem Namen*my-demo-repo*in der Region USA Ost (Ohio):

   ```
   git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
   ```

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Schritten inErste Schritte mit CodeCommit um mit der Nutzung zu beginnenCodeCommit.

Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitRepositorien unter Windows mit demAWS CLIHelfer für Anmeldeinformationen

Bevor Sie AWS CodeCommit verwenden können, müssen Sie die Schritte für die erste Konfiguration ausführen. Für die meisten Benutzer lässt sich das am einfachsten mit den Schritten unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden umsetzen. Wenn Sie jedoch über ein Root-Konto, über einen Verbundzugriff oder mit temporären Anmeldeinformationen eine Verbindung mit CodeCommit herstellen möchten, müssen Sie das AWS CLI-Hilfsprogramm für Anmeldeinformationen nutzen.

> **Note**
>
> Obwohl das Hilfsprogramm für Anmeldeinformationen eine unterstützte Methode für die Verbindung mit CodeCommit über den Verbundzugriff, über einen Identitätsanbieter oder mit temporären Anmeldeinformationen ist, empfiehlt sich die Installation und Verwendung des git-remote-codecommit-Dienstprogramms. Weitere Informationen finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit.

Dieses Thema führt Sie durch die Schritte zur Installation vonAWS CLI, richten Sie Ihren Computer ein undAWSProfil, stellen Sie eine Verbindung zu einem herCodeCommitRepository und klonen Sie dieses Repository auf Ihren Computer. Dies wird auch als Erstellen eines lokalen Repos bezeichnet. Wenn Sie mit Git noch nicht vertraut sind, finden Sie unter Wo kann ich mehr über Git erfahren? genauere Informationen.

Themen

- Schritt 1: Erstkonfiguration fürCodeCommit
- Schritt 2: Installieren von Git
- Schritt 3: Richten Sie den Credential Helper ein

- Schritt 4: Stellen Sie eine Verbindung mit dem herCodeCommitkonsole und klone das Repository
- Nächste Schritte

Schritt 1: Erstkonfiguration fürCodeCommit

Gehen Sie wie folgt vor, um ein Amazon Web Services-Konto einzurichten, einen IAM-Benutzer zu erstellen und zu konfigurieren und denAWS CLI. Die AWS CLI enthält ein Hilfsprogramm für Anmeldeinformationen, das Sie für HTTPS-Verbindungen zu Ihren CodeCommit-Repositorys konfigurieren.

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurierenCodeCommit

1. Erstellen Sie ein Amazon Web Services-Konto, indem Sie zuhttp://aws.amazon.comund wählenMelde dich an.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unterEinen IAM-Benutzer in Ihrem Amazon Web Services-Konto erstellen.

   > **Note**
   >
   > CodeCommit erfordert AWS Key Management Service Wenn Sie einen bestehenden IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die das ausdrücklich verbietenAWS KMSAktionen erforderlich vonCodeCommit. Weitere Informationen finden Sie unter AWS KMS und Verschlüsselung.

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
4. Wählen Sie in der IAM-Konsole im NavigationsbereichBenutzer, und wählen Sie dann den IAM-Benutzer aus, für den Sie konfigurieren möchtenCodeCommitZugriff.
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie in der Richtlinienliste den Eintrag AWSCodeCommitPowerUser oder eine andere verwaltete Richtlinie für den CodeCommit-Zugriff aus. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit.

Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen SieWeiter: Überprüfenum die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen über die verwalteten Richtlinien von CodeCommit und die Freigabe von Repositorys für andere Gruppen und Benutzer finden Sie unter Teilen Sie ein Repository und Authentifizierung und Zugriffskontrolle für AWS CodeCommit.

So installieren und konfigurieren Sie die AWS CLI

1. Laden Sie die AWS CLI auf den lokalen Computer herunter und installieren Sie. Dies ist eine Voraussetzung für das Arbeiten mit CodeCommit über die Befehlszeile. Wir empfehlen die InstallationAWS CLIVersion 2. Es ist die neueste Hauptversion vonAWS CLIund unterstützt die neuesten Funktionen. Es ist die einzige Version vonAWS CLIdas unterstützt die Verwendung eines Root-Kontos, Verbundzugriffs oder temporärer Anmeldeinformationen mitgit-remote-codecommit.

   Weitere Informationen finden Sie unter Erste Schritte mit der AWS-Befehlszeilenschnittstelle.

   > **Note**
   >
   > CodeCommitfunktioniert nur mitAWS CLIVersionen 1.7.38 und höher. Als bewährte Methode installieren oder aktualisieren Sie die AWS CLI auf die neueste verfügbare Version. Führen Sie den Befehl aws --version aus, um zu überprüfen, welche Version der AWS CLI installiert ist.
   > Informationen zum Upgraden von einer älteren auf die aktuelle Version der AWS CLI finden Sie unter Installieren der AWS Command Line Interface.

2. Führen Sie diesen Befehl aus, um zu überprüfen, obCodeCommitBefehle für denAWS CLIsind installiert.

   ```
   aws codecommit help
   ```

   Dieser Befehl gibt eine Liste vonCodeCommitBefehle.

3. Konfigurieren Sie dieAWS CLImit einem Profil unter Verwendung desconfigureBefehl, wie folgt:.

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie denAWSZugriffsschlüssel undAWSgeheimer Zugriffsschlüssel des IAM-Benutzers, mit dem verwendet werden sollCodeCommit. Stellen Sie außerdem sicher, dass Sie Folgendes angebenAWS-Regionwo das Repository existiert, z. B.`us-east-2`. Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

```
AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press
 Enter
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and
 then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then
 press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen für die AWS CLI finden Sie unter:

- Benannte Profile
- Verwenden einer IAM-Rolle in derAWS CLI
- Befehl „Set“
- Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen herzustellenAWS-Region, müssen Sie das neu konfigurierenAWS CLImit dem Standardnamen der Region. Zu den Standardregionsnamen für CodeCommit zählen:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1

- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3
- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und AWS-Region finden Sie unter Regionen und Git-Verbindungsendpunkte. Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unterWie erhalte ich Zugangsdaten?undVerwaltung von Zugriffsschlüsseln für IAM-Benutzer. Weitere Informationen zur AWS CLI und zu Profilen finden Sie unter Benannte Profile.

Schritt 2: Installieren von Git

Um mit Dateien, Commits und anderen Informationen zu arbeiten inCodeCommitRepositorys müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt die Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wieGit für Windows. Wenn du diesen Link verwendest, um Git zu installieren, kannst du alle Standardeinstellungen der Installation akzeptieren, mit Ausnahme der folgenden:

- Wenn Sie während desAnpassung Ihrer PATH-UmgebungSchritt, wählen Sie die Option zur Verwendung von Git von der Befehlszeile aus.
- (Optional) Wenn Sie beabsichtigen, HTTPS mit dem Credential Helper zu verwenden, der imAWS CLIanstatt Git-Anmeldeinformationen zu konfigurieren fürCodeCommit, auf demKonfiguration zusätzlicher OptionenSeite, stellen Sie sicher, dassAktivieren Sie Git Credential ManagerOption ist gelöscht. Der Git Credential Manager ist nur kompatibel mitCodeCommitwenn IAM-Benutzer Git-Anmeldeinformationen konfigurieren. Weitere Informationen erhalten Sie unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden und Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert (403).

> **Note**
>
> Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Es kommt vor, dass sich die Änderung eines Features darauf auswirkt, wie Git mit CodeCommit zusammen funktioniert. Wenn Sie Probleme mit einer bestimmten Version von Git und CodeCommit haben, lesen Sie die Informationen in Fehlerbehebung.

Schritt 3: Richten Sie den Credential Helper ein

Die AWS CLI enthält ein Git-Hilfsprogramm für Anmeldeinformationen, das mit CodeCommit verwendet werden kann. Der Git Credential Helper benötigt eineAWS Anmeldeinformationsprofil, in dem eine Kopie des Profils eines IAM-Benutzers gespeichert wirdAWSZugriffsschlüssel-ID undAWSgeheimer Zugriffsschlüssel (zusammen mit einem Standardschlüssel)AWS-RegionName und Standardausgabeformat). Der Git Credential Helper verwendet diese Informationen, um sich

automatisch zu authentifizieren mitCodeCommitSie müssen diese Informationen also nicht jedes Mal eingeben, wenn Sie Git für die Interaktion mitCodeCommit.

1. Öffne eine Eingabeaufforderung und verwende Git zum Ausführengit config, spezifiziert die Verwendung des Git Credential Helper mit demAWSCredential-Profil, das es dem Git Credential Helper ermöglicht, den Pfad an Repositorys zu senden:

```
git config --global credential.helper "!aws codecommit credential-helper $@"
git config --global credential.UseHttpPath true
```

Das Git-Hilfsprogramm für Anmeldeinformationen schreibt folgende Werte in die .gitconfig-Datei:

```
[credential]
    helper = !aws codecommit credential-helper $@
    UseHttpPath = true
```

> **Important**
>
> - Falls Sie anstelle der Windows-Befehlszeile einen Bash-Emulator einsetzen, müssen Sie einfache (statt doppelte) Anführungszeichen eingeben.
> - Der Credential Helper verwendet den StandardAWSProfil oder die Amazon EC2-Instance-Rolle. Wenn Sie ein eigenes erstelltes AWS-Anmeldeinformationsprofil verwenden möchten (z. B. *CodeCommitProfile*), können Sie den Befehl folgendermaßen anpassen:
>
> ```
> git config --global credential.helper "!aws codecommit credential-helper
>  --profile CodeCommitProfile $@"
> ```
>
> Dann werden folgende Werte in die .gitconfig-Datei geschrieben:
>
> ```
> [credential]
>     helper = !aws codecommit credential-helper --profile=CodeCommitProfile
>  $@
>     UseHttpPath = true
> ```

- Falls Ihr Profilname Leerzeichen enthält, müssen Sie nach der Ausführung dieses Befehls den Namen in der .gitconfig-Datei in einfache Anführungszeichen (') setzen. Andernfalls können Sie das Hilfsprogramm für Anmeldeinformationen nicht nutzen.
- Wenn Sie im Rahmen der Installation von Git für Windows auch das Git-Dienstprogramm zur Verwaltung von Anmeldeinformationen (Git Credential Manager) implementiert haben, werden Ihnen nach den ersten Verbindungsversuchen entweder 403-Fehlermeldungen angezeigt oder Sie werden vom Dienstprogramm aufgefordert, Ihre Anmeldeinformationen einzugeben. Dieses Problem lässt sich am besten lösen, indem Sie Git für Windows deinstallieren und dann ohne den Git Credential Manager erneut installieren, da das Dienstprogramm mit CodeCommit nicht kompatibel ist. Sofern Sie das Git Credential Manager-Dienstprogramm und CodeCommit verwenden möchten, sind weitere Konfigurationsschritte erforderlich. Beispielsweise müssen Sie die .gitconfig-Datei manuell ändern und angeben, dass bei einer Verbindungserstellung zu AWS CodeCommit das CodeCommit-Hilfsprogramm für Anmeldeinformationen verwendet werden soll. Entfernen Sie sämtliche gespeicherten Anmeldeinformationen aus dem Credential Manager-Dienstprogramm (dies finden Sie in der Systemsteuerung). Nachdem Sie alle gespeicherten Anmeldeinformationen gelöscht haben, fügen Sie folgenden Abschnitt zur .gitconfig-Datei hinzu, speichern diese und versuchen, über ein neues Befehlszeilenfenster eine Verbindung herzustellen:

  ```
  [credential "https://git-codecommit.us-east-2.amazonaws.com"]
      helper = !aws codecommit credential-helper $@
      UseHttpPath = true

  [credential "https://git-codecommit.us-east-1.amazonaws.com"]
      helper = !aws codecommit credential-helper $@
      UseHttpPath = true
  ```

  Möglicherweise ist zusätzlich eine Neukonfiguration der Einstellungen für git config erforderlich. Dabei geben Sie --system anstelle von --global oder --local an, damit alle Verbindungen ordnungsgemäß hergestellt werden können.
- Wenn Sie verschiedene IAM-Benutzer auf demselben lokalen Computer verwenden möchten fürCodeCommit, sollten Sie Folgendes angeben git config --localstattgit config --global, und führe die Konfiguration für jeden ausAWSProfil mit Anmeldeinformationen.

2. Führen Sie den Befehl git config --global --edit aus, um zu prüfen, ob die vorigen Werte in die .gitconfig-Datei Ihres Benutzerprofils geschrieben wurden (standardmäßig `%HOME%\.gitconfig` oder *`drive`*`:\Users\`*`UserName`*`\.gitconfig`). War der Schreibvorgang erfolgreich, werden die Werte angezeigt (zusätzlich zu den Werten, die ggf. bereits in der globalen Git-Konfigurationsdatei vorhanden sind). Zum Beenden geben Sie **`:q`** ein und drücken die Eingabetaste.

Schritt 4: Stellen Sie eine Verbindung mit dem herCodeCommitkonsole und klone das Repository

Wenn Sie den Namen und die Verbindungsdetails für das CodeCommit-Repository bereits von einem Administrator erhalten haben, können Sie diesen Schritt überspringen und das Repository direkt klonen.

So stellen Sie eine Verbindung mit einem CodeCommit-Repository her

1. Öffne dasCodeCommitKonsole beihttps://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl dieAWS-Regionwo das Repository erstellt wurde. Repositorien sind spezifisch für einAWS-Region. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.

   - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang vonAWS CLI.
   - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
   - Kopieren Sie die SSH-URL, wenn Sie mit Ihrem IAM-Benutzer ein öffentliches/privates SSH-Schlüsselpaar verwenden.

> **Note**
>
> Wenn Sie eine sehenWillkommenSeite statt einer Liste von Repositorien, es gibt keine Repositorys, die mit Ihrem verknüpft sindAWSKonto imAWS-Regionwo Sie angemeldet sind. Informationen zur Erstellung eines Repositorys finden Sie unter the section called "Erstellen eines -Repositorys", oder befolgen Sie die Schritte im Tutorial Erste Schritte mit Git und CodeCommit.

4. Öffnen Sie eine Eingabeaufforderung und führen Sie dengit cloneBefehl mit der HTTPS-URL, die Sie kopiert haben. Dadurch wird das lokale Repository in einem Unterverzeichnis des Verzeichnisses erstellt, in dem Sie den Befehl ausgeführt haben. Zum Beispiel, um ein Repository mit dem Namen zu klonen*MyDemoRepo*zu einem lokalen Repo mit dem Namen*my-demo-repo*in der Region USA Ost (Ohio):

   ```
   git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
   ```

   Bei einigen Windows-Versionen werden Sie möglicherweise über eine Popup-Meldung aufgefordert, Ihren Benutzernamen und Ihr Passwort einzugeben. Dabei handelt es sich um das integrierte System für die Anmeldeinformationsverwaltung von Windows, dieses ist jedoch nicht mit dem AWS CodeCommit-Hilfsprogramm für Anmeldeinformationen kompatibel. Klicken Sie auf Abbrechen.

Nächste Schritte

Sie haben die Voraussetzungen erfüllt. Folgen Sie den Schritten inErste Schritte mit CodeCommit um mit der Nutzung zu beginnenCodeCommit.

Erste Schritte

Sie lernen am einfachsten mit CodeCommit kennen und befolgen Sie die Schritte unterErste Schritte mit CodeCommit aus. Wenn Sie mit Git und CodeCommit vertraut sind, sollten Sie auch die Schritte im ausführenErste Schritte mit Git und CodeCommitaus. Auf diese Weise lernen Sie kennen und machen sich mit CodeCommit und den Grundlagen vertraut, die Sie bei der Verwendung von Git für Interaktionen mit den CodeCommit -Repositorys benötigen.

Sie können auch das Tutorial in ansehenAnleitung zum Erstellen einer einfachen Pipeline mit CodePipeline und CodeCommitum zu erfahren, wie Sie das -CodeCommit-Repository im Rahmen einer Pipeline für die fortlaufende Lieferung einsetzen.

Die Tutorials in diesem Abschnitt setzen voraus, dass Sie dasVoraussetzungen und Setupeinschließlich:

- Zuweisen von Berechtigungen zum IAM-Benutzer.
- Einrichten der Verwaltung von Anmeldeinformationen für HTTPS- oder SSH-Verbindungen auf dem lokalen Computer, der für dieses Tutorial verwendet wird.
- Konfigurieren von AWS CLI, sofern das Befehlszeilen- oder Terminalfenster für alle Vorgänge (auch die Repository-Erstellung) genutzt werden soll.

Themen

- Erste Schritte mit AWS CodeCommit
- Erste Schritte mit Git und AWS CodeCommit

Erste Schritte mit AWS CodeCommit

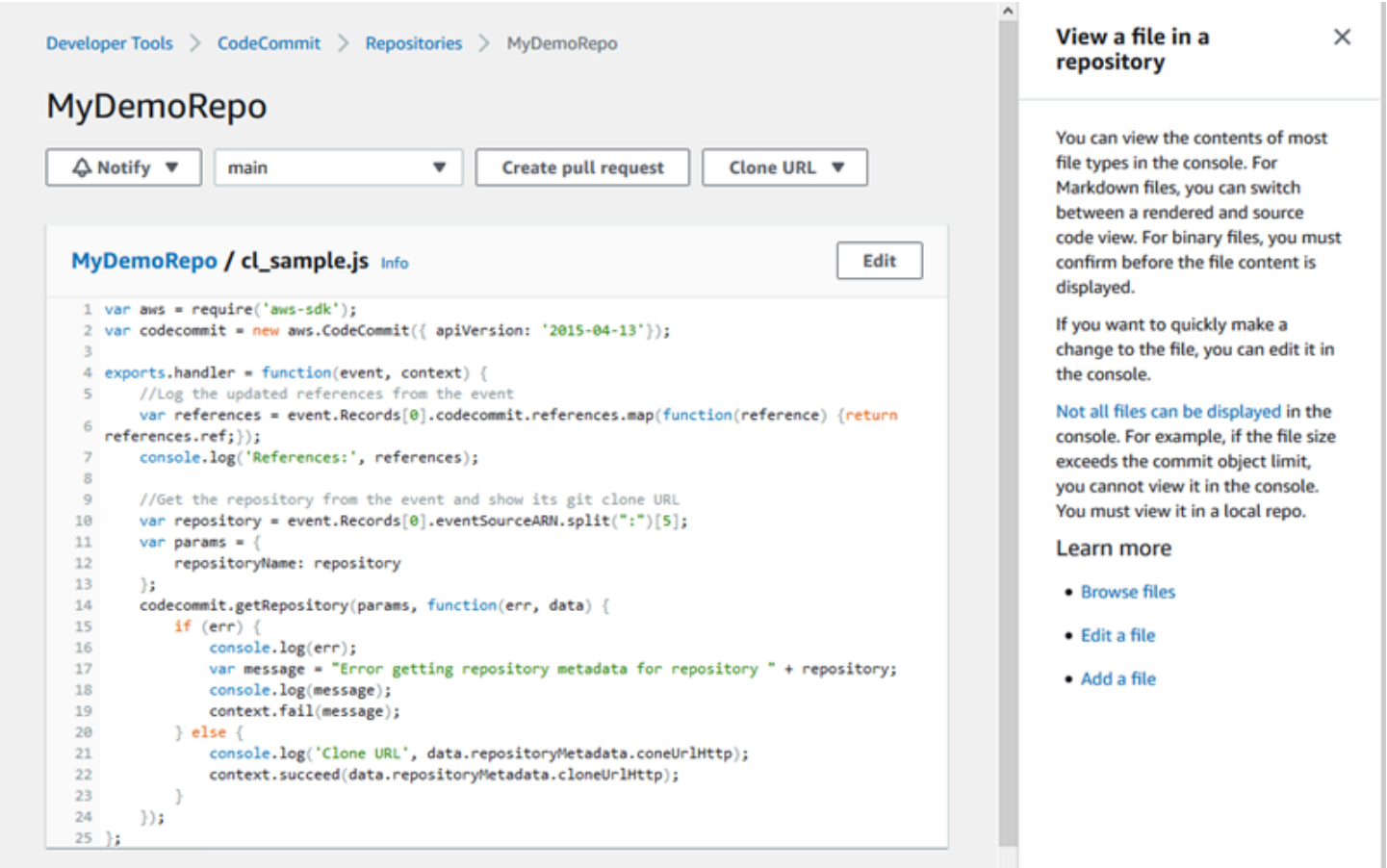
Dieses Tutorial zeigt Ihnen, wie Sie einige wichtige CodeCommit Funktionen verwenden. Zunächst erstellen Sie ein Repository und tragen einige Änderungen ein (Commit). Dann durchsuchen Sie die Dateien und sehen sich die Änderungen an. Sie können auch eine Pull-Anforderung erstellen, damit andere Benutzer die Änderungen an Ihrem Code prüfen und ggf. kommentieren.

Informationen dazu, wie Sie vorhandenen Code migrieren möchten CodeCommit, finden Sie unterMigrieren zu AWS CodeCommit.

Wenn Sie mit Git nicht vertraut sind, sollten Sie auch Erste Schritte mit Git und CodeCommit durcharbeiten. Nachdem Sie diese Tutorials abgeschlossen haben, sollten Sie über ausreichend

Erfahrung verfügen, um damit zu beginnen, sie CodeCommit für Ihre eigenen Projekte und in Teamumgebungen zu verwenden.

Die CodeCommit Konsole enthält hilfreiche Informationen in einem zusammenklappbaren Bereich, den Sie über das Informationssymbol ( ) oder einen beliebigen Informationslink auf der Seite öffnen können. Sie können diesen Bereich jederzeit schließen.



Die CodeCommit Konsole bietet auch eine Möglichkeit, schnell nach Ihren Ressourcen wie Repositorys, Build-Projekten, Bereitstellungsanwendungen und Pipelines zu suchen. Wählen Sie Go to Ressource (Zur Ressource) oder drücken Sie die Taste / und geben Sie dann den Namen der Ressource ein. Alle Übereinstimmungen werden in der Liste angezeigt. Bei der Suche wird nicht zwischen Groß- und Kleinschreibung unterschieden. Sie sehen nur die Ressourcen, für die Sie die Berechtigung besitzen. Weitere Informationen finden Sie unter Anzeigen von Ressourcen in der Konsole.

Voraussetzungen

Bevor Sie beginnen, müssen Sie die Voraussetzungen und das Einrichtungsverfahren abschließen, einschließlich:

- Zuweisen von Berechtigungen für den IAM-Benutzer
- Einrichten der Verwaltung von Anmeldeinformationen für HTTPS- oder SSH-Verbindungen auf dem lokalen Computer, der für dieses Tutorial verwendet wird.
- Konfiguration von, AWS CLI ob Sie die Befehlszeile oder das Terminal für alle Operationen verwenden möchten, einschließlich der Erstellung des Repositorys.

Themen

- Schritt 1: Erstellen eines CodeCommit-Repositorys
- Schritt 2: Fügen Sie Dateien zu Ihrem Repository hinzu
- Schritt 3: Durchsuchen Sie den Inhalt Ihres Repositorys
- Schritt 4: Einen Pull-Request erstellen und gemeinsam bearbeiten
- Schritt 5: Bereinigen
- Schritt 6: Nächste Schritte

Schritt 1: Erstellen eines CodeCommit-Repositorys

Sie können die CodeCommit Konsole verwenden, um ein CodeCommit Repository zu erstellen. Wenn Sie bereits ein Repository haben und für dieses Tutorial verwenden möchten, können Sie diesen Schritt überspringen.

> **Note**
>
> Abhängig von Ihrer Nutzung können Ihnen Gebühren für die Erstellung eines Repositorys oder den Zugriff auf ein Repository in Rechnung gestellt werden. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter Preise.

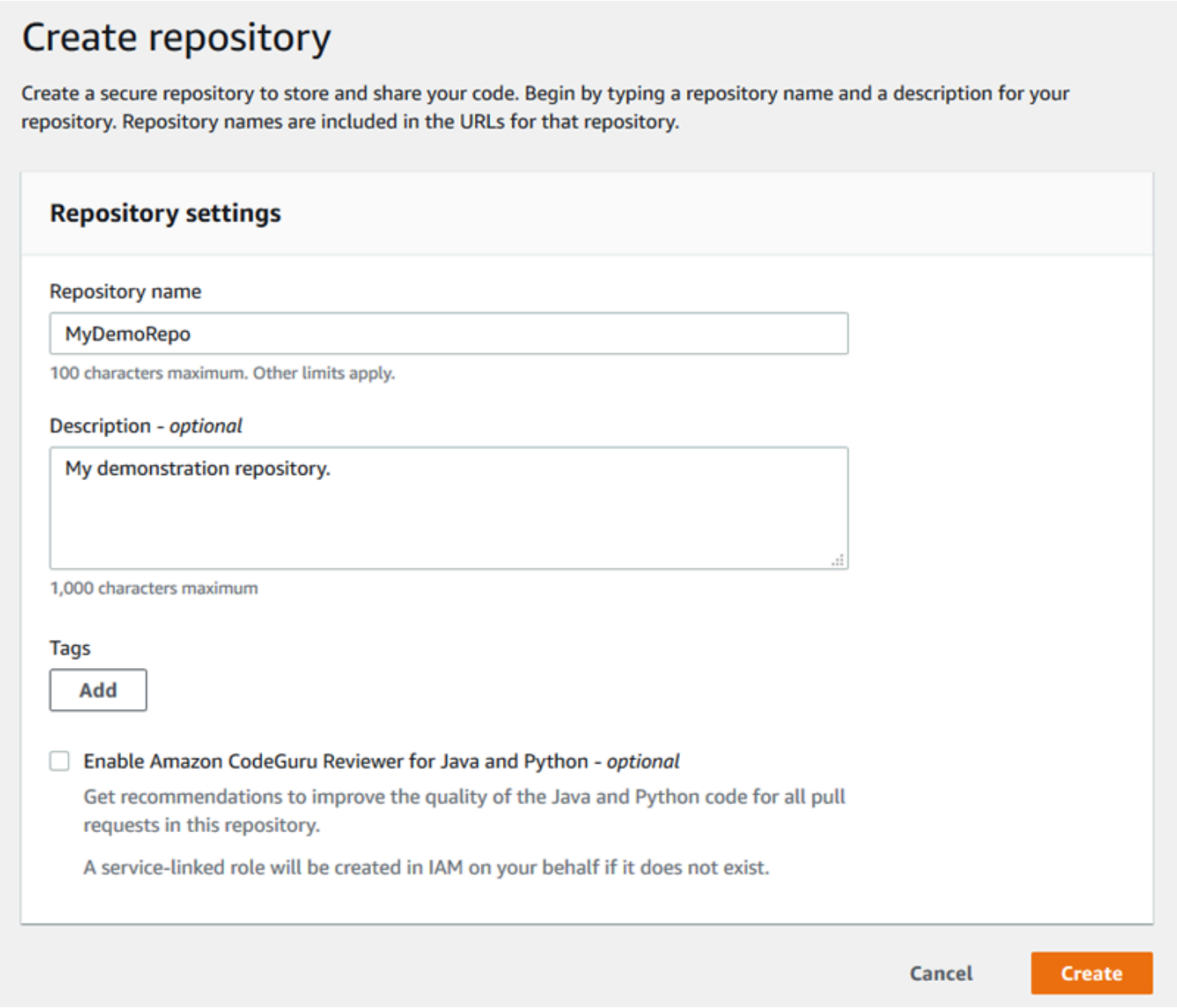
Um das CodeCommit Repository zu erstellen

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.

2. Verwenden Sie die Regionsauswahl, um den AWS-Region Ort auszuwählen, an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Wählen Sie auf der Seite Repositorys die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) unter Repository name (Repository-Name) einen Namen für Ihr Repository ein (z. B. **`MyDemoRepo`**).

   > **Note**
   >
   > Bei Repository-Namen muss die Groß- und Kleinschreibung beachtet werden. Er darf nicht mehr als 100 Zeichen enthalten. Weitere Informationen finden Sie unter Limits.

5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung ein (z. B. **`My demonstration repository`**). Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.
6. (Optional) Wählen Sie Tag hinzufügen, um Ihrem Repository ein oder mehrere Repository-Tags (eine benutzerdefinierte Attributbezeichnung, mit der Sie Ihre AWS Ressourcen organisieren und verwalten können) hinzuzufügen. Weitere Informationen finden Sie unter Kennzeichnen von Repositorys in AWS CodeCommit.
7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen, vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln der Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorys.
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer diesen Code analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um Codefehler zu finden und automatisch Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.
9. Wählen Sie Erstellen.



> **Note**
>
> Wenn Sie einen anderen Namen als `MyDemoRepo` für das Repository verwenden, muss dieser auch in den weiteren Schritten des Tutorials verwendet werden.

Wenn das Repository geöffnet wird, werden Informationen darüber angezeigt, wie Sie Dateien direkt von der CodeCommit Konsole aus hinzufügen können.

Schritt 2: Fügen Sie Dateien zu Ihrem Repository hinzu

Sie können Dateien wie folgt zu Ihrem Repository hinzufügen:

- Eine Datei in der CodeCommit Konsole erstellen. Wenn Sie die erste Datei für ein Repository in der Konsole erstellen, wird für Sie ein Branch mit dem Namen main erstellt. Dieser Branch ist der Standard-Branch für dein Repository.
- Eine Datei mithilfe der CodeCommit Konsole von Ihrem lokalen Computer hochladen. Wenn Sie die erste Datei für ein Repository von der Konsole hochladen, wird für Sie ein Zweig mit dem Namen main erstellt. Dieser Branch ist der Standard-Branch für dein Repository.
- Verwenden Sie einen Git-Client, um das Repository auf Ihren lokalen Computer zu klonen, und fügen Sie dann Dateien hinzu, übertragen Sie sie und übertragen Sie sie in das CodeCommit Repository. Im Rahmen dieses ersten Commits von Git wird für dich ein Branch erstellt, der als Standard-Branch für dein Repository festgelegt ist. Der Name des Branches ist die Standardauswahl Ihres Git-Clients. Erwägen Sie, Ihren Git-Client so zu konfigurieren, dass er main als Namen für den ersten Branch verwendet.

> **Note**
>
> Du kannst jederzeit Branches erstellen und den Standard-Branch für ein Repository ändern. Weitere Informationen finden Sie unter Mit Branches in AWS CodeCommit Repositorys arbeiten.

Der einfachste Weg, um loszulegen, besteht darin, die CodeCommit Konsole zu öffnen und eine Datei hinzuzufügen. Auf diese Weise erstellen Sie auch einen Standardzweig für Ihr Repository mit dem Namen main. Anweisungen zum Hinzufügen einer Datei und zum Erstellen eines ersten Commits zu einem Repository mit dem AWS CLI finden Sie unter Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI.

So fügen Sie dem Repository eine Datei hinzu

1. Wählen Sie in der Navigationsleiste des Repositorys die Option Code aus.
2. Wählen Sie Add file (Datei hinzufügen) aus. Erstellen Sie dann eine Datei oder laden Sie eine Datei vom Computer hoch. Dieses Tutorial erläutert beide Verfahren.
3. Gehen Sie folgendermaßen vor, um eine Datei hinzuzufügen:

a. Wählen Sie in der Dropdown-Liste der Branches den Branch aus, dem die Datei hinzugefügt werden soll. Der Standard-Branch wird automatisch für Sie ausgewählt. In dem hier gezeigten Beispiel heißt der Standard-Branch `main`. Wenn Sie die Datei einem anderen Branch hinzufügen möchten, wählen Sie einen anderen Branch aus.

b. Geben Sie in das Feld File name (Dateiname) einen Namen für die Datei ein. Geben Sie den Code für die Datei in den Code-Editor ein.

c. Geben Sie im Feld Author name (Name des Autors) den Namen ein, der anderen Repository-Benutzern angezeigt werden soll.

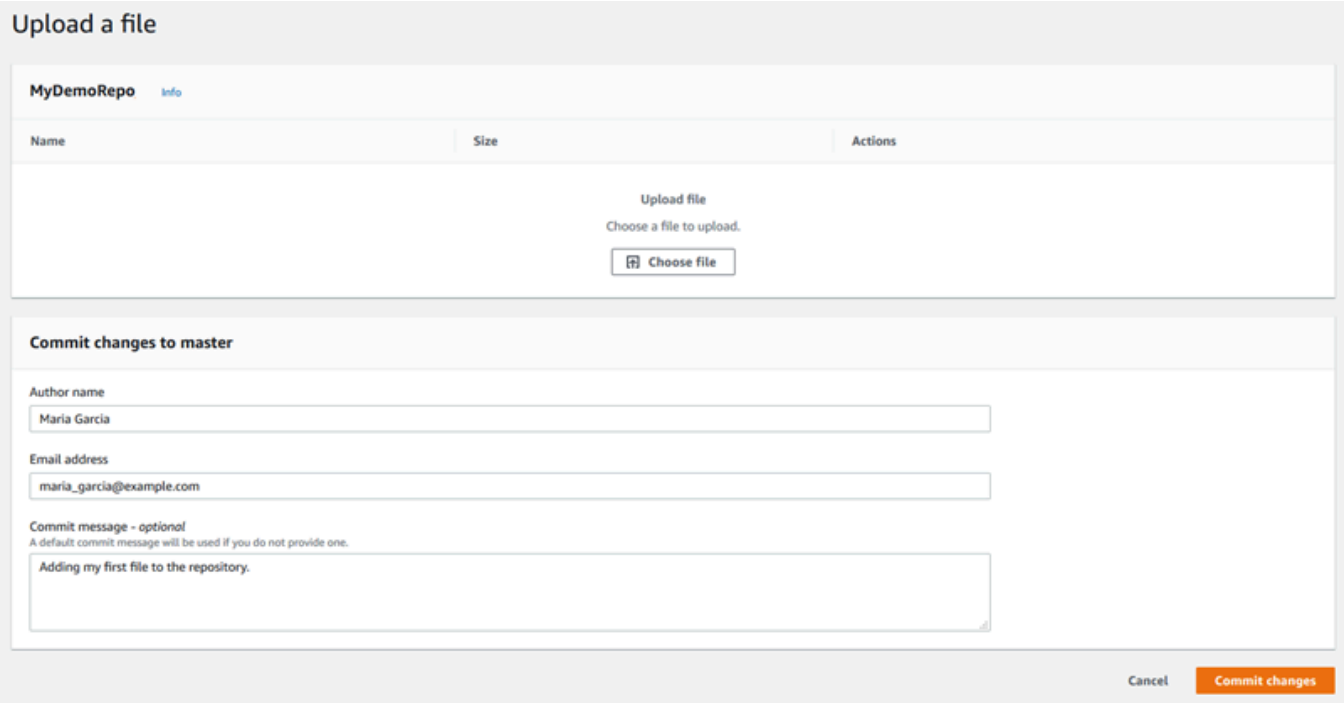
d. Geben Sie eine E-Mail-Adresse in das Feld Email address (E-Mail-Adresse) ein.

e. (Optional) Geben Sie in das Feld Commit message (Commit-Nachricht) eine kurze Nachricht ein. Dies ist zwar optional, wir empfehlen jedoch, eine Commit-Nachricht hinzuzufügen, da dies dazu beiträgt, dass Ihre Teammitglieder verstehen, warum Sie diese Datei hinzugefügt haben. Wenn Sie keine Commit-Nachricht eingeben, wird eine Standardnachricht verwendet.

f. Wählen Sie Commit changes (Änderungen übernehmen) aus.

Gehen Sie folgendermaßen vor, um eine Datei hochzuladen:

- Wenn Sie eine Datei hochladen möchten, wählen Sie die gewünschte Datei aus.



- Geben Sie im Feld Author name (Name des Autors) den Namen ein, der anderen Repository-Benutzern angezeigt werden soll.
- Geben Sie eine E-Mail-Adresse in das Feld Email address (E-Mail-Adresse) ein.
- (Optional) Geben Sie in das Feld Commit message (Commit-Nachricht) eine kurze Nachricht ein. Dies ist zwar optional, wir empfehlen jedoch, eine Commit-Nachricht hinzuzufügen, da dies dazu beiträgt, dass Ihre Teammitglieder verstehen, warum Sie diese Datei hinzugefügt haben. Wenn Sie keine Commit-Nachricht eingeben, wird eine Standardnachricht verwendet.
- Wählen Sie Commit changes (Änderungen übernehmen) aus.

Weitere Informationen finden Sie unter Arbeiten mit Dateien inAWS CodeCommitRepositories.

Um einen Git-Client zum Klonen des Repositorys zu verwenden, installieren Sie Git auf Ihrem lokalen Computer und klonen Sie dann das CodeCommit Repository. Fügen Sie dem lokalen Repo einige Dateien hinzu und übertragen Sie sie in das CodeCommit Repository. Eine umfassende Einführung finden Sie unter Erste Schritte mit Git und CodeCommit. Wenn du mit Git vertraut bist, dir aber nicht sicher bist, wie du das mit einem CodeCommit Repository bewerkstelligen kannst, findest du Beispiele und Anleitungen in Erstellen Sie einen CommitSchritt 2: Erstellen Sie ein lokales Repo, oderHerstellen einer Verbindung mit einem Repository.

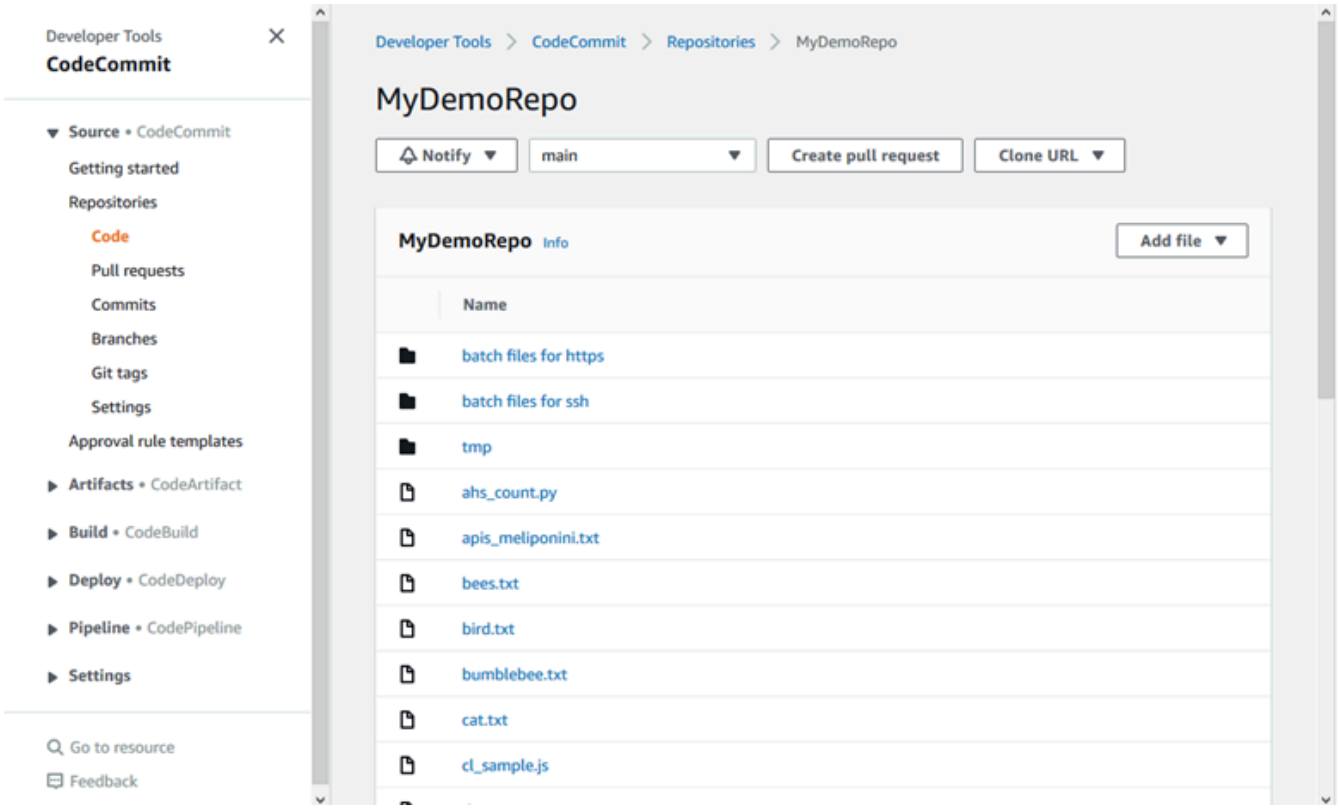
Nachdem Sie dem CodeCommit Repository einige Dateien hinzugefügt haben, können Sie sie in der Konsole ansehen.

Schritt 3: Durchsuchen Sie den Inhalt Ihres Repositorys

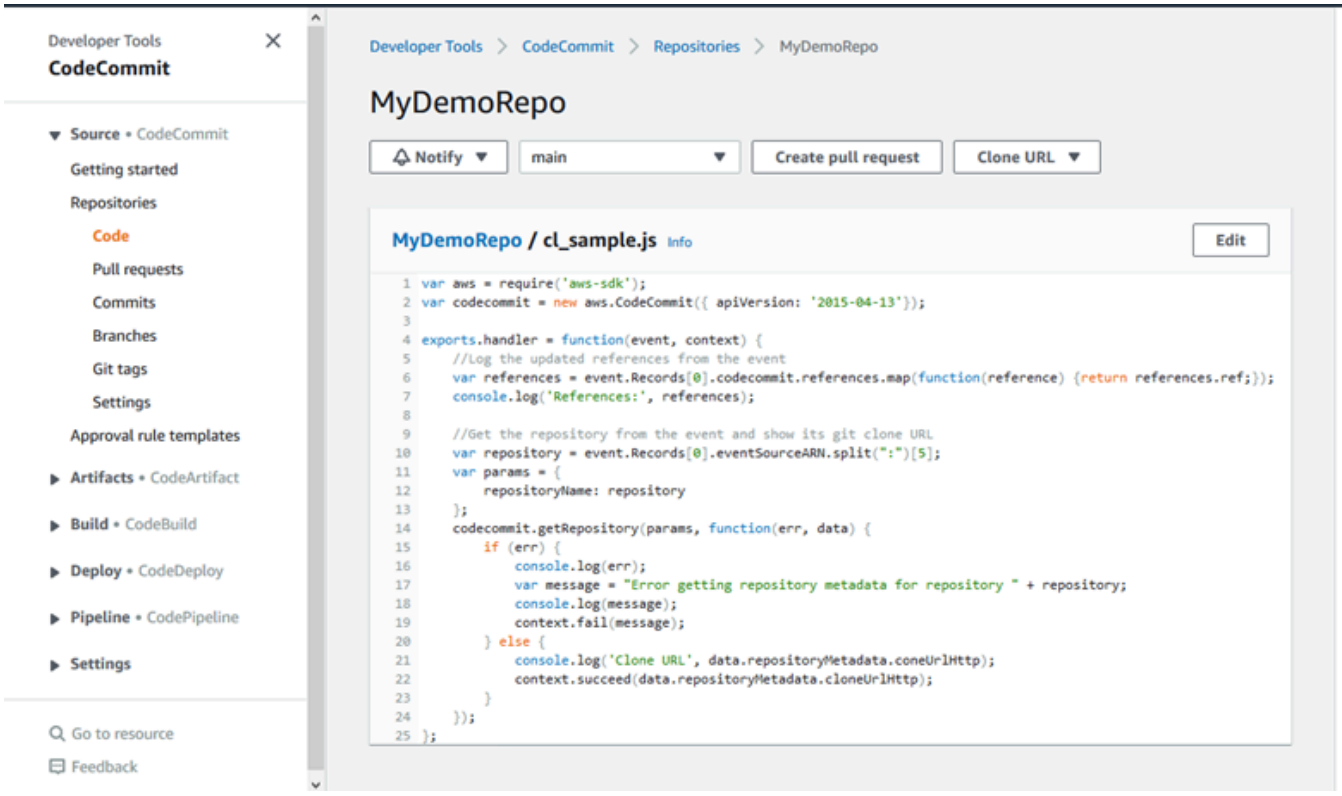
Sie können die CodeCommit Konsole verwenden, um die Dateien in einem Repository zu überprüfen oder den Inhalt einer Datei schnell zu lesen. So können Sie feststellen, welchen Branch Sie sich genauer ansehen oder ob Sie eine lokale Kopie eines Repositorys erstellen sollten.

So durchsuchen Sie das Repository

1. Wählen Sie unter Repositorys die Option MyDemoRepo.
2. Die Seite zeigt den Inhalt im Standard-Branch des Repositorys an. Um einen anderen Branch oder den Code an einem bestimmten Tag anzuzeigen, wählen Sie den Branch oder das Tag in der Liste aus. Im folgenden Screenshot ist die Ansicht auf den Hauptzweig eingestellt.

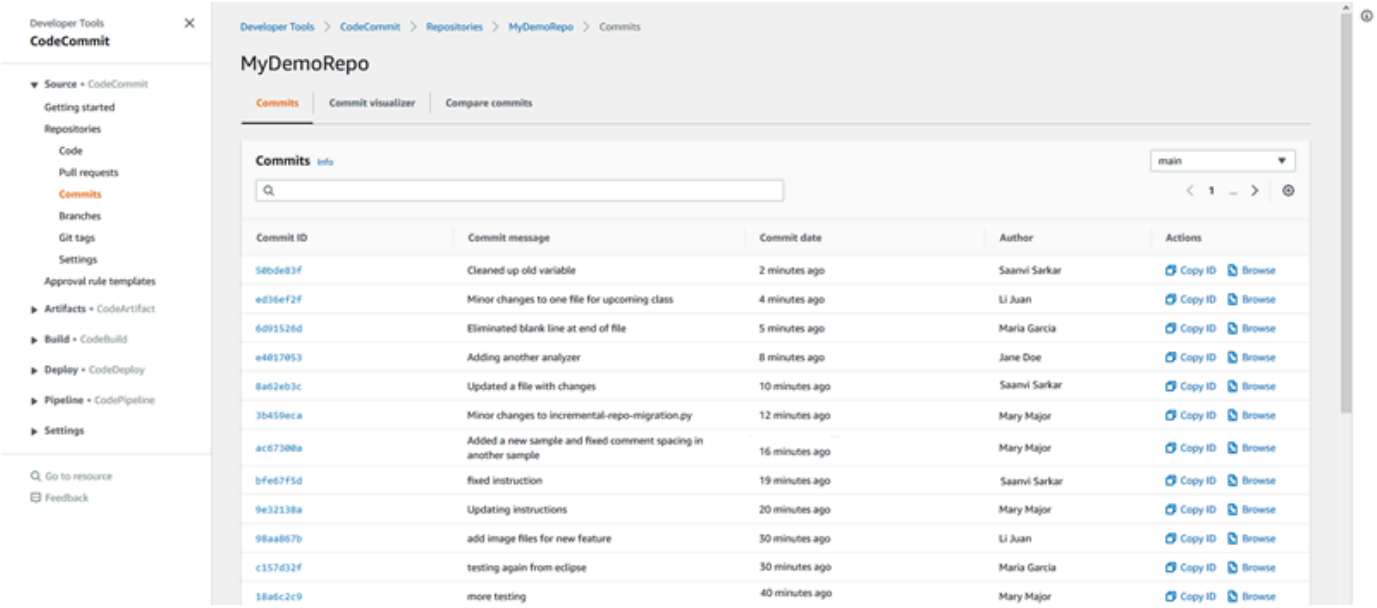


3. Um den Inhalt einer Datei in Ihrem Repository anzuzeigen, wählen Sie die Datei in der Liste aus. Wählen Sie das Einstellungssymbol aus, um die Farbe des angezeigten Codes zu ändern.



Weitere Informationen finden Sie unter Durchsuchen Sie Dateien in einem Repository.

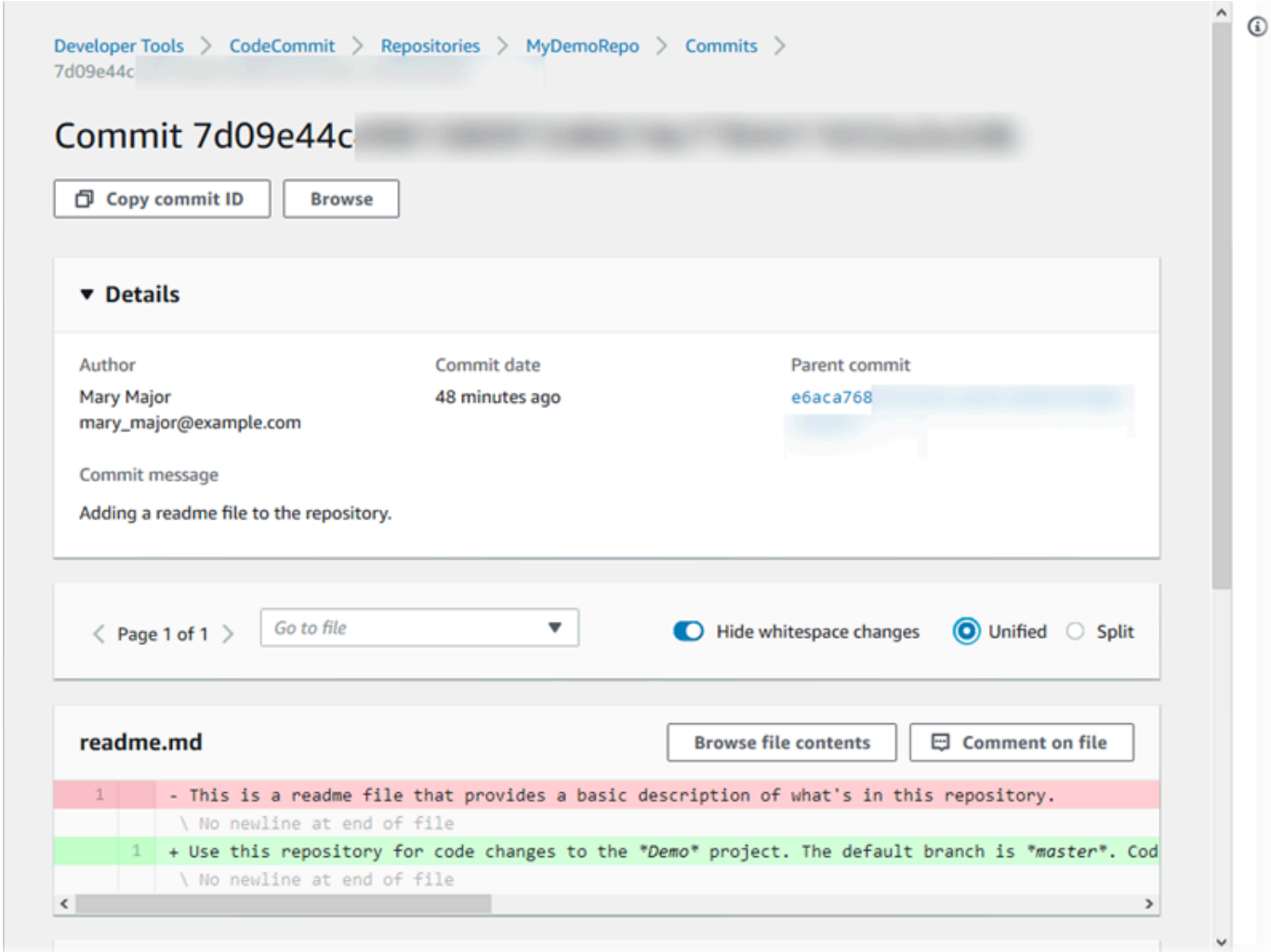
4. Um den Commit-Verlauf des Repositorys zu durchsuchen, wählen Sie Commits. Die Konsole zeigt den Commit-Verlauf für den Standard-Branch in umgekehrter chronologischer Reihenfolge an. Prüfen Sie die Commit-Details anhand von Autor, Datum usw.



5. Um den Commit-Verlauf nach Branch oder Git-Tag anzuzeigen, wählen Sie den Branch oder das Tag, den bzw. das Sie anzeigen möchten, in der Liste aus.

6. Wählen Sie die gekürzte Commit-ID aus, um die Unterschiede zwischen einem Commit und seinem übergeordneten Commit anzuzeigen. Sie können wählen, wie die Änderungen angezeigt werden sollen, z. B. ob Leerzeichenänderungen ein- oder ausgeblendet werden sollen und ob Änderungen inline (Ansicht Unified (Vereinheitlicht) oder nebeneinander (Ansicht Split (Geteilt)) angezeigt werden sollen.

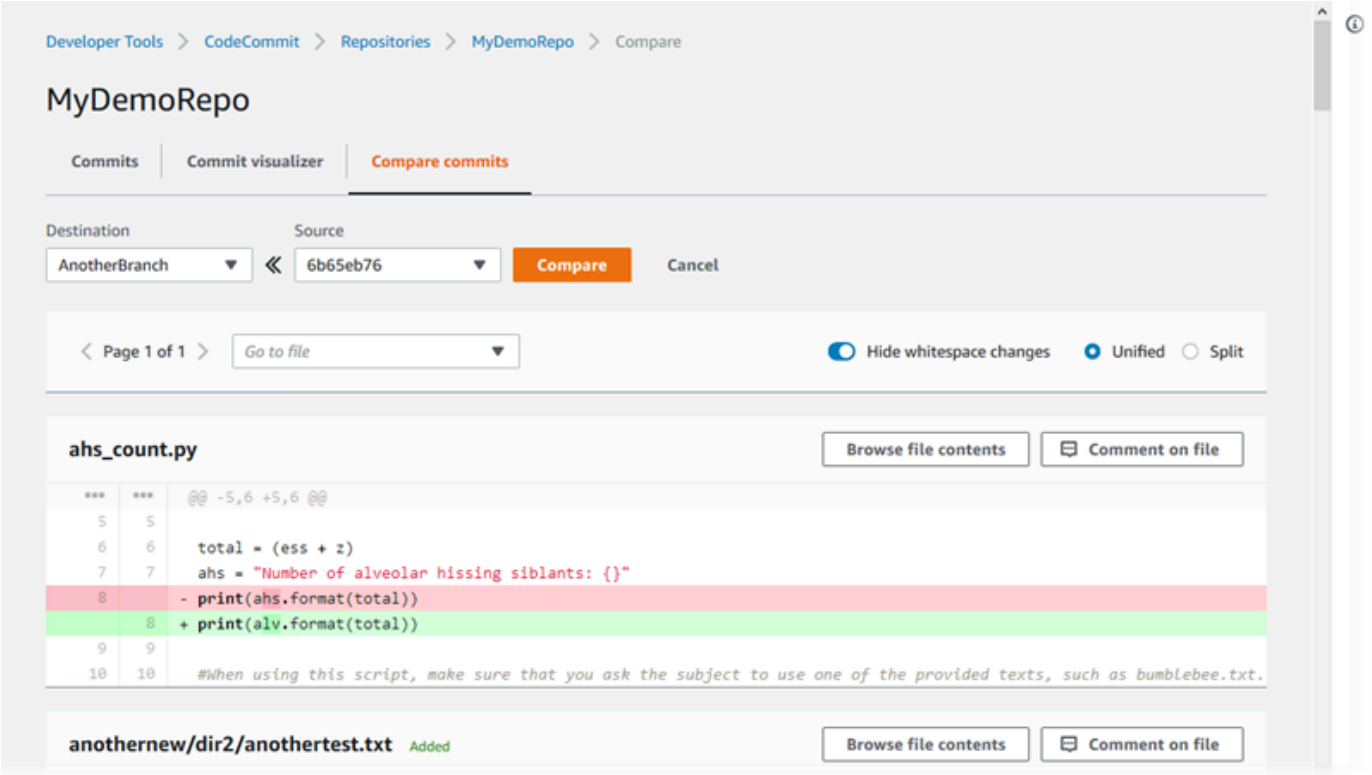
> **Note**
>
> Ihre Präferenzen für die Anzeige des Codes sowie andere Konsoleneinstellungen werden als Browser-Cookies gespeichert, wenn Sie sie ändern. Weitere Informationen finden Sie unter Arbeiten mit Benutzereinstellungen.



7. Um alle Kommentare zu einem Commit anzuzeigen, wählen Sie den Commi aus und blättern Sie durch die Änderungen, um sie inline anzuzeigen. Sie können auch Ihre eigenen Kommentare hinzufügen und auf Kommentare anderer Benutzer antworten.

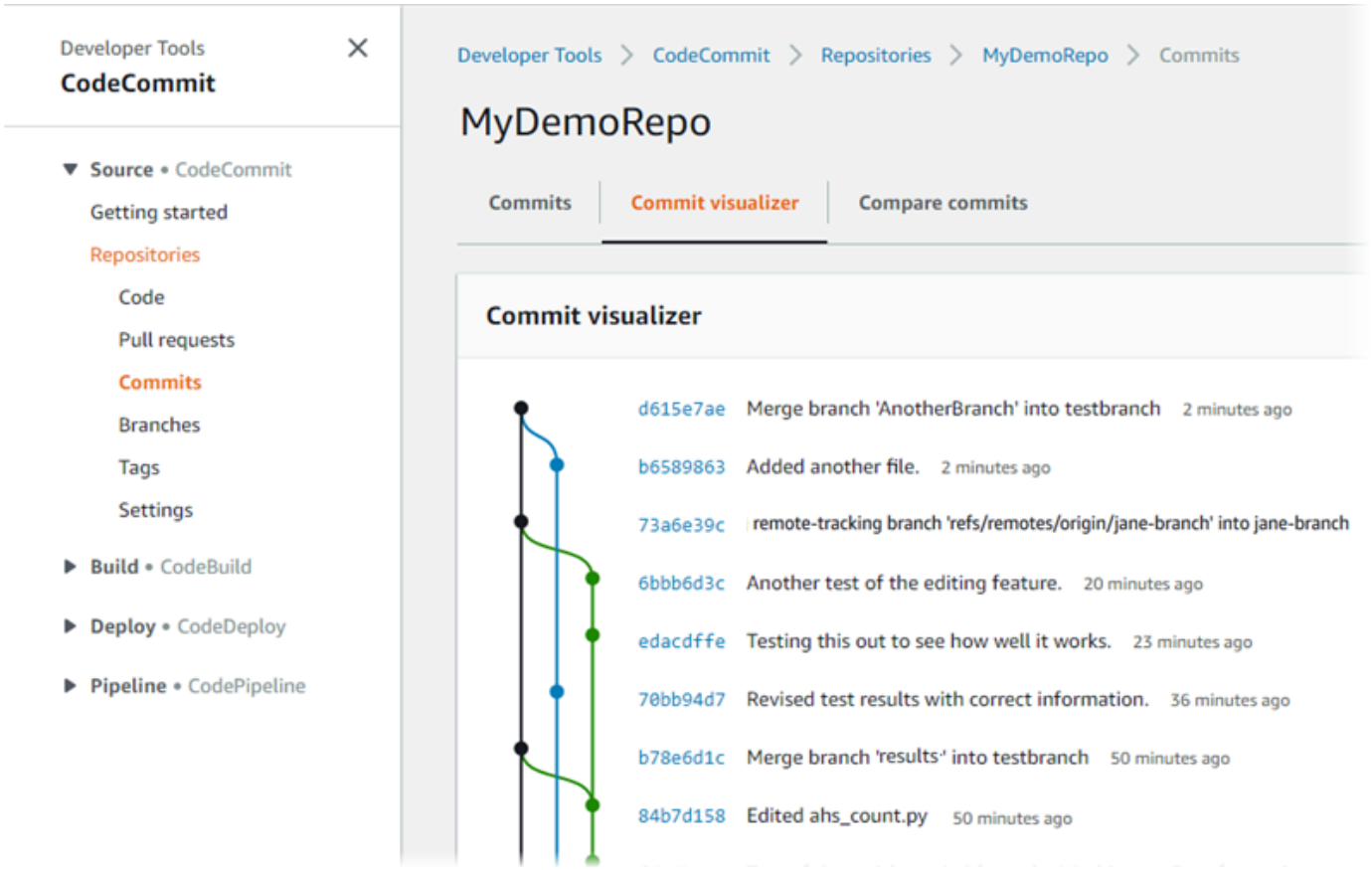
   Weitere Informationen finden Sie unter Kommentar zu einem Commit.

8. Um die Unterschiede zwischen zwei beliebigen Commit-Spezifizierern (einschließlich Branches, Tags und Commit-IDs) anzuzeigen, wählen Sie im Navigationsbereich Commits und dann Compare commits (Commits vergleichen) aus.



Weitere Informationen finden Sie unter Den Commit-Verlauf eines Repositorys durchsuchen und Vergleichen von Commits.

9. Wählen Sie unter Commits die Commit visualizer-Registerkarte aus.



Die Commit-Grafik wird angezeigt. Neben dem jeweiligen Punkt wird die Betreffzeile der einzelnen Commits in der Grafik aufgeführt. Die Anzeige der Betreffzeile ist auf 80 Zeichen beschränkt.

10. Um weitere Details zu einem Commit anzuzeigen, wählen Sie die abgekürzte Commit-ID aus. Um das Diagramm eines bestimmten Commits zu rendern, wählen Sie diesen Punkt im Diagramm aus. Weitere Informationen finden Sie unter Sehen Sie sich ein Diagramm des Commit-Verlaufs eines Repositorys an .

Schritt 4: Einen Pull-Request erstellen und gemeinsam bearbeiten

Wenn Sie mit anderen Benutzern in einem Repository arbeiten, können Sie Code gemeinsam bearbeiten und Änderungen prüfen. Sie können eine Pull-Anforderung erstellen, damit andere Benutzer die Codeänderungen in einem Branch prüfen und kommentieren können. Sie können auch einzelne oder mehrere Genehmigungsregeln für die Pull-Anforderung erstellen. Sie können beispielsweise eine Genehmigungsregel erstellen, die voraussetzt, dass mindestens zwei andere Benutzer die Pull-Anforderung genehmigen, bevor die Zusammenführung erfolgen kann. Nachdem die Pull-Anforderung genehmigt wurde, können Sie diese Änderungen im Ziel-Branch

zusammenführen. Wenn Sie Benachrichtigungen für Ihr Repository einrichten, können Repository-Benutzer E-Mails zu Repository-Ereignissen erhalten, z. B. für Pull-Anforderungen oder wenn jemand Kommentare zu Code hinterlässt. Weitere Informationen finden Sie unter Benachrichtigungen für Ereignisse in einem Repository konfigurieren AWS CodeCommit.

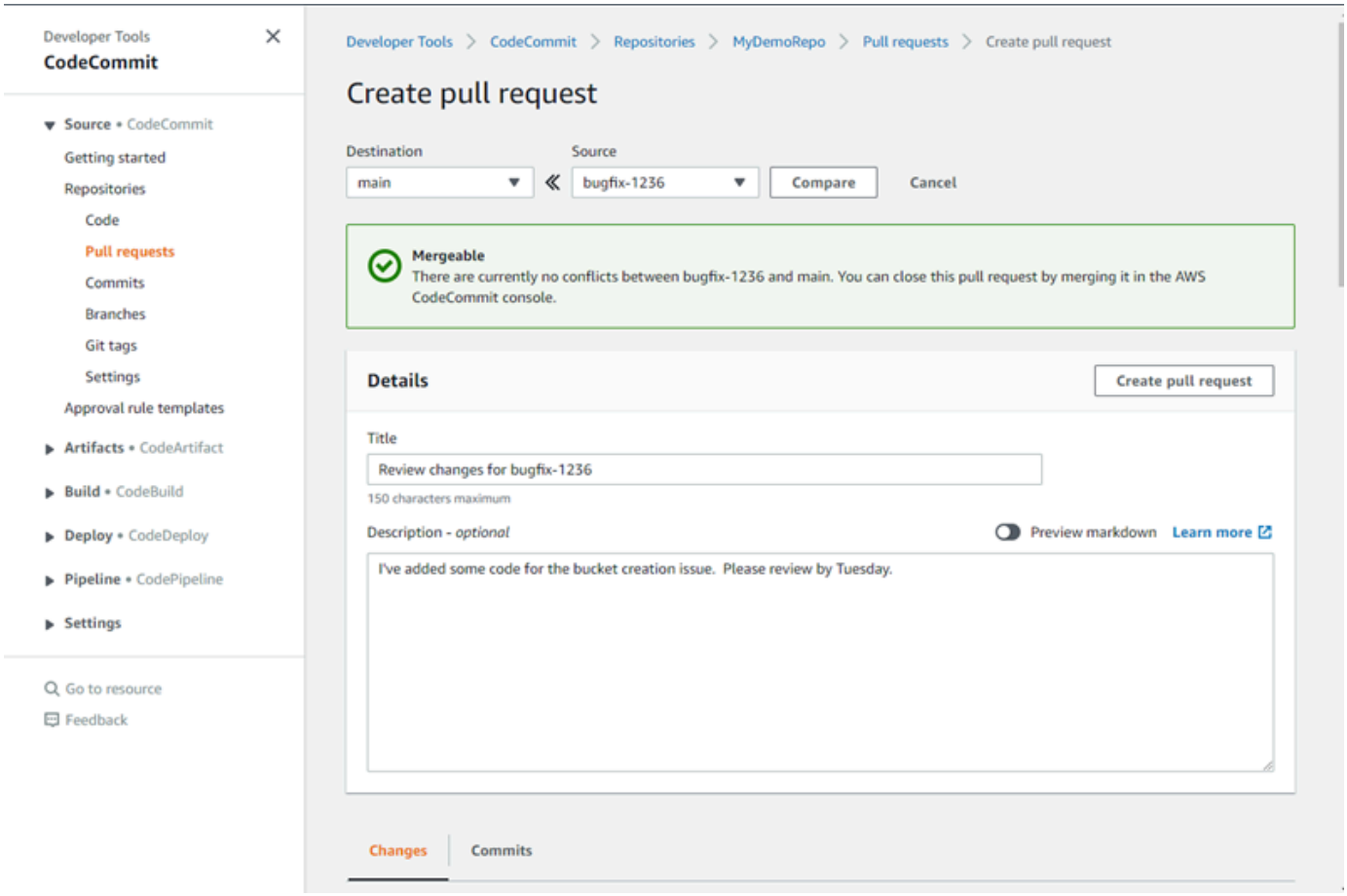
> **Important**
>
> Bevor Sie eine Pull-Anforderung erstellen können, müssen Sie einen Branch erstellen, der die Codeänderungen enthält, die Sie überprüfen möchten. Weitere Informationen finden Sie unter Erstellen eines Zweigs.

So erstellen Sie eine Pull-Anforderung und arbeiten gemeinsam daran

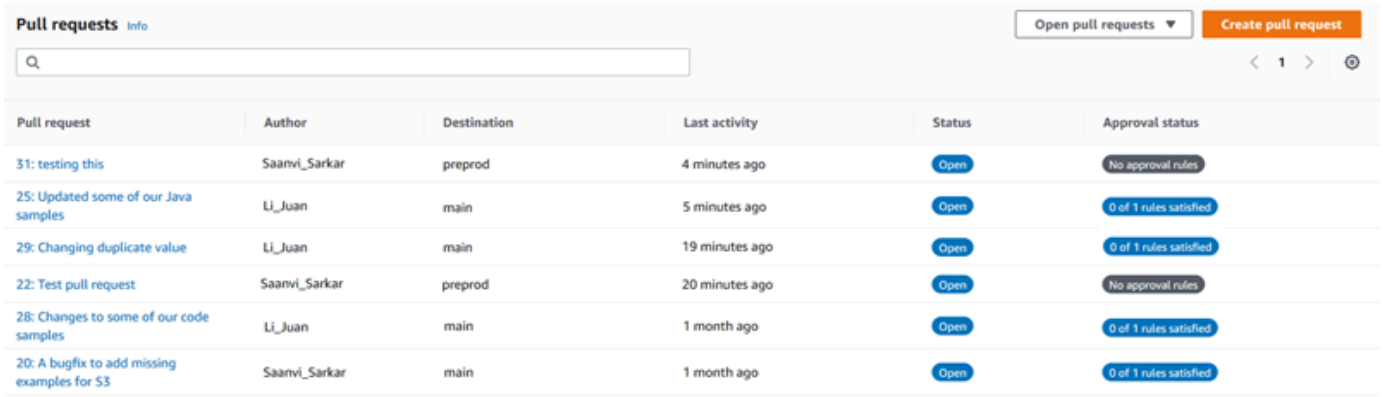
1. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.
2. Wählen Sie unter Pull request (Pull-Anforderung) die Option Create pull request (Pull-Anforderung erstellen) aus.

   > **Tip**
   >
   > Darüber hinaus können Sie Pull-Anforderungen über Branches und Code erstellen.

   Wählen Sie in Create pull request unter Source den Branch aus, der die zu prüfenden Änderungen enthält. Wählen Sie unter Destination (Ziel) den Branch aus, in dem der geprüfte Code nach dem Schließen der Pull-Anforderung zusammengeführt werden soll. Wählen Sie Compare aus.
3. Prüfen Sie die Details der Zusammenführung und die Änderungen, um sicherzustellen, dass die Pull-Anforderung die Änderungen und Commits enthält, die geprüft werden sollen. Wenn dies der Fall ist, geben Sie unter Title (Titel) einen Titel für diese Prüfung an. Dieser Titel wird in der Liste der Pull-Anforderungen für das Repository angezeigt. Unter Description (Beschreibung) können Sie eingeben, wofür diese Prüfung vorgesehen ist, sowie weitere nützliche Informationen für Prüfer hinterlassen. Wählen Sie Erstellen.

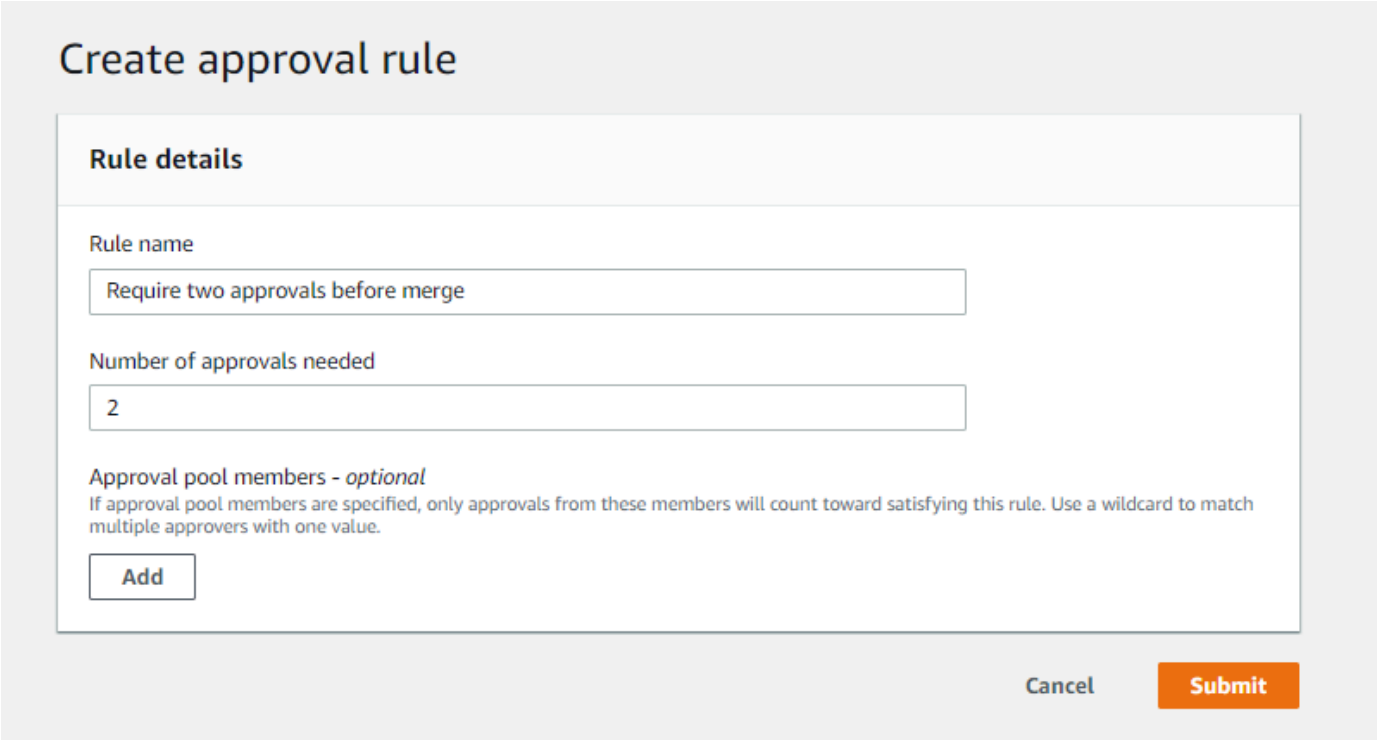


4. Ihre Pull-Anforderung wird in der Liste der Pull-Anforderungen für das Repository angezeigt. Sie können die Ansicht filtern, sodass nur offene Anforderungen, geschlossene Anforderungen, von Ihnen erstellte Anforderungen usw. angezeigt werden.

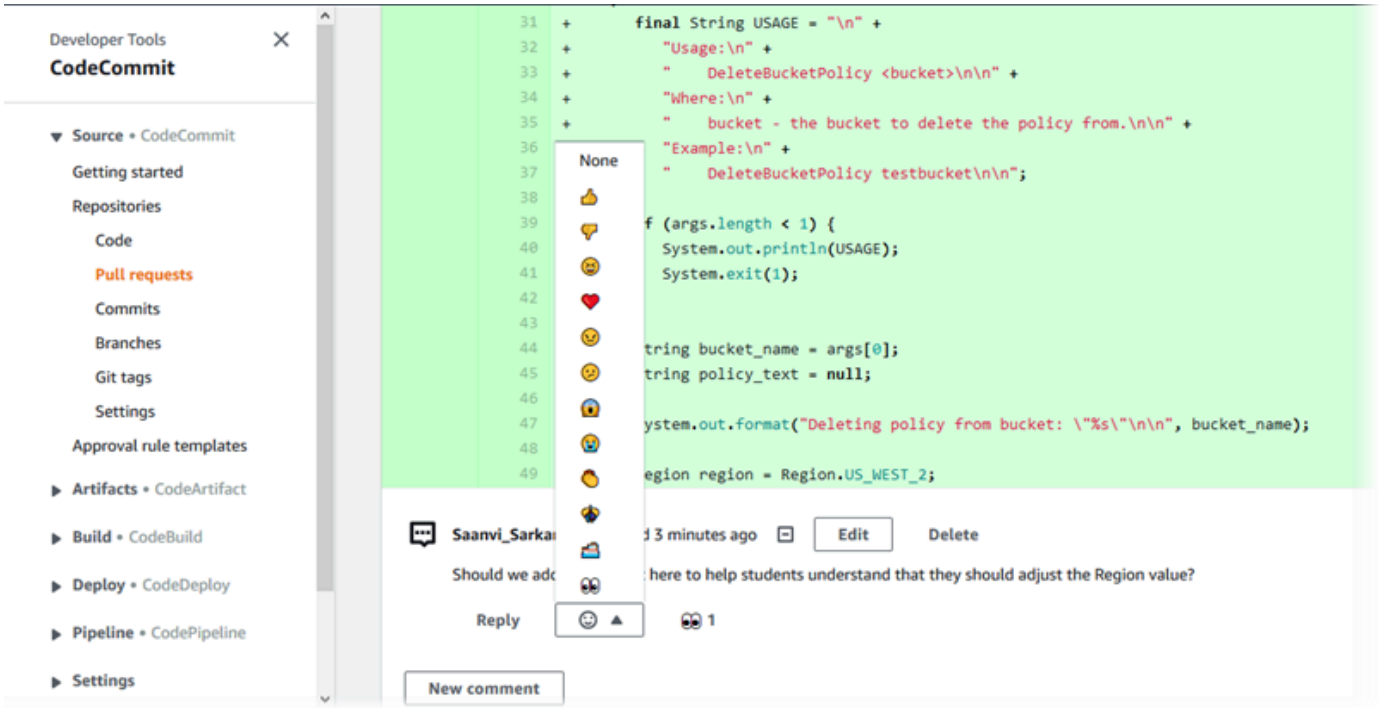


5. Du kannst deinem Pull Request eine Genehmigungsregel hinzufügen, um sicherzustellen, dass bestimmte Bedingungen erfüllt sind, bevor er zusammengeführt werden kann. Um der Pull-Anforderung eine Genehmigungsregel hinzuzufügen, wählen Sie die Pull-Anforderung in der Liste aus. Wählen Sie auf der Registerkarte Approvals (Genehmigungen) die Option Create approval rule (Genehmigungsregel erstellen) aus.

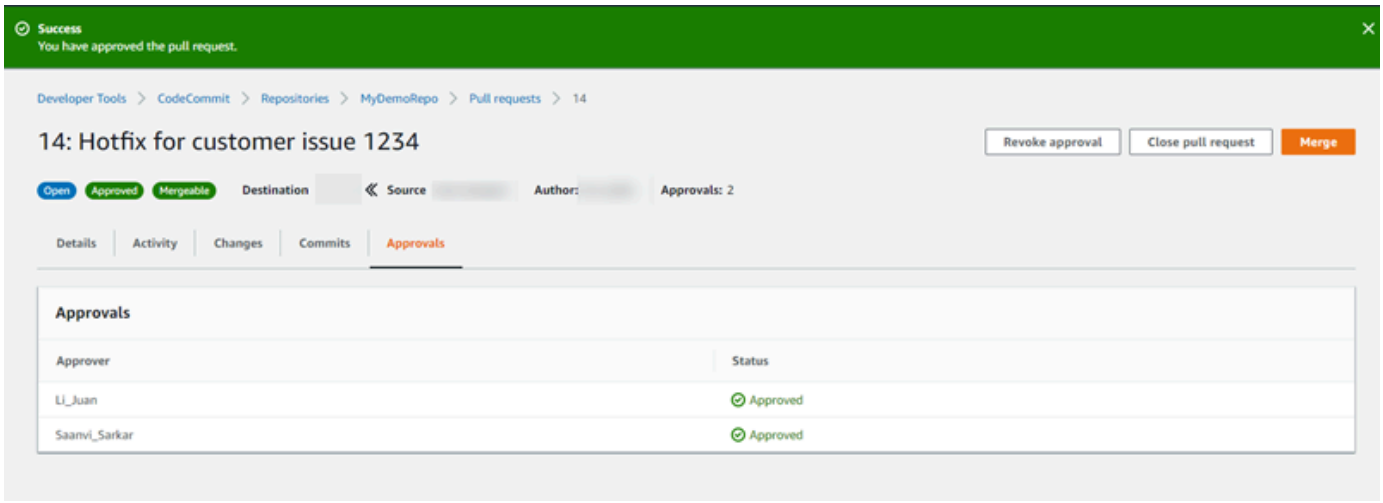
6. Geben Sie in das Feld Rule name (Regelname) einen beschreibenden Namen für die Regel ein. Wenn Sie z. B. möchten, dass eine Pull-Anforderung von zwei Personen genehmigt werden muss, bevor sie zusammengeführt werden kann, können Sie die Regel beispielsweise **`Require two approvals before merge`** nennen. Geben Sie in das Feld Number of approvals needed (Anzahl erforderlicher Genehmigungen) den gewünschten Wert (**2**) ein. Der Standardwert ist 1. Wählen Sie Absenden aus. Weitere Informationen zu Genehmigungsregeln und Genehmigungs-Pool-Mitgliedern finden Sie unter Erstellen einer Genehmigungsregel für eine Pull-Anforderung.



7. Wenn Sie Benachrichtigungen für Ihr Repository konfiguriert haben und Benutzer über Pull-Anforderungereignisse benachrichtigen wollen, erhalten die Benutzer eine E-Mail über Ihre neue Pull-Anforderung. Benutzer können die Änderungen für bestimmte Codezeilen, Dateien und die Pull-Anforderung selbst sehen. Sie können auch auf Kommentare mit Text und Emojis antworten. Falls erforderlich, können Sie Änderungen an den Pull-Anforderung-Branch senden, der die Pull-Anforderung aktualisiert.



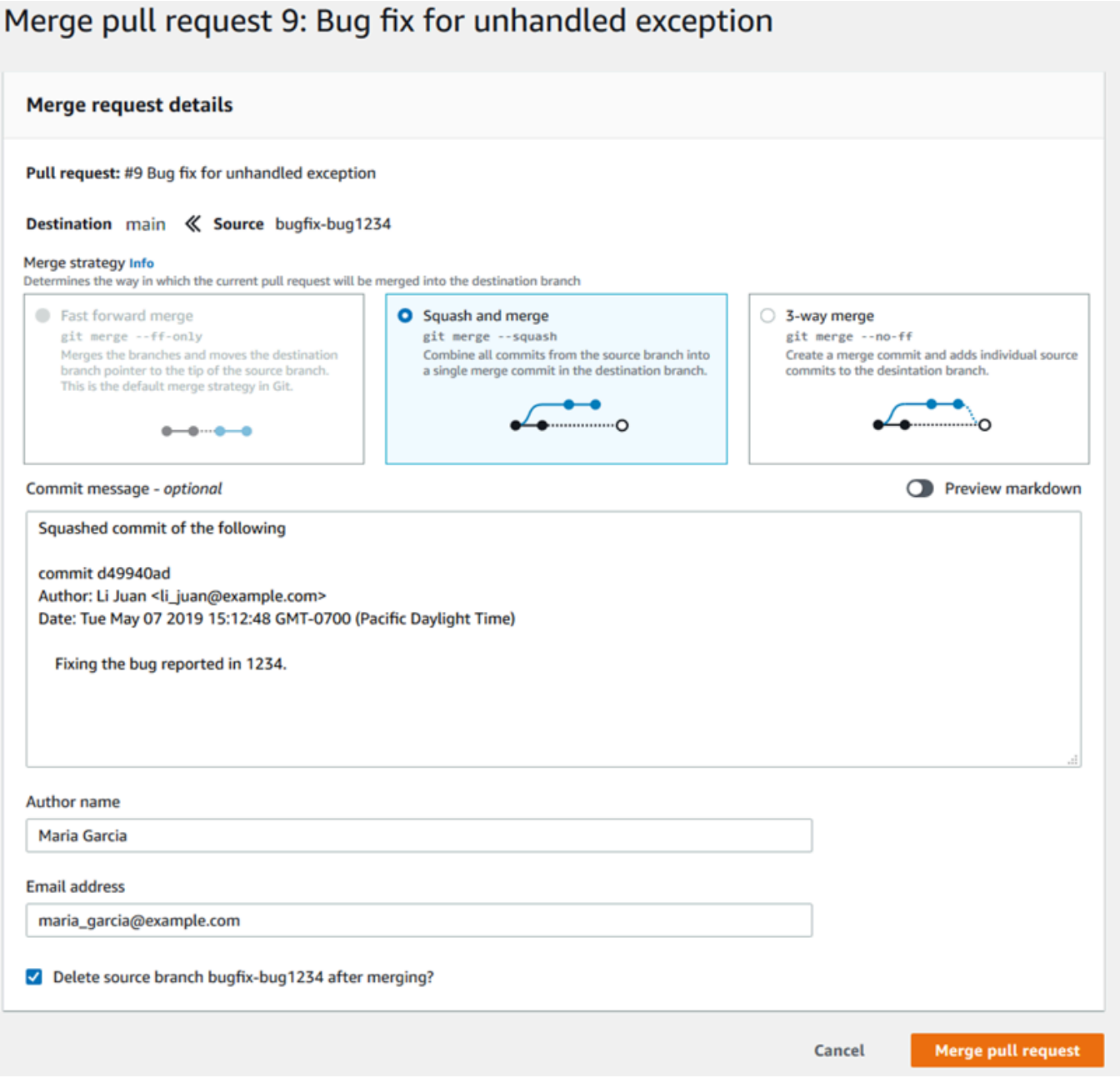
8. Wenn Sie mit den Änderungen in der Anforderung zufrieden sind, wählen Sie Approve (Genehmigen). Sie können eine Pull-Anforderung auch dann genehmigen, wenn für diese Pull-Anforderung keine Genehmigungsregeln konfiguriert sind. Dies belegt, dass Sie die Pull-Anforderung geprüft haben und die Änderungen genehmigen. Sie können Ihre Genehmigung auch widerrufen, wenn Sie Ihre Meinung ändern.



> **Note**
>
> Sie können eine Pull-Anforderung nicht genehmigen, wenn Sie sie erstellt haben.

9. Wenn alle Codeänderungen geprüft wurden und Sie mit diesen Änderungen einverstanden sind, führen Sie von der Pull-Anforderung einen der folgenden Schritte aus:

   - Wenn Sie die Pull-Anforderung schließen möchten, ohne Branches zusammenzuführen, wählen Sie Close pull request (Pull-Anforderung schließen).
   - Wenn Sie die Branches zusammenführen und die Pull-Anforderung schließen möchten, wählen Sie Merge (Zusammenführen) aus. Sie können unter den für Ihren Code verfügbaren Zusammenführungsstrategien wählen. Dabei spielen die Unterschiede zwischen den Quell- und Ziel-Branches eine Rolle und ob der Quell-Branch automatisch gelöscht werden soll, sobald die Zusammenführung abgeschlossen ist. Wenn Sie Ihre Auswahl getroffen haben, wählen Sie Merge pull request (Pull-Anforderung zusammenführen) aus, um die Zusammenführung abzuschließen.



- Wenn es in den Branches Merge-Konflikte gibt, die nicht automatisch gelöst werden können, kannst du sie in der CodeCommit Konsole lösen, oder du kannst deinen lokalen Git-Client verwenden, um die Branches zusammenzuführen und dann die Zusammenführung zu pushen. Weitere Informationen finden Sie unter Lösen von Konflikten in einer Pull-Anforderung in einemAWS CodeCommitEndlager.

> **Note**
>
> Sie können Branches jederzeit manuell zusammenführen, einschließlich von Pull-Anforderungs-Branches, indem Sie den Befehl git mergein Ihrem lokalen Repository ausführen und Ihre Änderungen per Push senden.

Weitere Informationen finden Sie unter Verwenden von Pull-Anforderungen und Arbeiten mit Genehmigungsregelvorlagen.

Schritt 5: Bereinigen

Wenn Sie das CodeCommit Repository nicht mehr benötigen, sollten Sie das CodeCommit Repository und andere Ressourcen, die Sie in dieser Übung verwendet haben, löschen, damit Ihnen der Speicherplatz nicht weiter in Rechnung gestellt wird.

> **Important**
>
> Diese Aktion kann nicht rückgängig gemacht werden. Nachdem Sie dieses Repository gelöscht haben, können Sie es nicht mehr in ein lokales oder gemeinsam genutztes Repository klonen. Sie können auch keine Daten mehr aus einem lokalen oder gemeinsam genutzten Repo abrufen oder Daten dorthin übertragen oder Git-Operationen ausführen.
> Wenn Sie Benachrichtigungen für Ihr Repository konfiguriert haben, wird beim Löschen des Repositorys auch die für das Repository erstellte Amazon CloudWatch Events-Regel gelöscht. Das Amazon SNS SNS-Thema, das als Ziel für diese Regel verwendet wurde, wird nicht gelöscht.
> Wenn Sie Trigger für Ihr Repository konfiguriert haben, werden beim Löschen des Repositorys nicht die Amazon SNS SNS-Themen oder Lambda-Funktionen gelöscht, die Sie als Ziele dieser Trigger konfiguriert haben. Achten Sie darauf, diese Ressourcen zu löschen, wenn sie nicht benötigt werden. Weitere Informationen finden Sie unter Trigger aus einem Repository löschen.

Um das Repository zu löschen CodeCommit

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.

2. Wählen Sie unter Repositories (Repositorys) das Repositorys aus, das Sie löschen möchten. Wenn Sie die Benennungskonvention in diesem Thema befolgt haben, lautet der Name MyDemoRepo.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Klicken Sie auf der Seite Settings unter Delete repository auf Delete repository.
5. Geben Sie **`delete`** ein und wählen Sie dann Delete (Löschen) aus. Das Repository wird dauerhaft gelöscht.

Schritt 6: Nächste Schritte

Nachdem Sie sich mit CodeCommit einigen ihrer Funktionen vertraut gemacht haben, sollten Sie Folgendes in Betracht ziehen:

- Wenn du neu bei Git bist und/oder CodeCommit Beispiele für die Verwendung von Git mit lesen möchtest CodeCommit, fahre mit dem Erste Schritte mit Git und CodeCommit Tutorial fort.
- Wenn Sie mit anderen in einem CodeCommit Repository arbeiten möchten, finden Sie weitere Informationen unterTeilen Sie ein Repository. (Wenn Sie Ihr Repository mit Benutzern in einem anderen Amazon Web Services Services-Konto teilen möchten, finden Sie weitere Informationen unterKonfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen.)
- Wenn Sie ein Repository zu migrieren möchten CodeCommit, folgen Sie den Schritten unterMigration zu CodeCommit.
- Wenn Sie das Repository einer kontinuierlichen Bereitstellungs-Pipeline hinzufügen möchten, führen Sie die schrittweise Anleitung zum Erstellen einer einfachen Pipeline aus.
- Weitere Informationen zu Produkten und Services, die sich integrieren lassen CodeCommit, sowie Beispiele aus der Community finden Sie unterProdukt- und Service-Integrationen.

Erste Schritte mit Git und AWS CodeCommit

Wenn du neu bei Git und bist CodeCommit, hilft dir dieses Tutorial dabei, einige einfache Befehle zu erlernen, um dir den Einstieg zu erleichtern. Wenn Sie bereits mit Git vertraut sind, können Sie dieses Tutorial überspringen und mit Erste Schritte mit CodeCommit fortfahren.

In diesem Tutorial erstellst du ein Repository, das eine lokale Kopie des CodeCommit Repositorys darstellt, das wir als lokales Repo bezeichnen.

Nachdem Sie das lokale Repo erstellt haben, nehmen Sie einige Änderungen daran vor. Dann senden (pushen) Sie Ihre Änderungen an das CodeCommit Repository.

Sie simulieren auch eine Teamumgebung, in der zwei Benutzer unabhängig voneinander Änderungen in ihr lokales Repository übernehmen und diese Änderungen in das CodeCommit Repository übertragen. Die Benutzer ziehen dann die Änderungen aus dem CodeCommit Repository in ihr eigenes lokales Repository, um die Änderungen zu sehen, die der andere Benutzer vorgenommen hat.

Außerdem erstellen Sie Branches und Tags und verwalten einige Zugriffsberechtigungen im CodeCommit Repository.

Wenn Sie dieses Tutorial abgeschlossen haben, sollten Sie ausreichend Übung im Umgang mit den grundlegenden Git- und CodeCommit -Konzepten haben, um sie für Ihre eigenen Projekte anzuwenden.

Sorgen Sie dafür, dass die Voraussetzungen erfüllt sind und die Einrichtung abgeschlossen ist, einschließlich:

- Weisen Sie dem IAM-Benutzer Berechtigungen zu.
- Richten Sie CodeCommit die Verbindung zu einem Repository über HTTPS, SSH oder ein. git-remote-codecommit Weitere Informationen zu diesen Optionen finden Sie unter Einrichten für AWS CodeCommit.
- Konfigurieren Sie die AWS CLI, sofern das Befehlszeilen- oder Terminalfenster für alle Operationen (auch die Repository-Erstellung) genutzt werden soll.

Themen

- Schritt 1: Erstellen Sie ein Repository CodeCommit
- Schritt 2: Erstellen Sie ein lokales Repo
- Schritt 3: Erstelle deinen ersten Commit
- Schritt 4: Push deinen ersten Commit
- Schritt 5: Teile das CodeCommit Repository und führe einen weiteren Commit per Push und Pull durch
- Schritt 6: Einen Branch erstellen und teilen
- Schritt 7: Erstelle und teile ein Tag
- Schritt 8: Zugriffsberechtigungen einrichten

- Schritt 9: Bereinigen

Schritt 1: Erstellen Sie ein Repository CodeCommit

In diesem Schritt verwenden Sie die CodeCommit Konsole, um das Repository zu erstellen.

Sie können diesen Schritt überspringen, wenn Sie bereits über ein CodeCommit Repository verfügen, das Sie verwenden möchten.

> **Note**
>
> Abhängig von Ihrer Nutzung können Ihnen Gebühren für die Erstellung eines Repositorys oder den Zugriff auf ein Repository in Rechnung gestellt werden. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter Preise.

Um das CodeCommit Repository zu erstellen

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Verwenden Sie die Regionsauswahl, um den AWS-Region Ort auszuwählen, an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Wählen Sie auf der Seite Repositorys die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) unter Repository name (Repository-Name) einen Namen für Ihr Repository ein (z. B. **`MyDemoRepo`**).

   > **Note**
   >
   > Bei Repository-Namen muss die Groß- und Kleinschreibung beachtet werden. Er darf nicht mehr als 100 Zeichen enthalten. Weitere Informationen finden Sie unter Limits.

5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung ein (z. B. **`My demonstration repository`**). Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.
6. (Optional) Wählen Sie Add tag (Tag hinzufügen) aus, um ein oder mehrere Repository-Tags (ein benutzerdefiniertes Attribut-Label, mit dem Sie Ihre AWS-Ressourcen organisieren

und verwalten), zu Ihrem Repository hinzuzufügen. Weitere Informationen finden Sie unter Kennzeichnen von Repositorys in AWS CodeCommit.

7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen, vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln der Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorys.
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer diesen Code analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um Codefehler zu finden und automatisch Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.
9. Wählen Sie Erstellen aus.

> **Note**
>
> Die verbleibenden Schritte in diesem Tutorial werden `MyDemoRepo` für den Namen des CodeCommit Repositorys verwendet. Wenn Sie einen anderen Namen auswählen, müssen Sie diesen im gesamten Tutorial verwenden.

Weitere Informationen zum Erstellen von Repositorys, darunter auch die Erstellung eines Repositorys vom Terminal oder über die Befehlszeile, finden Sie unter Erstellen eines -Repositorys.

Schritt 2: Erstellen Sie ein lokales Repo

In diesem Schritt richten Sie ein lokales Repository auf Ihrem lokalen Computer ein, um eine Verbindung zu Ihrem Repository herzustellen. Dazu wählen Sie ein Verzeichnis auf Ihrem lokalen Computer aus, das das lokale Repository darstellt. Sie verwenden Git, um eine Kopie Ihres leeren CodeCommit Repositorys in diesem Verzeichnis zu klonen und zu initialisieren. Anschließend geben Sie den Git-Benutzernamen und die E-Mail-Adresse an, die Sie zum Kommentieren Ihrer Commits verwendet haben.

1. Öffne die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositorys sind spezifisch für ein. AWS-Region Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Suchen Sie das Repository, zu dem Sie eine Verbindung herstellen möchten, in der Liste und wählen Sie es aus. Wählen Sie Clone URL (URL klonen) und dann das Protokoll aus, das Sie beim Klonen oder bei der Verbindung zu dem Repository verwenden möchten. Dadurch wird die Klon-URL kopiert.

   - Kopieren Sie die HTTPS-URL, wenn Sie entweder Git-Anmeldeinformationen mit Ihrem IAM-Benutzer oder den Credential Helper verwenden, der im Lieferumfang von enthalten ist. AWS CLI
   - Kopieren Sie die HTTPS-URL (GRC), wenn Sie den Befehl git-remote-codecommit auf Ihrem lokalen Computer verwenden.
   - Kopieren Sie die SSH-URL, wenn Sie ein öffentliches/privates SSH-Schlüsselpaar mit Ihrem IAM-Benutzer verwenden.

   > **Note**
   >
   > Wenn Sie statt einer Liste von Repositorys eine Willkommensseite sehen, sind in dem Land, in dem Sie angemeldet sind, keine Repositorys mit Ihrem AWS Konto verknüpft. AWS-Region Informationen zur Erstellung eines Repositorys finden Sie unter the section called “Erstellen eines -Repositorys”, oder befolgen Sie die Schritte im Tutorial Erste Schritte mit Git und CodeCommit.

4. (Optional) Wir empfehlen, dass du deinen lokalen Git-Client so konfigurierst, dass er **`main`** als Namen für den Standard-Branch für dein Repository verwendet wird. Dies ist der Name, der in allen Beispielen in diesem Handbuch für den Standard-Branch verwendet wird. Es ist auch derselbe Standard-Branch-Name, den Sie CodeCommit verwenden, wenn Sie Ihren ersten Commit in der Konsole durchführen. Führen Sie den folgenden Befehl aus, um den Standard-Branch-Namen global für Ihr System zu konfigurieren:

   ```
   git config --global init.defaultBranch main
   ```

Wenn Sie es vorziehen, einen anderen Standard-Branch-Namen für all Ihre Repositorys zu verwenden, **`main`** ersetzen Sie ihn durch Ihren bevorzugten Namen. In diesem Tutorial wird davon ausgegangen, dass Ihr Standard-Branch den Namen main trägt.

Wenn Sie unterschiedliche Standard-Branch-Namen für verschiedene Repositorys verwenden möchten, können Sie dieses Attribut lokal (--local) statt global (--global) setzen.

5. Klonen Sie das Repository am Terminal oder in der Befehlszeile mit dem git clone Befehl und geben Sie die Klon-URL ein, die Sie in Schritt 3 kopiert haben. Ihre Klon-URL hängt davon ab, welches Protokoll und welche Konfiguration Sie verwenden. Wenn Sie beispielsweise HTTPS mit Git-Anmeldeinformationen verwenden, um ein Repository zu klonen, das *MyDemoRepo*in der Region USA Ost (Ohio) benannt ist:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

Wenn Sie HTTPS mit git-remote-codecommit verwenden:

```
git clone codecommit://MyDemoRepo my-demo-repo
```

Bei Verwendung von SSH:

```
git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

> **Note**
>
> Wenn bei dem Versuch, ein Repository zu klonen, ein Fehler angezeigt wird, haben Sie möglicherweise die Einrichtung nicht abgeschlossen, die für Ihren lokalen Computer erforderlich ist. Weitere Informationen finden Sie unter Einrichten für AWS CodeCommit.

Schritt 3: Erstelle deinen ersten Commit

In diesem Schritt erstellen Sie Ihren ersten Commit in Ihrem lokalen Repo. Dazu erstellen Sie zwei Beispieldateien in Ihrem lokalen Repo. Sie verwenden Git, um die Änderung in Ihrem lokalen Repository bereitzustellen und die Änderung dann in Ihr lokales Repository zu übertragen.

1. Erstellen Sie mit einem Texteditor die beiden folgenden Beispieltextdateien in Ihrem Verzeichnis. Geben Sie den Dateien die Namen `cat.txt` und `dog.txt`:

```
cat.txt
-------
The domestic cat (Felis catus or Felis silvestris catus) is a small, usually furry,
 domesticated, and carnivorous mammal.
```

```
dog.txt
-------
The domestic dog (Canis lupus familiaris) is a canid that is known as man's best
 friend.
```

2. Führen Sie ausgit config, um Ihren Benutzernamen und Ihre E-Mail-Adresse, dargestellt durch Platzhalter *`your-user-name`*, *`your-email-address`*zu Ihrem lokalen Repository hinzuzufügen. Dies vereinfacht die Identifizierung der erstellten Commits:

```
git config --local user.name "your-user-name"
git config --local user.email your-email-address
```

3. Wenn Sie Ihren Standard-Branch-Namen nicht global festgelegt haben, als Sie das lokale Repository erstellt haben, führen Sie den folgenden Befehl aus, um den Standard-Branch-Namen auf festzulegen: **`main`**

```
git config --local init.defaultBranch main
```

4. Führen Sie git add aus, um die Änderung per Stage zu übertragen:

```
git add cat.txt dog.txt
```

5. Führen Sie git commit aus, um die Änderung festzuschreiben:

```
git commit -m "Added cat.txt and dog.txt"
```

> **Tip**
>
> Um Details zu dem soeben durchgeführten Commit anzuzeigen, führen Sie git log aus.

Schritt 4: Push deinen ersten Commit

In diesem Schritt pushen Sie den Commit von Ihrem lokalen Repository in Ihr CodeCommit Repository.

Führen Sie den Befehl ausgit push, um Ihren Commit über den Standard-Remote-Namen, den Git für Ihr CodeCommit Repository (`origin`) verwendet, aus dem Standard-Branch in Ihrem lokalen Repo (`main`) zu pushen:

```
git push -u origin main
```

> **Tip**
>
> Nachdem du Dateien in dein CodeCommit Repository gepusht hast, kannst du die CodeCommit Konsole verwenden, um den Inhalt anzusehen. Weitere Informationen finden Sie unter Durchsuchen Sie Dateien in einem Repository.

Schritt 5: Teile das CodeCommit Repository und führe einen weiteren Commit per Push und Pull durch

In diesem Schritt teilst du Informationen über das CodeCommit Repository mit einem anderen Teammitglied. Das Teammitglied verwendet diese Informationen, um eine lokale Kopie zu erstellen, einige Änderungen daran vorzunehmen und dann die geänderte lokale Kopie in Ihr CodeCommit Repository zu übertragen. Anschließend ziehen Sie die Änderungen aus dem CodeCommit Repository in Ihr lokales Repository.

In diesem Tutorial wird der zweite Benutzer simuliert. Dazu lassen Sie Git zusätzlich zu dem in Schritt 2 erstellten Verzeichnis ein weiteres Verzeichnis erstellen. (In der Regel befindet sich dieses Verzeichnis auf einem anderen Computer.) Dieses neue Verzeichnis ist eine Kopie Ihres CodeCommit Repositorys. Alle Änderungen, die Sie an dem vorhandenen oder diesem neuen Verzeichnis vornehmen, werden unabhängig voneinander vorgenommen. Die einzige Möglichkeit, Änderungen an diesen Verzeichnissen zu erkennen, besteht darin, sie aus dem CodeCommit Repository abzurufen.

Obwohl sie sich auf demselben lokalen Computer befinden, nennen wir das vorhandene Verzeichnis lokales Repository (local repo) und das neue Verzeichnis freigegebenes Repository (shared repo).

Aus dem neuen Verzeichnis erhalten Sie eine separate Kopie des CodeCommit Repositorys. Anschließend fügen Sie eine neue Beispieldatei hinzu, übertragen die Änderungen in das gemeinsam genutzte Repo und übertragen dann den Commit aus dem gemeinsam genutzten Repository in Ihr CodeCommit Repository.

Schließlich ziehen Sie die Änderungen aus Ihrem Repository in Ihr lokales Repo und durchsuchen es dann, um die Änderungen zu sehen, die vom anderen Benutzer übernommen wurden.

1. Wechseln Sie in das Verzeichnis `/tmp` oder in das Verzeichnis `c:\temp`.
2. Führen Sie git clone aus, um eine Pull-Übertragung einer Kopie des Repositorys in das freigegebene Repository (shared repo) durchzuführen:

   Für HTTPS:

   ```
   git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
    shared-demo-repo
   ```

   Für HTTPS mit git-remote-codecommit:

   ```
   git clone codecommit://MyDemoRepo shared-demo-repo
   ```

   Für SSH:

   ```
   git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo shared-
   demo-repo
   ```

   > **Note**
   >
   > Beim Klonen eines Repositorys mit SSH auf Windows-Betriebssystemen müssen Sie möglicherweise die SSH-Schlüssel-ID wie folgt der Verbindungszeichenfolge hinzufügen:
   >
   > ```
   > git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/
   > repos/MyDemoRepo my-demo-repo
   > ```
   >
   > Weitere Informationen finden Sie unter Für SSH-Verbindungen unter Windows.

In diesem Befehl `MyDemoRepo` steht der Name Ihres CodeCommit Repositorys. `shared-demo-repo`ist der Name des Verzeichnisses, das Git in dem `/tmp` Verzeichnis oder dem `c:\temp` Verzeichnis erstellt. Wenn Git das Verzeichnis erstellt hat, überträgt Git eine Kopie Ihres Repositorys per Pull in das Verzeichnis `shared-demo-repo`.

3. Wechseln Sie in das Verzeichnis `shared-demo-repo`:

```
(For Linux, macOS, or Unix) cd /tmp/shared-demo-repo
(For Windows) cd c:\temp\shared-demo-repo
```

4. Führen Sie git config den Befehl aus, um einen weiteren Benutzernamen und eine E-Mail-Adresse hinzuzufügen, die durch Platzhalter *other-user-name*und *other-email-address*dargestellt werden. Dadurch wird es einfacher, festzustellen, welche Commits von dem anderen Benutzer durchgeführt werden:

```
git config --local user.name "other-user-name"
git config --local user.email other-email-address
```

5. Erstellen Sie mit einem Texteditor die folgende Beispieltextdateien im Verzeichnis `shared-demo-repo`. Benennen Sie die Datei `horse.txt`:

```
horse.txt
-------
The horse (Equus ferus caballus) is one of two extant subspecies of Equus ferus.
```

6. Führen Sie git add aus, um die Änderung per Stage an das freigegebene Repository (shared repo) zu übertragen:

```
git add horse.txt
```

7. Führen Sie git commit aus, um die Änderung im dem freigegebenen Repository (shared repo) festzuschreiben:

```
git commit -m "Added horse.txt"
```

8. Führen Sie ausgit push, um Ihren ersten Commit über den Standard-Remote-Namen, den Git für Ihr CodeCommit Repository (`origin`) verwendet, aus dem Standard-Branch in Ihrem lokalen Repo (`main`) zu pushen:

```
git push -u origin main
```

9. Wechsle zu deinem lokalen Repo und führe den Befehl ausgit pull, um den Commit, den das gemeinsame Repo für das Repository vorgenommen hat, in dein lokales Repo zu ziehen. CodeCommit Führen Sie anschließend git log aus, um den Commit anzeigen zu lassen, der von dem freigegebenen Repository (shared repo) initiiert wurde.

Schritt 6: Einen Branch erstellen und teilen

In diesem Schritt erstellen Sie einen Branch in Ihrem lokalen Repository, nehmen einige Änderungen vor und übertragen den Branch dann per Push in Ihr CodeCommit Repository. Anschließend ziehen Sie den Branch aus Ihrem CodeCommit Repository in das gemeinsam genutzte Repo.

Ein Branch ermöglicht Ihnen die unabhängige Entwicklung einer abweichenden Version der Repository-Inhalte (zum Beispiel, um an einer neuen Softwarefunktion zu arbeiten, ohne die Arbeit Ihrer Teammitglieder zu beeinträchtigen). Wenn diese Funktion stabil ist, können Sie den Branch mit einem stabileren Software-Branch zusammenführen.

Sie erstellen den Branch mit Git und verweisen damit auf den ersten Commit, den Sie durchgeführt haben. Sie verwenden Git, um den Branch in das CodeCommit Repository zu übertragen. Dann wechselst du zu deinem gemeinsamen Repo und verwendest Git, um den neuen Branch in dein gemeinsames lokales Repo zu ziehen und den Branch zu erkunden.

1. Führen git checkout Sie von Ihrem lokalen Repo aus den Befehl aus und geben Sie dabei den Namen des Branches (zum Beispiel`MyNewBranch`) und die ID des ersten Commits an, den Sie im lokalen Repository vorgenommen haben.

   Wenn Sie die Commit-ID nicht kennen, führen Sie git log aus, um sie abzurufen. Stellen Sie sicher, dass in dem Commit Ihr Benutzername und Ihre E-Mail-Adresse und nicht die Daten des anderen Benutzers verwendet werden. Dies dient dazu, zu simulieren, dass `main` es sich um eine stabile Version des CodeCommit Projektarchivs handelt und dass der `MyNewBranch` Branch für ein neues, relativ instabiles Feature vorgesehen ist:

   ```
   git checkout -b MyNewBranch commit-ID
   ```

2. Führen Sie den Befehl ausgit push, um den neuen Branch vom lokalen Repository an das CodeCommit Repository zu senden:

```
git push origin MyNewBranch
```

3. Übertragen Sie nun den Branch per Pull an das freigegebene Repository (shared repo) und überprüfen Sie die Ergebnisse:

   1. Wechseln Sie in das Verzeichnis des freigegebenen Repositorys (shared-demo-repo).
   2. Rufen Sie den neuen Branch per Pull ab (git fetch origin).
   3. Überprüfen Sie, ob der Branch per Pull übertragen wurde (git branch --all zeigt eine Liste aller Branches für das Repository an).
   4. Umstellen auf den neuen Branch (git checkout MyNewBranch).
   5. Überprüfen Sie, ob Sie zum Branch `MyNewBranch` gewechselt haben, indem Sie git status oder git branch ausführen. Die Ausgabe zeigt Ihnen, in welchem Branch Sie sich befinden. In diesem Fall sollte es sich um `MyNewBranch` handeln.
   6. Lassen Sie die Liste der Commits in dem Branch anzeigen (git log).

   Im Folgenden sind die aufzurufenden Git-Befehle aufgeführt:

   ```
   git fetch origin
   git branch --all
   git checkout MyNewBranch
   git branch or git status
   git log
   ```

4. Wechseln Sie zurück zum Branch `main` und lassen Sie die Liste der zugehörigen Commits anzeigen. Die Git-Befehle sollten wie folgt aussehen:

   ```
   git checkout main
   git log
   ```

5. Wechseln Sie zu dem `main` Zweig in Ihrem lokalen Repo. Sie können git status oder git branch ausführen. Die Ausgabe zeigt Ihnen, in welchem Branch Sie sich befinden. In diesem Fall sollte es sich um `main` handeln. Die Git-Befehle sollten wie folgt aussehen:

   ```
   git checkout main
   git branch or git status
   ```

Schritt 7: Erstelle und teile ein Tag

In diesem Schritt erstellen Sie zwei Tags in Ihrem lokalen Repository, verknüpfen die Tags mit Commits und übertragen die Tags dann in Ihr CodeCommit Repository. Anschließend ziehen Sie die Änderungen aus dem CodeCommit Repository in das gemeinsam genutzte Repo.

Ein Tag wird verwendet, um einem Commit (oder einem Branch oder sogar einem anderen Tag) einen für Menschen lesbaren Namen zuzuweisen. Dies könnte zum Beispiel vorkommen, wenn Sie einen Commit mit `v2.1` markieren möchten. Ein Commit, ein Branch oder ein Tag kann mit beliebig vielen Tags verknüpft sein, aber ein bestimmter Tag ist mit nur einem bestimmten Commit, Branch oder Tag verknüpft. In diesem Tutorial markieren Sie einen Commit mit `release` und einen weiteren mit `beta`.

Sie erstellen die Tags mit Git und verweisen mit dem Tag `release` auf den ersten Commit und mit dem Tag `beta` auf den vom anderen Benutzer durchgeführten Commit. Anschließend verwenden Sie Git, um die Tags in das CodeCommit Repository zu übertragen. Dann wechselst du zu deinem gemeinsamen Repo und verwendest Git, um die Tags in dein gemeinsames lokales Repo zu ziehen und die Tags zu erkunden.

1. Führen git tag Sie von Ihrem lokalen Repo aus den Befehl aus und geben Sie dabei den Namen des neuen Tags (`release`) und die ID des ersten Commits an, den Sie im lokalen Repo vorgenommen haben.

   Wenn Sie die Commit-ID nicht kennen, führen Sie git log aus, um sie abzurufen. Stellen Sie sicher, dass in dem Commit Ihr Benutzername und Ihre E-Mail-Adresse und nicht die Daten des anderen Benutzers verwendet werden. Dies dient dazu, zu simulieren, dass es sich bei Ihrem Commit um eine stabile Version des CodeCommit Repositorys handelt:

   ```
   git tag release commit-ID
   ```

   Führen Sie git tag erneut aus, um den Commit des anderen Benutzers mit dem Tag `beta` zu versehen. Dadurch wird simuliert, dass der Commit für eine neue, relativ instabile Funktion durchgeführt wird:

   ```
   git tag beta commit-ID
   ```

2. Führen Sie ausgit push --tags, um die Tags an das CodeCommit Repository zu senden.

3. Übertragen Sie nun die Tags per Pull an das freigegebene Repository (shared repo) und überprüfen Sie die Ergebnisse:

   1. Wechseln Sie in das Verzeichnis des freigegebenen Repositorys (shared-demo-repo).
   2. Rufen Sie die neuen Tags per Pull ab (git fetch origin).
   3. Überprüfen Sie, ob die Tags per Pull übertragen wurden (git tag zeigt eine Liste aller Tags für das Repository an).
   4. Zeigen Sie Informationen zu jedem Tag an (git log release und git log beta).

   Im Folgenden sind die aufzurufenden Git-Befehle aufgeführt:

   ```
   git fetch origin
   git tag
   git log release
   git log beta
   ```

4. Probieren Sie das auch im lokalen Repo aus:

   ```
   git log release
   git log beta
   ```

Schritt 8: Zugriffsberechtigungen einrichten

In diesem Schritt erteilen Sie einem Benutzer die Erlaubnis, das gemeinsam genutzte Repo mit dem CodeCommit Repository zu synchronisieren. Dieser Schritt ist optional. Es wird Benutzern empfohlen, die mehr darüber erfahren möchten, wie der Zugriff auf CodeCommit Repositorys gesteuert werden kann, wenn Benutzer Git-Anmeldeinformationen verwenden oder SSH-Schlüsselpaare mit IAM-Benutzern für den Zugriff auf Repositorys verwendet werden. CodeCommit

> **Note**
>
> Wenn Sie Verbundzugriff, temporäre Anmeldeinformationen oder einen Web-Identitätsanbieter wie IAM Identity Center verwenden, richten Sie Benutzer, Zugriff und Berechtigungen für Ihren Identitätsanbieter ein und verwenden Sie dann. git-remote-codecommit Weitere Informationen finden Sie unter Einrichtungsschritte für HTTPS-

Verbindungen zuAWS CodeCommitmitgit-remote-codecommit und Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen.

Dazu verwenden Sie die IAM-Konsole, um einen Benutzer zu erstellen, der standardmäßig nicht berechtigt ist, das gemeinsam genutzte Repository mit dem Repository zu synchronisieren. CodeCommit Sie können git pull ausführen, um dies zu überprüfen. Wenn der neue Benutzer keine Berechtigung zur Synchronisierung hat, funktioniert der Befehl nicht. Dann kehren Sie zur IAM-Konsole zurück und wenden eine Richtlinie an, die dem Benutzer die Verwendung ermöglicht. git pull Sie können wieder git pull ausführen, um dies zu überprüfen.

Dieser Schritt wurde unter der Annahme geschrieben, dass Sie berechtigt sind, IAM-Benutzer in Ihrem Amazon Web Services Services-Konto zu erstellen. Wenn Sie diese Berechtigungen nicht haben, können Sie diesen Schritt nicht ausführen. Fahren Sie mit Schritt 9: Bereinigen fort, um die Ressourcen zu bereinigen, die Sie für das Tutorial verwendet haben.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.

   Stellen Sie sicher, dass Sie sich mit demselben Benutzernamen und demselben Passwort anmelden, das Sie auch in Einrichtung verwendet haben.
2. Wählen Sie im Navigationsbereich Users und dann Create New Users aus.
3. Geben Sie im ersten Feld unter Enter User Names (Benutzernamen eingeben) einen Beispiel-Benutzernamen ein (zum Beispiel **`JaneDoe-CodeCommit`**). Aktivieren Sie das Kontrollkästchen Generate an access key for each user (Zugriffsschlüssel für jeden Benutzer generieren) und wählen Sie dann Create (Erstellen).
4. Wählen Sie die Option Show User Security Credentials. Notieren Sie sich die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel oder wählen Sie Download Credentials.
5. Folgen Sie den Anweisungen unterFür HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden, um die Anmeldeinformationen des IAM-Benutzers zu generieren und bereitzustellen.

   Wenn Sie SSH verwenden möchten, befolgen Sie die Anweisungen in SSH und Linux, macOS oder Unix: Richte die öffentlichen und privaten Schlüssel für Git ein undCodeCommit oder Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein, um für den Benutzer öffentliche und private Schlüssel einzurichten.
6. Führen Sie git pull. Die folgende Fehlermeldung sollte angezeigt werden:

Für HTTPS:

```
fatal: unable to access 'https://git-codecommit.us-
east-2.amazonaws.com/v1/repos/repository-name/': The requested URL
returned error: 403.
```

Für SSH:

```
fatal: unable to access 'ssh://git-codecommit.us-east-2.amazonaws.com/
v1/repos/repository-name/': The requested URL returned error: 403.
```

Der Fehler tritt auf, weil der neue Benutzer nicht berechtigt ist, das gemeinsam genutzte Repository mit dem CodeCommit Repository zu synchronisieren.

7. Kehren Sie zur IAM-Konsole zurück. Wählen Sie im Navigationsbereich Policies und dann Create Policy. (Wenn die Schaltfläche Get Started (Erste Schritte) angezeigt wird, klicken Sie darauf und wählen Sie anschließend Create Policy (Richtlinie erstellen) aus.)
8. Klicken Sie neben Create Your Own Policy auf Select.
9. Geben Sie in das Feld Policy Name (Richtlinienname) einen Namen ein (z. B. **`CodeCommitAccess-GettingStarted`**).
10. Geben Sie im Feld Richtliniendokument Folgendes ein, sodass ein IAM-Benutzer Daten aus jedem Repository abrufen kann, das dem IAM-Benutzer zugeordnet ist:

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

> **Tip**
>
> Wenn Sie möchten, dass der IAM-Benutzer Commits in jedes Repository übertragen kann, das dem IAM-Benutzer zugeordnet ist, geben Sie stattdessen Folgendes ein:
>
> ```
> {
>   "Version": "2012-10-17",
>   "Statement": [
>     {
>       "Effect": "Allow",
>       "Action": [
>         "codecommit:GitPull",
>         "codecommit:GitPush"
>       ],
>       "Resource": "*"
>     }
>   ]
> }
> ```
>
> Informationen zu anderen CodeCommit Aktions- und Ressourcenberechtigungen, die Sie Benutzern gewähren können, finden Sie unter. Authentifizierung und Zugriffskontrolle für AWS CodeCommit

11. Klicken Sie im Navigationsbereich auf Users (Benutzer).
12. Wählen Sie den Beispiel-Benutzernamen (zum Beispiel **`JaneDoe-CodeCommit`**), dem Sie die Richtlinie anfügen möchten.
13. Wählen Sie die Registerkarte Berechtigungen.
14. Klicken Sie unter Managed Policies auf Attach Policy.
15. Wählen Sie die **`CodeCommitAccess-GettingStarted`**-Richtlinie aus, die Sie eben erstellt haben, und dann Attach Policy (Richtlinie anfügen).
16. Führen Sie git pull. Nun sollte der Befehl funktionieren und eine Meldung des Typs `Already up-to-date` sollte angezeigt werden.
17. Wenn Sie HTTPS verwenden, wechseln Sie zu Ihren ursprünglichen Git-Anmeldeinformationen oder, falls Sie git-remote-codecommit verwenden, zu Ihrem üblichen Profil. Weitere Informationen finden Sie in den Anweisungen unter Einrichtung für HTTPS-Benutzer mit Git-Anmeldeinformationen oder Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit.

Wenn Sie SSH verwenden, wechseln Sie zu Ihren ursprünglichen Schlüsseln. Weitere Informationen finden Sie unter SSH und Linux, macOS oder Unix: Richte die öffentlichen und privaten Schlüssel für Git ein undCodeCommit oder Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein.

Sie haben das Ende dieses Tutorials erreicht.

Schritt 9: Bereinigen

In diesem Schritt löschen Sie das CodeCommit Repository, das Sie in diesem Tutorial verwendet haben, sodass Ihnen der Speicherplatz nicht weiter in Rechnung gestellt wird.

Sie entfernen auch das lokale Repository und das gemeinsam genutzte Repo auf Ihrem lokalen Computer, da sie nach dem Löschen des CodeCommit Repositorys nicht mehr benötigt werden.

> **Important**
>
> Nachdem Sie dieses Repository gelöscht haben, können Sie es nicht mehr in ein lokales Repo oder ein gemeinsam genutztes Repository klonen. Sie werden auch nicht in der Lage sein, Daten aus einem lokalen oder gemeinsam genutzten Repo daraus abzurufen oder Daten dorthin zu übertragen. Diese Aktion kann nicht rückgängig gemacht werden.

Um das CodeCommit Repository zu löschen (Konsole)

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie auf der Seite Dashboard in der Liste der Repositorys den Eintrag MyDemoRepo.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Klicken Sie auf der Seite Settings unter Delete repository auf Delete repository.
5. Geben Sie in das Feld neben Type the name of the repository to confirm deletion (Namen des Repositorys eingeben, um das Löschen zu bestätigen) die Zeichenfolge **`MyDemoRepo`** ein und klicken Sie dann auf Delete (Löschen).

Um das CodeCommit Repository zu löschen (AWS CLI)

Führen Sie den Befehl delete-repository aus:

```
aws codecommit delete-repository --repository-name MyDemoRepo
```

Um das lokale Repo und das gemeinsame Repo zu löschen

Für Linux, macOS oder Unix:

```
cd /tmp
rm -rf /tmp/my-demo-repo
rm -rf /tmp/shared-demo-repo
```

Für Windows:

```
cd c:\temp
rd /s /q c:\temp\my-demo-repo
rd /s /q c:\temp\shared-demo-repo
```

Produkt- und Serviceintegrationen mit AWS CodeCommit

Standardmäßig CodeCommit ist es in eine Reihe von AWS Diensten integriert. Sie können es auch CodeCommit mit Produkten und Dienstleistungen außerhalb von verwenden AWS. Die folgenden Informationen können Ihnen bei der Konfiguration CodeCommit zur Integration mit den von Ihnen verwendeten Produkten und Diensten helfen.

> **Note**
>
> Durch die Integration mit CodePipeline können Sie automatisch Commits erstellen und für ein CodeCommit Repository bereitstellen. Um mehr zu erfahren, folgen Sie den Schritten im AWS Handbuch DevOps Erste Schritte.

Themen

- Integration mit anderen AWS Diensten
- Integrationsbeispiele der Community

Integration mit anderen AWS Diensten

CodeCommit ist in die folgenden AWS Dienste integriert:

AWS Amplify	AWS Amplifymacht es einfach, skalierbare mobile Anwendungen zu erstellen, zu konfiguri eren und zu implementieren, die von AWS. Amplify sorgt für eine nahtlose Bereitstellung und Verwaltung Ihres mobilen Backends und bietet ein einfaches Framework für die problemlose Integration Ihres Backends in Ihre iOS-, Android-, Web- und React Native-Frontends. Außerdem automatisiert Amplify den Anwendungsfreigabeprozess Ihres Frontends und Backends, sodass Sie Funktionen schneller bereitstellen können.

	Sie können Ihr CodeCommit Repository in der Amplify-Konsole verbinden. Nachdem Sie die Amplify-Konsole autorisiert haben, ruft Amplify ein Zugriffstoken vom Repository-Anbieter ab, speichert das Token jedoch nicht auf den Servern. AWS Amplify greift auf Ihr Repository nur mit Bereitstellungsschlüsseln zu, die in einem bestimmten Repository installiert sind. Weitere Informationen: • AWS Amplify Benutzerhandbuch • Erste Schritte
AWS Cloud9	AWS Cloud9 enthält eine Sammlung von Tools, mit denen Sie Software kodieren, erstellen, ausführen, testen, debuggen und in der Cloud veröffentlichen können. Diese Sammlung von Tools wird als AWS Cloud9 integrierte Entwicklungsumgebung oder IDE bezeichnet. Sie greifen über einen Webbrowser auf die AWS Cloud9 IDE zu. Die IDE bietet eine umfassende Codebearbeitung mit Unterstützung mehrerer Programmiersprachen und Laufzeit-Debugger sowie ein integriertes Terminal. Weitere Informationen: • AWS Cloud9 -Benutzerhandbuch • AWS CodeCommit Beispiel für AWS Cloud9 • Integration von AWS Cloud9 in AWS CodeCommit

AWS CloudFormation	AWS CloudFormationist ein Service, der Sie bei der Modellierung und Einrichtung Ihrer AWS Ressourcen unterstützt, sodass Sie weniger Zeit mit der Verwaltung dieser Ressourcen verbringen und sich mehr auf Ihre Anwendung en konzentrieren können. Sie erstellen eine Vorlage, die Ressourcen, einschließlich eines CodeCommit Repositorys, beschreibt und die Bereitstellung und Konfiguration dieser Ressourcen für Sie AWS CloudFormation übernimmt. Weitere Informationen: • Seite für AWS CodeCommit -Repository-Ressourcen
AWS CloudTrail	CloudTrailerfasst AWS API-Aufrufe und zugehörige Ereignisse, die von oder im Namen eines Amazon Web Services Services-Kontos getätigt wurden, und übermittelt Protokoll dateien an einen von Ihnen angegebenen Amazon S3 S3-Bucket. Sie können so konfiguri eren CloudTrail , dass API-Aufrufe von der AWS CodeCommit Konsole, CodeCommit Befehle vom AWS CLI lokalen Git-Client und von der CodeCommit API erfasst werden. Weitere Informationen: • Protokollierung von AWS CodeCommit-API-Aufrufen mit AWS CloudTrail

CloudWatch Amazon-Veranstaltungen	CloudWatch Events bietet einen Stream von Systemereignissen, die Änderungen an AWS Ressourcen beschreiben, nahezu in Echtzeit. Mithilfe einfacher Regeln, die Sie schnell einrichten können, können Sie Ereignisse zuordnen und sie an eine oder mehrere Zielfunktionen oder Streams weiterlei ten. CloudWatch Events erkennt betriebli che Änderungen, sobald sie eintreten. CloudWatch Events reagiert auf diese betriebli chen Änderungen und ergreift bei Bedarf Maßnahmen, indem es Nachrichten sendet, um auf die Umgebung zu reagieren, Funktionen aktiviert, Änderungen vornimmt und Statusinf ormationen erfasst. Sie können CloudWatch Ereignisse so konfiguri eren, dass sie CodeCommit Repositorys überwachen und auf Repository-Ereignisse reagieren, indem Sie auf Streams, Funktione n, Aufgaben oder andere Prozesse in anderen AWS Services wie Amazon Simple Queue Service, Amazon Kinesis und vielen AWS Lambda mehr abzielen. Weitere Informationen: • CloudWatch Benutzerhandbuch für Veranstal tungen • AWS CodeCommit -Ereignisse • Blogbeitrag: Serverlose AWS CodeCommit Workflows mit Amazon CloudWatch Events und JGit erstellen

AWS CodeBuild	CodeBuildist ein vollständig verwalteter Build-Service in der Cloud, der Ihren Quellcode kompiliert, Komponententests durchführt und Artefakte erzeugt, die sofort einsatzbereit sind. Sie können den zu erstellenden Quellcode und die Build-Spezifikation in einem CodeCommit Repository speichern. Sie können es CodeBuild direkt mit CodeCommit verwenden oder Sie können beides integrieren CodeBuild und CodeCommit in eine Continuous-Delivery-Pipeline mit integrieren CodePipeline. Weitere Informationen: • Planen eines Builds • Erstellen eines Build-Projekts • Verwenden Sie CodePipeline with AWS CodeBuild , um Builds auszuführen

CodeGuru Amazon-Rezensent	Amazon CodeGuru Reviewer ist ein automatisierter Code-Review-Service, der mithilfe von Programmanalyse und maschinellem Lernen häufig auftretende Probleme erkennt und Korrekturen in Ihrem Java- oder Python-Code empfiehlt. Sie können Repositorys in Ihrem Amazon Web Services Services-Konto mit CodeGuru Reviewer verknüpfen. Wenn Sie dies tun, erstellt der CodeGuru Prüfer eine servicebezogene Rolle, die es dem CodeGuru Prüfer ermöglicht, den Code in allen Pull-Requests zu analysieren, die nach der Zuordnung erstellt wurden. Weitere Informationen: • Ein AWS CodeCommit Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben • Amazon CodeGuru Reviewer-Benutzerhandbuch

AWS CodePipeline	CodePipelineist ein Continuous-Delivery-Service, mit dem Sie die zur Veröffentlichung Ihrer Software erforderlichen Schritte modellieren, visualisieren und automatisieren können. Sie können ein CodeCommit Repository so konfigurieren CodePipeline , dass es als Quellaktion in einer Pipeline verwendet wird, und das Erstellen, Testen und Bereitstellen Ihrer Änderungen automatisieren. Weitere Informationen: • Einfache Pipeline-Komplettlösung mit und CodePipeline AWS CodeCommit • Migrieren Sie zu Amazon CloudWatch Events Change Detection für Pipelines mit einem Repository CodeCommit • Änderungserkennungsmethoden zum automatischen Starten von Pipelines

AWS CodeStar	AWS CodeStarist ein cloudbasierter Service für die Erstellung, Verwaltung und Bearbeitung von Softwareentwicklungsprojekten AWS. Sie können im Handumdrehen Anwendungen für ein AWS CodeStar Projekt entwickeln, erstellen und bereitstellen. AWS Ein AWS CodeStar Projekt erstellt und integriert AWS Dienste für Ihre Projektentwicklungs-Toolchain, einschließlich eines CodeCommit Repositorys für das Projekt. AWS CodeStar weist Teammitgliedern auch Berechtigungen für dieses Projekt zu. Diese Berechtigungen werden automatisch angewendet, einschließlich Berechtigungen für den Zugriff CodeCommit, die Erstellung und Verwaltung von Git-Anmeldeinformationen und mehr. Sie können Repositorys, die für AWS CodeStar Projekte erstellt wurden, genau wie jedes andere CodeCommit Repository konfigurieren, indem Sie die AWS CodeCommit Konsole, CodeCommit Befehle vom AWS CLI, dem lokalen Git-Client und der CodeCommit API verwenden. Weitere Informationen: • Arbeiten mit Repositorien • Arbeiten mit AWS CodeStar -Projekten • Arbeiten mit AWS CodeStar -Teams

AWS Elastic Beanstalk	Elastic Beanstalk ist ein Managed Service, der es einfach macht, Anwendungen in der AWS Cloud bereitzustellen und zu verwalten , ohne sich Gedanken über die Infrastruktur machen zu müssen, auf der diese Anwendung en ausgeführt werden. Sie können die Elastic Beanstalk Beanstalk-Befehlszeilenschnittstelle (EB CLI) verwenden, um Ihre Anwendung direkt aus einem neuen oder bestehenden CodeCommit Repository bereitzustellen. Weitere Informationen: • Verwenden der EB CLI mit AWS CodeCommit • Verwenden eines vorhandenen Repositorys AWS CodeCommit • eb codesource (EB CLI-Befehl)
AWS Key Management Service	AWS KMS ist ein verwalteter Service, der das Erstellen und Kontrollieren der Schlüssel zum Verschlüsseln Ihrer Daten vereinfacht. Wird standardmäßig AWS KMS zum Verschlüsseln von Repositorys CodeCommit verwendet. Weitere Informationen: • AWS KMS und Verschlüsselung

AWS Lambda	Mit Lambda können Sie Code ausführen, ohne Server bereitzustellen oder zu verwalten. Sie können Trigger für CodeCommit Repositorys konfigurieren, die Lambda-Funktionen als Reaktion auf Repository-Ereignisse aufrufen. Weitere Informationen: • Erstellen Sie einen Trigger für eine Lambda-Funktion • AWS Lambda Entwicklerhandbuch
Amazon Simple Notification Service	Amazon SNS ist ein Webservice, der es Anwendungen, Endbenutzern und Geräten ermöglicht, sofort Benachrichtigungen aus der Cloud zu senden und zu empfangen. Sie können Auslöser für CodeCommit Repositorys konfigurieren, die Amazon SNS SNS-Benachrichtigungen als Reaktion auf Repository-Ereignisse senden. Sie können Amazon SNS SNS-Benachrichtigungen auch verwenden, um sie in andere AWS Dienste zu integrieren. Sie können beispielsweise eine Amazon SNS SNS-Benachrichtigung verwenden, um Nachrichten an eine Amazon Simple Queue Service-Warteschlange zu senden. Weitere Informationen: • Einen Auslöser für ein Amazon SNS SNS-Thema erstellen • Entwicklerhandbuch zu Amazon Simple Notification Service

Integrationsbeispiele der Community

In den folgenden Abschnitten werden Links zu Blog-Posts, Artikel und von der Community bereitgestellte Beispiele vorgestellt.

> **Note**
>
> Diese Links werden nur zu Informationszwecken bereitgestellt und sollten weder als umfassende Liste noch als Bestätigung des Inhalts der Beispiele betrachtet werden. AWS ist nicht verantwortlich für den Inhalt oder die Richtigkeit externer Inhalte.

Themen

- Blog-Posts
- Codebeispiele

Blog-Posts

- Integration SonarQube als Pull Request Approver auf AWS CodeCommit

  Erfahren Sie, wie Sie ein CodeCommit Repository erstellen, für das eine erfolgreiche SonarQube Qualitätsanalyse erforderlich ist, bevor Pull-Requests zusammengeführt werden können.

  Veröffentlichung: 12. Dezember 2019
- Migration zu AWS CodeCommitAWS CodePipeline, und AWS CodeBuild Von GitLab

  Erfahren Sie, wie Sie mehrere Repositorys zu „ AWS CodeCommit von“ migrieren GitLab und mithilfe von und eine CI/CD-Pipeline einrichten. AWS CodePipeline AWS CodeBuild

  Veröffentlichung: 22. November 2019
- Implementierung GitFlow mithilfe von AWS CodePipeline,, und AWS CodeCommitAWS CodeBuildAWS CodeDeploy

  Erfahren Sie, wie Sie GitFlow mithilfe von AWS CodePipeline AWS CodeCommit, AWS CodeBuild, und implementieren AWS CodeDeploy.

  Veröffentlichung: 22. Februar 2019

- Git mit AWS CodeCommit mehreren AWS Konten verwenden

  Erfahren Sie, wie Sie Ihre Git-Konfiguration für mehrere Amazon Web Services Services-Konten verwalten können.

  Veröffentlichung: 12. Februar 2019

- Validierung von AWS CodeCommit Pull-Requests mit und AWS CodeBuildAWS Lambda

  Erfahre, wie du Pull-Requests mit AWS CodeCommit AWS CodeBuild, und AWS Lambda validierst. Indem Sie Tests mit den vorgeschlagenen Änderungen durchführen, bevor Sie sie in den Standard-Branch zusammenführen, können Sie dazu beitragen, ein hohes Qualitätsniveau bei Pull-Requests sicherzustellen, potenzielle Probleme zu catch und das Vertrauen der Entwickler in ihre Änderungen zu stärken.

  Veröffentlichung: 11. Februar 2019

- Verwenden von föderierten Identitäten mit AWS CodeCommit

  Erfahren Sie, wie Sie AWS CodeCommit mithilfe der in Ihrem Unternehmen verwendeten Identitäten auf Repositorys zugreifen können.

  Veröffentlicht am 5. Oktober 2018

- Verfeinerung des Zugriffs auf Filialen in AWS CodeCommit

  Erfahren Sie, wie Sie Commits auf Repository-Branches beschränken können, indem Sie eine IAM-Richtlinie erstellen und anwenden, die einen Kontextschlüssel verwendet.

  Veröffentlicht am 16. Mai 2018

- AWS CodeCommit Repositorys mit Fargate zwischen Regionen replizieren AWS

  Erfahren Sie, wie Sie mithilfe einer serverlosen Architektur die kontinuierliche Replikation eines CodeCommit Repositorys von einer AWS Region in eine andere einrichten.

  Veröffentlicht am 11. April 2018

- Verteilung Ihrer Infrastruktur AWS OpsWorks for Chef Automate

  Erfahren Sie CodePipeline,, CodeCommit, und AWS Lambda wie Sie sicherstellen können CodeBuild, dass Kochbücher und andere Konfigurationen konsistent auf zwei oder mehr Chef-Servern bereitgestellt werden, die sich auf einem oder mehreren befinden. AWS-Regionen

Veröffentlicht am 9. März 2018

- Peanut Butter und Chocolate: Azure Functions CI/CD-Pipeline mit AWS CodeCommit

  Erfahren Sie, wie Sie eine PowerShell auf Azure Functions basierende CI/CD-Pipeline erstellen, bei der der Code in einem Repository gespeichert wird. CodeCommit

  Veröffentlicht am 19. Februar 2018

- Kontinuierliche Bereitstellung auf Kubernetes mithilfe von AWS CodePipeline,, AWS CodeCommitAWS CodeBuild, Amazon ECR und AWS Lambda

  Erfahren Sie, wie Sie Kubernetes verwenden und AWS gemeinsam eine vollständig verwaltete, kontinuierliche Bereitstellungspipeline für containerbasierte Anwendungen erstellen können.

  Veröffentlicht am 11. Januar 2018

- Verwenden Sie AWS CodeCommit Pull-Requests, um Code-Reviews anzufordern und Code zu besprechen

  Erfahre, wie du mithilfe von Pull Requests Codeänderungen in einem CodeCommit Repository überprüfen, kommentieren und interaktiv wiederholen kannst.

  Veröffentlicht am 20. November 2017

- Erstellen Sie serverlose AWS CodeCommit Workflows mit Amazon CloudWatch Events und JGit

  Erfahren Sie, wie Sie mithilfe von CloudWatch Repository-Ereignissen und Zielaktionen in anderen AWS Services Regeln für Ereignisse erstellen, die Änderungen in einem CodeCommit Repository verarbeiten. Beispiele hierfür sind AWS Lambda Funktionen, die Git-Commit-Nachrichtenrichtlinien für Commits durchsetzen, ein CodeCommit Repository replizieren und ein CodeCommit Repository auf Amazon S3 sichern.

  Veröffentlicht am 3. August 2017

- Migration zu AWS CodeCommit

  Erfahre, wie du im Rahmen der Migration von einem anderen Git-Repository auf CodeCommit wann Code in zwei Repositorys pushen kannst. SourceTree

  Veröffentlicht am 6. September 2016

- Richten Sie kontinuierliche Tests mit Appium AWS CodeCommit, Jenkins und ein AWS Device Farm

Erfahren Sie, wie Sie mit Appium, CodeCommit Jenkins und Device Farm einen kontinuierlichen Testprozess für mobile Geräte einrichten.

Veröffentlicht am 2. Februar 2016

- Verwendung AWS CodeCommit mit Git-Repositorys in mehreren Amazon Web Services Services-Konten

  Erfahren Sie, wie Sie Ihr CodeCommit Repository klonen und mit einem Befehl den Credential Helper so konfigurieren, dass er eine bestimmte IAM-Rolle für Verbindungen zu diesem Repository verwendet.

  Veröffentlichung November 2015

- Integrieren und AWS OpsWorksAWS CodeCommit

  Erfahren Sie, wie AWS OpsWorks Sie Apps und Kochbücher von Chef automatisch abrufen können. CodeCommit

  Veröffentlicht am 25. August 2015

- Helper AWS CodeCommit und Credential Helpers GitHub

  Erfahren Sie, wie Sie Ihre Gitconfig-Datei so konfigurieren, dass sie sowohl mit Credential Helpers als auch mit GitHub Credential CodeCommit Helpers funktioniert.

  Veröffentlichung September 2015

- Von Eclipse aus verwenden AWS CodeCommit

  Erfahren Sie, wie Sie die eGit-Tools in Eclipse verwenden, um damit zu arbeiten. CodeCommit

  Veröffentlichung August 2015

- AWS CodeCommit mit Amazon EC2 EC2-Rollenanmeldedaten

  Erfahren Sie, wie Sie bei der Konfiguration des automatisierten Agentenzugriffs auf ein CodeCommit Repository ein Instance-Profil für Amazon EC2 verwenden.

  Veröffentlichung Juli 2015

- Integration AWS CodeCommit mit Jenkins

  Erfahren Sie, wie Sie Jenkins verwenden können CodeCommit , um zwei einfache CI-Szenarien (Continuous Integration) zu unterstützen.

Veröffentlichung Juli 2015

- Integration AWS CodeCommit mit Review Board

  Erfahren Sie, wie Sie sich mithilfe des Review Board-Code-Review-Systems CodeCommit in einen Entwicklungsworkflow integrieren können.

  Veröffentlichung Juli 2015

Codebeispiele

Im Folgenden finden Sie Codebeispiele, die für CodeCommit Benutzer von Interesse sein könnten.

- Mac OS X Script to Periodically Delete Cached Credentials in the OS X Certificate Store

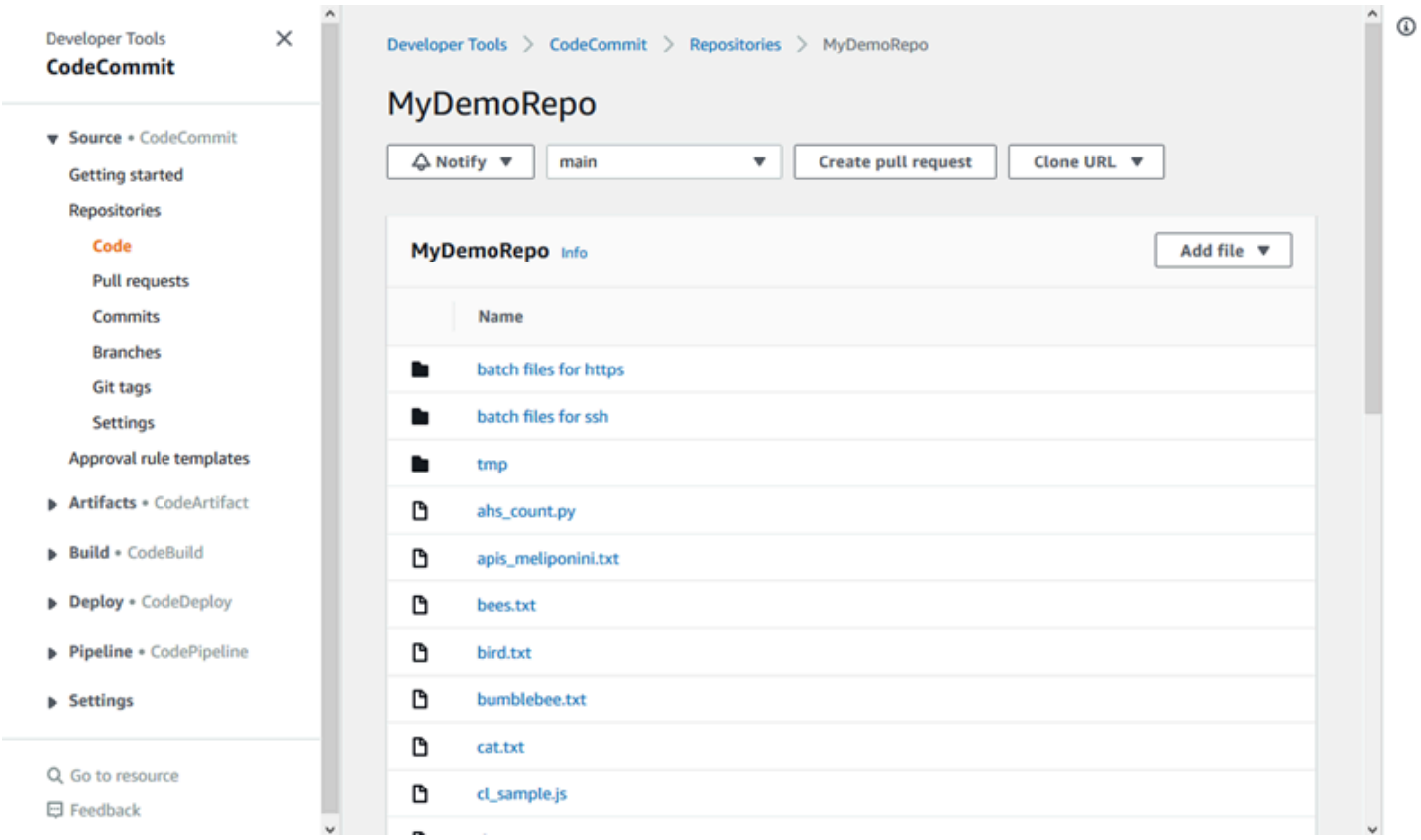
  Wenn Sie den Credential Helper für CodeCommit unter Mac OS X verwenden, kennen Sie wahrscheinlich das Problem mit zwischengespeicherten Anmeldeinformationen. Dieses Skript veranschaulicht eine Lösung.

  Autor: Nico Coetzee

  Veröffentlichung Februar 2016

Arbeiten mit Repositorien in AWS CodeCommit

Ein Repository ist das grundlegende Versionskontrollobjekt in CodeCommit. Hier können Sie Code und Dateien für Ihr Projekt sicher speichern. Zudem speichert es den gesamten Projektverlauf – vom ersten Commit bis zur letzten Änderung. Sie können das Repository mit anderen Benutzern teilen, um gemeinsam an einem Projekt zu arbeiten. Wenn Sie AWS Tags zu Repositorys hinzufügen, können Sie Benachrichtigungen einrichten, sodass Repository-Benutzer E-Mails über Ereignisse erhalten (z. B. wenn ein anderer Benutzer Code kommentiert). Außerdem können Sie die Standardeinstellungen für das Repository ändern, seine Inhalte durchsuchen und vieles mehr. Sie können Auslöser für das Repository erstellen, sodass Code per Push ausgeführt wird oder andere Ereignisse Aktionen wie E-Mails oder Codefunktionen auslösen. Wenn Sie ein Repository auf dem lokalen Computer konfigurieren (ein lokales Repository), können Sie Änderungen per Push sogar an mehrere Repositorys übertragen.



Bevor Sie Änderungen an ein CodeCommit Repository übertragen können, müssen Sie einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto konfigurieren oder den Zugriff für Verbundzugriff oder temporäre Anmeldeinformationen einrichten. Weitere Informationen finden Sie

unter Schritt 1: Erstkonfiguration für CodeCommit und Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit.

Informationen zur Arbeit mit anderen Aspekten Ihres Repositorys finden Sie unter CodeCommit,Arbeiten mit Dateien, Verwenden von Pull-Anforderungen Mit Commits arbeitenArbeiten mit Zweigen, und. Arbeiten mit Benutzereinstellungen Hinweise zur Migration zu finden Sie CodeCommit unterMigration zu CodeCommit.

Themen

- Erstellen Sie ein Repository AWS CodeCommit
- Stellen Sie eine Connect zu einem AWS CodeCommit Repository her
- Teilen Sie ein Repository AWS CodeCommit
- Benachrichtigungen für Ereignisse in einem Repository konfigurieren AWS CodeCommit
- Kennzeichnen von Repositorys in AWS CodeCommit
- Trigger für ein AWS CodeCommit Repository verwalten
- Ein AWS CodeCommit Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben
- CodeCommit Repository-Details anzeigen
- AWS CodeCommit Repository-Einstellungen ändern
- Synchronisieren Sie Änderungen zwischen einem lokalen Repository und einem Repository AWS CodeCommit
- Push Commits in ein zusätzliches Git-Repository
- Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen
- Löschen Sie ein AWS CodeCommit Repository

Erstellen Sie ein Repository AWS CodeCommit

Verwenden Sie die AWS CodeCommit Konsole oder die AWS Command Line Interface (AWS CLI), um ein leeres CodeCommit Repository zu erstellen. Informationen zum Hinzufügen von Tags zu einem von Ihnen erstellten Repository finden Sie unter Hinzufügen eines Tags zu einem Repository.

Diese Anleitungen setzen die Durchführung der Schritte unter Einrichtung voraus.

> **Note**
>
> Je nach Nutzung können Ihnen Gebühren für die Erstellung eines Repositorys oder den Zugriff auf ein Repository in Rechnung gestellt werden. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter Preise.

Themen

- Erstellen Sie ein Repository (Konsole)
- Erstellen Sie ein Repository ()AWS CLI

Erstellen Sie ein Repository (Konsole)

Um ein CodeCommit Repository zu erstellen

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl den AWS-Region Ort aus, an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Wählen Sie auf der Seite Repositorys die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) im Feld Repository name (Repository-Name) einen Namen für das Repository ein.

   > **Note**
   >
   > Bei den Repository-Namen wird nach Groß- und Kleinschreibung unterschieden. Der Name muss AWS-Region für Ihr Amazon Web Services Services-Konto eindeutig sein.

5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung für das Repository ein. Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.

   > **Note**
   >
   > Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie ein Anwendungsentwickler sind, der

die `BatchGetRepositories` APIs `GetRepository` oder verwendet und planen, das Repository-Beschreibungsfeld in einem Webbrowser anzuzeigen, finden Sie weitere Informationen in der CodeCommit API-Referenz.

6. (Optional) Wählen Sie Tag hinzufügen, um Ihrem Repository ein oder mehrere Repository-Tags (eine benutzerdefinierte Attributbezeichnung, mit der Sie Ihre AWS Ressourcen organisieren und verwalten können) hinzuzufügen. Weitere Informationen finden Sie unter Kennzeichnen von Repositorys in AWS CodeCommit.
7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen, vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln von Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorys.
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer ihn analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um Codefehler zu finden und Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.
9. Wählen Sie Erstellen.

Nachdem Sie ein Repository erstellt haben, können Sie eine Verbindung zu diesem herstellen und mit dem Hinzufügen von Code beginnen, entweder über die CodeCommit Konsole oder einen lokalen Git-Client oder indem Sie Ihr CodeCommit Repository in Ihre bevorzugte IDE integrieren. Weitere Informationen finden Sie unter Einrichten für AWS CodeCommit. Sie können das Repository auch zu einer kontinuierlichen Bereitstellungs-Pipeline hinzufügen. Weitere Informationen finden Sie unter Anleitung zum Erstellen einer einfachen Pipeline.

Um Informationen über das neue CodeCommit Repository zu erhalten, z. B. die URLs, die beim Klonen des Repositorys verwendet werden sollen, wählen Sie den Namen des Repositorys aus der Liste aus oder wählen Sie einfach das Verbindungsprotokoll, das Sie neben dem Namen des Repositorys verwenden möchten.

Um dieses Repository für andere Benutzer freizugeben, müssen Sie ihnen den zum Klonen des Repositorys zu verwendenden HTTPS- oder SSH-Link senden. Vergewissern Sie sich, dass sie

über die Berechtigungen verfügen, die zum Zugriff auf das Repository erforderlich sind. Weitere Informationen finden Sie unter Teilen Sie ein Repository und Authentifizierung und Zugriffskontrolle für AWS CodeCommit.

Erstellen Sie ein Repository ()AWS CLI

Sie können den verwenden AWS CLI , um ein CodeCommit Repository zu erstellen. Anders als bei der Konsole können Sie dem Repository Tags hinzufügen, wenn Sie es mithilfe der AWS CLI erstellen.

1. Stellen Sie sicher, dass Sie das AWS CLI mit dem konfiguriert haben AWS-Region , wo das Repository existiert. Um die Region zu verifizieren, geben Sie den folgenden Befehl in das Befehlszeilen- oder Terminalfenster ein. Prüfen Sie dann die Angaben des Standardregionsnamens.

   ```
   aws configure
   ```

   Der Name der Standardregion muss mit dem Namen des AWS-Region Repositorys in übereinstimmen CodeCommit. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.

2. Führen Sie den Befehl create-repository aus und geben Sie Folgendes an:

   - Ein Name, der das CodeCommit Repository eindeutig identifiziert (mit der `--repository-name` Option).

     > **Note**
     >
     > Dieser Name muss in einem Amazon Web Services Services-Konto eindeutig sein.

   - Ein optionaler Kommentar zum CodeCommit Repository (mit der `--repository-description` Option).
   - Ein oder mehrere optionale Schlüssel-Wert-Paare, die als Tags für das CodeCommit Repository verwendet werden sollen (mit der `--tags` Option).
   - Ein optionaler, vom Kunden verwalteter Schlüssel, der beim Verschlüsseln und Entschlüsseln dieses Repositorys verwendet werden kann. Alle Repositorys werden bei der Übertragung und im Ruhezustand mit einem Schlüssel verschlüsselt. AWS KMS Wenn kein Schlüssel angegeben ist, wird der AWS verwaltete Standardschlüssel `aws/codecommit` verwendet.

Verwenden Sie beispielsweise diesen Befehl, um ein CodeCommit Repository `MyDemoRepo` mit der Beschreibung `"My demonstration repository"` und einem Tag mit einem Schlüssel namens *`Team`* mit dem Wert *`Saanvi`* zu erstellen.

```
aws codecommit create-repository --repository-name MyDemoRepo --repository-
description "My demonstration repository" --tags Team=Saanvi
```

> **Note**
>
> Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie ein Anwendungsentwickler sind, der die `BatchGetRepositories` APIs `GetRepository` oder verwendet und planen, das Repository-Beschreibungsfeld in einem Webbrowser anzuzeigen, finden Sie weitere Informationen in der CodeCommit API-Referenz.

3. Ist der Befehl erfolgreich, wird ein `repositoryMetadata`-Objekt mit den folgenden Informationen ausgegeben:

   - Beschreibung (`repositoryDescription`)
   - Eindeutige, systemgenerierte ID (`repositoryId`)
   - Name (`repositoryName`)
   - Die ID des Amazon Web Services Services-Kontos, das dem CodeCommit Repository zugeordnet ist (`accountId`).

   Es folgt eine Beispielausgabe basierend auf dem vorangegangenen Beispielbefehl:

```
{
    "repositoryMetadata": {
        "repositoryName": "MyDemoRepo",
        "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/
repos/MyDemoRepo",
        "lastModifiedDate": 1446071622.494,
        "repositoryDescription": "My demonstration repository",
        "cloneUrlHttp": "https://git-codecommit.us-east-2.amazonaws.com/v1/
repos/MyDemoRepo",
        "defaultBranch": main,
        "kmsKeyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
```

```
        "creationDate": 1446071622.494,
        "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
        "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
        "accountId": "111111111111"
    }
}
```

> **Note**
>
> Tags, die beim Erstellen des Repository hinzugefügt wurden, werden nicht in der Ausgabe zurückgegeben. Um die einem Repository zugewiesenen Tags aufzulisten, führen Sie den Befehl list-tags-for-resource aus.

4. Notieren Sie sich den Namen und die ID des CodeCommit Repositorys. Sie benötigen sie, um Informationen über das CodeCommit Repository zu überwachen und zu ändern, insbesondere wenn Sie es verwenden AWS CLI.

   Wenn Sie den Namen oder die ID vergessen haben, führen Sie die Anweisungen unter CodeCommit Repository-Details anzeigen (AWS CLI) aus.

Nach dem Erstellen eines Repositorys können Sie eine Verbindung mit diesem herstellen und Code hinzufügen. Weitere Informationen finden Sie unter Herstellen einer Verbindung mit einem Repository. Sie können das Repository auch zu einer kontinuierlichen Bereitstellungs-Pipeline hinzufügen. Weitere Informationen finden Sie unter Anleitung zum Erstellen einer einfachen Pipeline.

Stellen Sie eine Connect zu einem AWS CodeCommit Repository her

Wenn Sie zum ersten Mal eine Verbindung zu einem CodeCommit Repository herstellen, klonen Sie in der Regel dessen Inhalt auf Ihren lokalen Computer. Sie können einem Repository auch direkt von der CodeCommit Konsole aus Dateien hinzufügen und Dateien darin bearbeiten. Wenn Sie bereits über ein lokales Repository verfügen, können Sie alternativ ein CodeCommit Repository als Remote-Repository hinzufügen. Dieses Thema enthält Anweisungen zum Herstellen einer Verbindung zu einem CodeCommit Repository. Informationen dazu, wie Sie ein vorhandenes Repository migrieren möchten CodeCommit, finden Sie unterMigration zu CodeCommit.

> **Note**
>
> Je nach Nutzung fallen möglicherweise Gebühren für die Erstellung eines Repositorys oder den Zugriff darauf an. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter Preise.

Themen

- Voraussetzungen für die Verbindung mit einem CodeCommit Repository
- Connect zum CodeCommit Repository her, indem Sie das Repository klonen
- Ein lokales Repo mit dem CodeCommit Repository Connect

Voraussetzungen für die Verbindung mit einem CodeCommit Repository

Bevor Sie ein CodeCommit Repository klonen oder ein lokales Repository mit einem CodeCommit Repository verbinden können:

- Sie müssen Ihren lokalen Computer mit der Software und den Einstellungen konfiguriert haben, die für die Verbindung erforderlich sind. CodeCommit Dazu gehört die Installation und Konfiguration von Git. Weitere Informationen finden Sie unter Einrichtung und Erste Schritte mit Git und AWS CodeCommit.
- Sie müssen die Klon-URL des CodeCommit Repositorys haben, zu dem Sie eine Verbindung herstellen möchten. Weitere Informationen finden Sie unter Repository-Details anzeigen.

  Wenn Sie noch kein CodeCommit Repository erstellt haben, folgen Sie den Anweisungen unterErstellen eines -Repositorys, kopieren Sie die Klon-URL des CodeCommit Repositorys und kehren Sie zu dieser Seite zurück.

  Wenn Sie ein CodeCommit Repository haben, dessen Namen Sie jedoch nicht kennen, folgen Sie den Anweisungen unterRepository-Details anzeigen.
- Sie benötigen einen Speicherort auf Ihrem lokalen Computer, an dem Sie eine lokale Kopie des CodeCommit Repositorys speichern können, zu dem Sie eine Verbindung herstellen. (Diese lokale Kopie des CodeCommit Repositorys wird als lokales Repo bezeichnet.) Sie wechseln dann zu Git-Befehlen, die Sie an diesem Ort ausführen. Sie könnten beispielsweise `/tmp` (für Linux, macOS oder Unix) oder `c:\temp` (für Windows) verwenden, wenn Sie zu Testzwecken einen temporären Klon erstellen. Dies ist der Verzeichnispfad, der in diesen Beispielen verwendet wird.

> **Note**
>
> Sie können jedes beliebige Verzeichnis verwenden. Wenn Sie ein Repository für eine langfristige Nutzung klonen, sollten Sie den Klon aus einem Arbeitsverzeichnis erstellen, das nicht für temporäre Dateien genutzt wird. Bei Verwendung eines anderen Verzeichnisses als `/tmp` oder `c:\temp` müssen Sie das von uns angegebene Verzeichnis in diesen Anweisungen mit diesem Verzeichnis ersetzen.

Connect zum CodeCommit Repository her, indem Sie das Repository klonen

Wenn Sie noch kein lokales Repository haben, folgen Sie den Schritten in diesem Verfahren, um das CodeCommit Repository auf Ihren lokalen Computer zu klonen.

1. Sorgen Sie dafür, dass die Voraussetzungen erfüllt sind, einschließlich Einrichtung .

   > **Important**
   >
   > Wenn Sie die Einrichtung nicht abgeschlossen haben, können Sie weder eine Verbindung herstellen noch das Repository klonen.

2. Verwenden Sie im Verzeichnis `/tmp` oder `c:\temp` Git, um den Befehl clone auszuführen. Die folgenden Beispiele zeigen, wie Sie ein Repository klonen, das *`MyDemoRepo`*in der Region USA Ost (Ohio) benannt ist.

   Für HTTPS mit Git-Anmeldeinformationen oder dem Hilfsprogramm für Anmeldeinformationen, der in der AWS CLI enthalten ist:

   ```
   git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
   ```

   Für die Verwendung von git-remote-codecommit HTTPS wird das Standardprofil vorausgesetzt und AWS-Region konfiguriert in AWS CLI:

   ```
   git clone codecommit://MyDemoRepo my-demo-repo
   ```

Für SSH:

```
git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

In diesem Beispiel `git-codecommit.us-east-2.amazonaws.com` steht der Git-Verbindungspunkt für die Region USA Ost (Ohio), in der das Repository existiert, `MyDemoRepo` für den Namen Ihres CodeCommit Repositorys und `my-demo-repo` für den Namen des Verzeichnisses, das Git in dem `/tmp` Verzeichnis oder dem `c:\temp` Verzeichnis erstellt. Weitere Informationen zur That-Unterstützung CodeCommit und zu den Git-Verbindungen für diese AWS-Regionen finden Sie unterRegionen und Git-Verbindungsendpunkte. AWS-Regionen

> **Note**
>
> Beim Klonen eines Repositorys mit SSH auf Windows-Betriebssystemen müssen Sie gegebenenfalls die SSH-Schlüssel-ID wie folgt der Verbindungszeichenfolge hinzufügen:
>
> ```
> git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
> ```
>
> Weitere Informationen finden Sie unter Für SSH-Verbindungen unter Windows und Fehlerbehebung.

Nachdem Git das Verzeichnis erstellt hat, wird eine Kopie Ihres CodeCommit Repositorys in das neu erstellte Verzeichnis heruntergeladen.

Wenn das CodeCommit Repository neu oder anderweitig leer ist, siehst du eine Meldung, dass du ein leeres Repository klonst. Das ist normal.

> **Note**
>
> Wenn Sie eine Fehlermeldung erhalten, dass Git das CodeCommit Repository nicht finden kann oder dass Sie nicht berechtigt sind, eine Verbindung zum CodeCommit Repository herzustellen, stellen Sie sicher, dass Sie die Voraussetzungen erfüllt haben, einschließlich der Zuweisung von Berechtigungen für den IAM-Benutzer und der

Einrichtung Ihrer IAM-Benutzeranmeldeinformationen für Git und CodeCommit auf dem lokalen Computer. Vergewissern Sie sich auch, dass der Repository-Name stimmt.

Nachdem du dein lokales Repo erfolgreich mit deinem CodeCommit Repository verbunden hast, kannst du jetzt damit beginnen, Git-Befehle vom lokalen Repo aus auszuführen, um Commits, Branches und Tags zu erstellen und zum Repository zu pushen und daraus zu ziehen. CodeCommit

Ein lokales Repo mit dem CodeCommit Repository Connect

Gehen Sie wie folgt vor, wenn Sie bereits über ein lokales Repository verfügen und ein CodeCommit Repository als Remote-Repository hinzufügen möchten. Wenn du bereits ein Remote-Repository hast und deine Commits in dieses andere Remote-Repository übertragen möchtest, befolge die Schritte unter. CodeCommit Senden Sie Commits an zwei Repositorys

1. Sorgen Sie dafür, dass die Voraussetzungen erfüllt sind.
2. Wechseln Sie von der Befehlszeile oder dem Terminal aus zu Ihrem lokalen Repo-Verzeichnis und führen Sie den git remote add Befehl aus, um das CodeCommit Repository als Remote-Repository für Ihr lokales Repo hinzuzufügen.

   Mit dem folgenden Befehl wird beispielsweise das Remote-Objekt mit dem Spitznamen https://git-codecommit.us-east-2.amazonaws.com/v1/repos/ **`origin`** hinzugefügt: MyDemoRepo

   Für HTTPS:

   ```
   git remote add origin https://git-codecommit.us-east-2.amazonaws.com/v1/repos/
   MyDemoRepo
   ```

   Für SSH:

   ```
   git remote add origin ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/
   MyDemoRepo
   ```

   Mit diesem Befehl wird kein Inhalt zurückgegeben.
3. Um zu überprüfen, ob Sie das CodeCommit Repository als Remote-Repository für Ihr lokales Repository hinzugefügt haben, führen Sie den git remote -v Befehl aus, der eine Ausgabe ähnlich der folgenden erzeugen sollte:

Für HTTPS:

```
origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)

origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
```

Für SSH:

```
origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)

origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
```

Nachdem Sie Ihr lokales Repo erfolgreich mit Ihrem CodeCommit Repository verbunden haben, können Sie damit beginnen, Git-Befehle vom lokalen Repo aus auszuführen, um Commits, Branches und Tags zu erstellen und Push zum Repository zu senden und aus dem Repository zu ziehen. CodeCommit

Teilen Sie ein Repository AWS CodeCommit

Nachdem Sie ein CodeCommit Repository erstellt haben, können Sie es für andere Benutzer freigeben. Stellen Sie zunächst fest, ob Sie beim Zugriff CodeCommit einen Verbundzugriff, temporäre Anmeldeinformationen oder einen Web-Identitätsanbieter wie IAM Identity Center verwenden oder ob Sie Git-Anmeldeinformationen oder SSH-Schlüsselpaare mit IAM-Benutzern verwenden möchten. Wenn Sie ersteres verwenden, müssen Sie Benutzer, Zugriff und Berechtigungen für Ihren Identitätsanbieter einrichten und dann Anweisungen zur Verwendung durch Ihre Benutzer bereitstellen. git-remote-codecommit Weitere Informationen finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit und Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen.

Sie können Git-Anmeldeinformationen oder SSH-Schlüsselpaare nicht mit Verbundzugriffs- oder Identitätsanbietern verwenden, aber viele IDEs funktionieren am besten mit diesen Anmeldeinformationen. Entscheiden Sie in diesem Fall, welches Protokoll (HTTPS oder SSH) Sie Benutzern empfehlen möchten, wenn Sie klonen und einen Git-Client oder eine IDE verwenden, um eine Verbindung zu Ihrem Repository herzustellen. Senden Sie dann die URL- und Verbindungsinformationen an die Benutzer, für die Sie das Repository freigeben möchten. Abhängig

von Ihren Sicherheitsanforderungen kann es für die gemeinsame Nutzung eines Repositorys auch erforderlich sein, eine IAM-Gruppe zu erstellen, verwaltete Richtlinien auf diese Gruppe anzuwenden und IAM-Richtlinien zu bearbeiten, um den Zugriff zu verfeinern, oder IAM-Rollen zu erstellen und zu verwenden.

> **Note**
>
> Nachdem Sie Benutzern Konsolenzugriff auf das Repository gewährt haben, können sie Dateien direkt in der Konsole hinzufügen oder bearbeiten, ohne einen Git-Client einrichten oder eine andere Verbindung herstellen zu müssen. Weitere Informationen finden Sie unter Erstellen oder Hinzufügen einer Datei zu einemAWS CodeCommitEndlager und Bearbeiten Sie den Inhalt einer Datei in einemAWS CodeCommit Repository.

Für diese Anweisungen wird davon ausgegangen, dass Sie bereits die Schritte unter Einrichtung und Erstellen eines -Repositorys ausgeführt haben.

> **Note**
>
> Je nach Nutzung fallen möglicherweise Gebühren für die Erstellung oder den Zugriff auf ein Repository an. Weitere Informationen finden Sie auf der CodeCommit Produktinformationsseite unter Preise.

Themen

- Wählen Sie das Verbindungsprotokoll, das Sie mit Ihren Benutzern teilen möchten
- Erstelle IAM-Richtlinien für dein Repository
- Erstellen Sie eine IAM-Gruppe für Repository-Benutzer
- Teilen Sie die Verbindungsinformationen mit Ihren Benutzern

Wählen Sie das Verbindungsprotokoll, das Sie mit Ihren Benutzern teilen möchten

Wenn Sie ein Repository in erstellen CodeCommit, werden zwei Endpunkte generiert: einer für HTTPS-Verbindungen und einer für SSH-Verbindungen. Beide bieten sichere Verbindungen über

ein Netzwerk. Ihre Benutzer können beide Protokolle verwenden. Beide Endpunkte bleiben aktiv, unabhängig davon, welches Protokoll Sie Ihren Benutzern empfehlen.

HTTPS-Verbindungen erfordern:

- Git-Anmeldeinformationen, die IAM-Benutzer in IAM für sich selbst generieren können. Git-Anmeldeinformationen stellen die für Benutzer Ihres Repositorys am einfachsten einzurichtende und zu nutzende Methode dar.
- Ein AWS Zugriffsschlüssel oder eine Rolle, die Sie annehmen müssen und die Ihre Repository-Benutzer in ihrem Anmeldeinformationsprofil konfigurieren müssen. Sie können git-remote-codecommit konfigurieren (empfohlen) oder das Hilfsprogramm für Anmeldeinformationen, der in der AWS CLI enthalten ist, verwenden. Diese stellen die einzige mögliche Methode für Root-Konto- oder verbundene Benutzer dar.

Für SSH-Verbindungen müssen Ihre Benutzer Folgendes beachten:

- Generieren Sie ein Schlüsselpaar aus öffentlichem und privatem Schlüssel.
- Speichern Sie den öffentlichen Schlüssel.
- Ordnen Sie den öffentlichen Schlüssel ihrem IAM-Benutzer zu.
- Konfigurieren Sie bekannte Host-Dateien auf dem lokalen Computer.
- Erstellen und verwalten Sie eine Konfigurationsdatei auf dem lokalen Computer.

Da dies ein komplexerer Konfigurationsprozess ist, empfehlen wir, dass Sie HTTPS- und Git-Anmeldeinformationen für Verbindungen zu wählen CodeCommit.

Weitere Informationen über HTTPS, SSH, Git, git-remote-codecommit und Remote-Repositorys finden Sie unter Einrichtung , Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen oder in der Git-Dokumentation. Eine allgemeine Übersicht über Kommunikationsprotokolle und Informationen darüber, wie diese jeweils mit Remote-Repositorys kommunizieren, finden Sie unter Git auf dem Server - Die Protokolle.

> **Note**
>
> Git unterstützt zwar eine Vielzahl von Verbindungsprotokollen, unterstützt CodeCommit jedoch keine Verbindungen mit ungesicherten Protokollen wie dem lokalen Protokoll oder generischem HTTP.

Erstelle IAM-Richtlinien für dein Repository

AWS bietet drei verwaltete Richtlinien in IAM für. CodeCommit Diese Richtlinien können nicht bearbeitet werden und gelten für alle Repositorys, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind. Sie können diese Richtlinien jedoch als Vorlagen verwenden, um eigene kundenverwaltete Richtlinien zu erstellen, die nur für das Repository gelten, das Sie freigeben möchten. Ihre kundenverwaltete Richtlinie kann speziell für das Repository gelten, das Sie freigeben möchten. Weitere Informationen finden Sie unter Verwaltete Richtlinien und IAM-Benutzer und -Gruppen.

> **Tip**
>
> Für eine genauere Steuerung des Zugriffs auf Ihr Repository können Sie mehr als eine vom Kunden verwaltete Richtlinie erstellen und die Richtlinien auf verschiedene IAM-Benutzer und -Gruppen anwenden.

Weitere Informationen zum Überprüfen der Inhalte von verwalteten Richtlinien und Verwenden von Richtlinien zum Erstellen und Anwenden von Berechtigungen finden Sie unter Authentifizierung und Zugriffskontrolle für AWS CodeCommit.

Erstellen einer kundenverwalteten Richtlinie für das Repository

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Dashboard die Optionen Policies und dann Create Policy aus.
3. Wählen Sie auf der Seite „Richtlinie erstellen“ die Option Verwaltete Richtlinie importieren aus.
4. Geben Sie auf der Seite Verwaltete Richtlinien importieren im Feld Richtlinien filtern den Text ein**`AWSCodeCommitPowerUser`**. Klicken Sie auf die Schaltfläche neben dem Richtliniennamen und wählen Sie dann Importieren aus.
5. Wählen Sie auf der Seite Create policy (Richtlinie erstellen) die Option JSON aus. Ersetzen Sie den Teil „*“ in der `Resource` Zeile für CodeCommit Aktionen durch den Amazon-Ressourcennamen (ARN) des CodeCommit Repositorys, wie hier gezeigt:

```
"Resource": [
 "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"
 ]
```

> **Tip**
>
> Um den ARN für das CodeCommit Repository zu finden, gehen Sie zur CodeCommit Konsole, wählen Sie den Repository-Namen aus der Liste aus und wählen Sie dann Einstellungen. Weitere Informationen finden Sie unter Repository-Details anzeigen.

Wenn Sie diese Richtlinie auf mehr als ein Repository anwenden möchten, fügen Sie jedes Repository als Ressource hinzu, indem Sie seinen ARN angeben. Fügen Sie zwischen den Ressourcenanweisungen ein Komma ein, wie hier dargestellt:

```
"Resource": [
 "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
 "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo"
 ]
```

Wenn Sie mit der Bearbeitung fertig sind, wählen Sie „Richtlinie überprüfen“.

6. Geben Sie auf der Seite „Richtlinie überprüfen“ im Feld Name einen neuen Namen für die Richtlinie ein (z. B. *AWSCodeCommitPowerUser- MyDemoRepo*). Geben Sie optional eine Beschreibung für diese Richtlinie ein.
7. Wählen Sie Richtlinie erstellen aus.

Erstellen Sie eine IAM-Gruppe für Repository-Benutzer

Um den Zugriff auf Ihr Repository zu verwalten, erstellen Sie eine IAM-Gruppe für deren Benutzer, fügen Sie dieser Gruppe IAM-Benutzer hinzu und fügen Sie dann die vom Kunden verwaltete Richtlinie hinzu, die Sie im vorherigen Schritt erstellt haben. Alternativ können Sie eine Rolle mit einer angehängten, vom Kunden verwalteten Richtlinie erstellen und Benutzer diese Rolle übernehmen lassen.

Wenn Sie SSH verwenden, müssen Sie der Gruppe IAMuserSSHKeys eine weitere verwaltete Richtlinie hinzufügen, die von IAM verwaltete Richtlinie, die es Benutzern ermöglicht, ihren öffentlichen SSH-Schlüssel hochzuladen und ihn dem IAM-Benutzer zuzuordnen, mit dem sie sich verbinden. CodeCommit

1. Melden Sie sich bei der an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/ AWS Management Console .

2. Wählen Sie im Navigationsbereich Dashboard die Option Groups und dann Create New Group aus.
3. Geben Sie auf der Seite Gruppennamen festlegen im Feld Gruppenname einen Namen für die Gruppe ein (z. B. *MyDemoRepoGroup*), und wählen Sie dann Next Step aus. Sie sollten den Repository-Namen als Teil des Gruppennamens einfügen.

   > **Note**
   >
   > Dieser Name muss in einem Amazon Web Services Services-Konto eindeutig sein.

4. Markieren Sie das Kästchen neben der vom Kunden verwalteten Richtlinie, die Sie im vorherigen Abschnitt erstellt haben (z. B. AWSCodeCommitPowerUser- MyDemoRepo).
5. Klicken Sie auf der Seite Review auf Create Group. IAM erstellt diese Gruppe mit den angegebenen Richtlinien, die bereits angehängt sind. Die Gruppe erscheint in der Liste der Gruppen, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind.
6. Wählen Sie Ihre Gruppe in der Liste aus.
7. Wählen Sie auf der Gruppenübersichtsseite die Registerkarte Users und anschließend die Option Add Users to Groups aus. Markieren Sie in der Liste, in der alle mit Ihrem Amazon Web Services Services-Konto verknüpften Benutzer angezeigt werden, die Kästchen neben den Benutzern, denen Sie Zugriff auf das CodeCommit Repository gewähren möchten, und wählen Sie dann Benutzer hinzufügen.

   > **Tip**
   >
   > Sie können über das Suchfeld schnell nach Benutzern anhand ihres Namens suchen.

8. Wenn Sie Ihre Benutzer hinzugefügt haben, schließen Sie die IAM-Konsole.

Teilen Sie die Verbindungsinformationen mit Ihren Benutzern

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositorys sind spezifisch für ein. AWS-Region Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.

3. Wählen Sie auf der Seite Repositories (Repositorys) das Repository aus, das Sie freigeben möchten.
4. Wählen Sie unter Clone URL (Klon-URL) das Protokoll aus, das die Benutzer verwenden sollen. Dadurch wird die Klon-URL für das Verbindungsprotokoll kopiert.
5. Senden Sie Ihren Benutzern die Klon-URL zusammen mit allen anderen Anweisungen, z. B. zur Installation von AWS CLI, zur Konfiguration eines Profils oder zur Installation von Git. Stellen Sie sicher, dass Sie die Konfigurationsinformationen für das Verbindungsprotokoll (z. B. HTTPS) angeben.

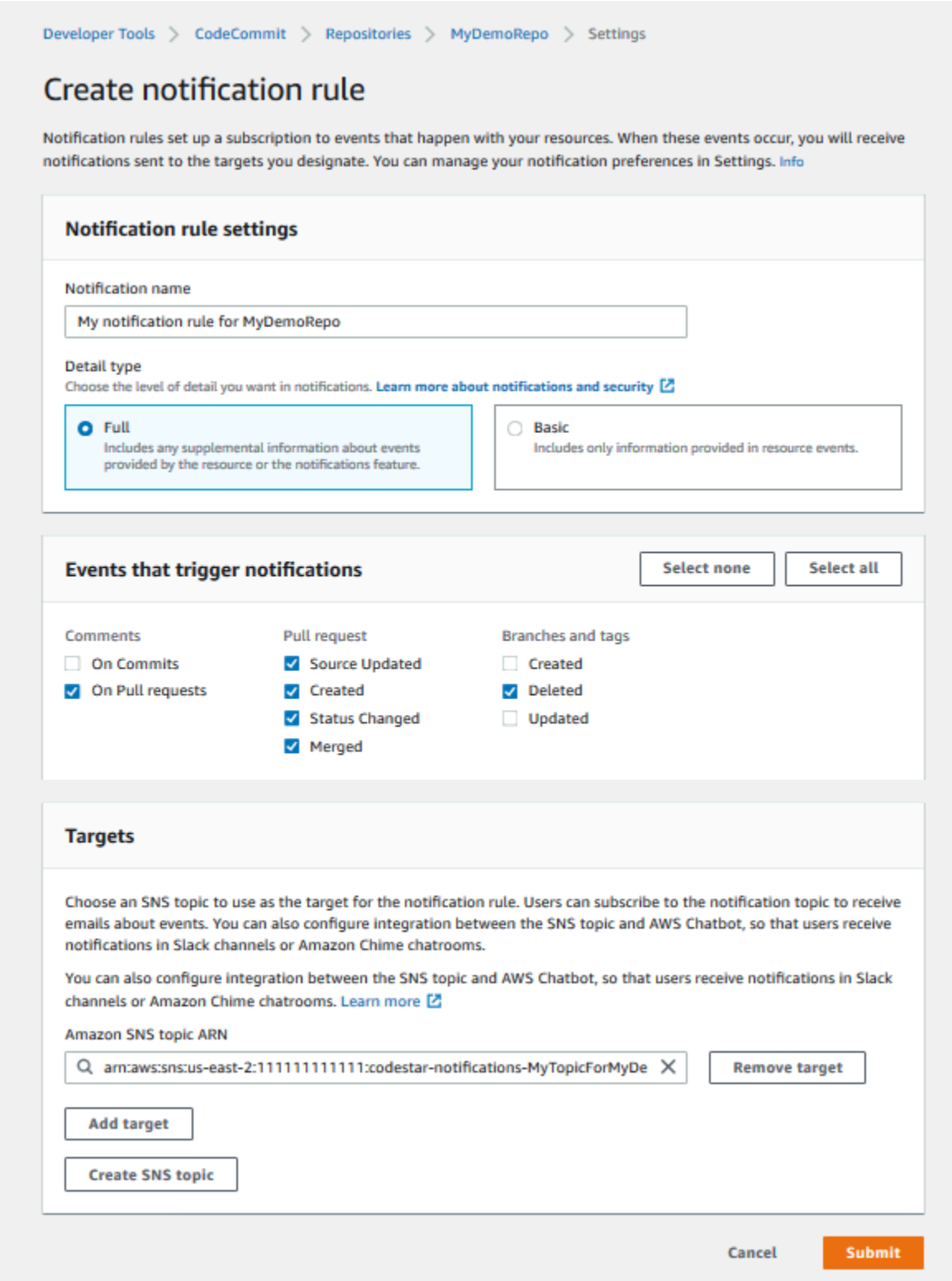
Die folgende Beispiel-E-Mail enthält Informationen für Benutzer, die sich mit dem HTTPS-Verbindungsprotokoll und Git-Anmeldeinformationen in der Region USA Ost (Ohio) (us-east-2) mit dem MyDemoRepo Repository verbinden. Diese E-Mail setzt voraus, dass der Benutzer Git bereits installiert hat und mit der Verwendung vertraut ist.

```
I've created a CodeCommit repository for us to use while working on our project.
The name of the repository is MyDemoRepo, and
it is in the  US East (Ohio) (us-east-2) region.
Here's what you need to do in order to get started using it:

1. Make sure that your version of Git on your local computer is 1.7.9 or later.
2. Generate Git credentials for your IAM user by signing into the IAM console
 here: https://console.aws.amazon.com/iam/  .
Switch to the Security credentials tab for your IAM user and choose the Generate button
 in HTTPS Git credentials for CodeCommit.
Make sure to save your credentials in a secure location!
3. Switch to a directory of your choice and clone the CodeCommit repository to your
 local machine by running the following command:
    git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-
demo-repo
4. When prompted for user name and password, use the Git credentials you just saved.


That's it!  If you'd like to learn more about using CodeCommit, you can start with the
 tutorial here.
```

Umfassende Einrichtungsanweisungen finden Sie unter Einrichtung .

Benachrichtigungen für Ereignisse in einem Repository konfigurieren AWS CodeCommit

Sie können Benachrichtigungsregeln für ein Repository einrichten, sodass Repository-Benutzer E-Mails zu den von Ihnen angegebenen Repository-Ereignistypen erhalten. Es werden Benachrichtigungen gesendet, wenn Ereignisse mit den Einstellungen der Benachrichtigungsregel übereinstimmen. Sie können ein Amazon SNS SNS-Thema für Benachrichtigungen erstellen oder ein vorhandenes Thema in Ihrem Amazon Web Services Services-Konto verwenden. Sie können die CodeCommit Konsole und die verwenden AWS CLI , um Benachrichtigungsregeln zu konfigurieren.



Themen

- Verwenden von Repository-Benachrichtigungsregeln
- Erstellen einer Benachrichtigungsregel

- Benachrichtigungen ändern oder deaktivieren
- Benachrichtigungen löschen

Verwenden von Repository-Benachrichtigungsregeln

Durch das Konfigurieren von Benachrichtigungsregeln werden Ihre Repository-Benutzer unterstützt, da Benutzer E-Mails erhalten, wenn eine Aktion ausgeführt wird, die sich auf einen anderen Benutzer auswirkt. Sie können Benachrichtigungsregeln beispielsweise so konfigurieren, dass Benachrichtigungen gesendet werden, wenn Kommentare zu Commits erstellt werden. Wenn in dieser Konfiguration ein Repository-Benutzer eine Codezeile in einem Commit kommentiert, werden andere Repository-Benutzer per E-Mail benachrichtigt. Sie können sich anmelden und den Kommentar ansehen. Mit den Antworten auf Kommentare werden ebenfalls E-Mails generiert, sodass Repository-Benutzer informiert bleiben.

Benachrichtigungsregeln unterscheiden sich von Repository-Triggern und sie unterscheiden sich auch von den Benachrichtigungen, die Sie vor dem 5. November 2019 in der CodeCommit Konsole konfigurieren konnten.

- Sie können zwar einen Trigger so konfigurieren, dass er Amazon SNS verwendet, um E-Mails über bestimmte Repository-Ereignisse zu senden, aber diese Ereignisse sind auf betriebliche Ereignisse beschränkt, z. B. das Erstellen von Branches und das Übertragen von Code an einen Branch. Trigger verwenden keine CloudWatch Ereignisregeln zur Auswertung von Repository-Ereignissen. Sie sind begrenzt. Weitere Informationen zur Verwendung von Auslösern finden Sie unter Verwalten von Auslösern für ein Repository.
- Für Benachrichtigungen, die vor dem 5. November 2019 konfiguriert wurden, waren weniger Ereignistypen verfügbar und sie konnten nicht für die Integration mit Amazon Chime Chime-Chatrooms oder Slack-Kanälen konfiguriert werden. Sie können weiterhin Benachrichtigungen verwenden, die vor dem 5. November 2019 konfiguriert wurden. Sie können jedoch keine Benachrichtigungen dieses Typs erstellen. Erstellen und verwenden Sie stattdessen Benachrichtigungsregeln. Wir empfehlen, Benachrichtigungsregeln zu verwenden und Benachrichtigungen, die vor dem 5. November 2019 erstellt wurden, zu deaktivieren oder zu löschen. Weitere Informationen finden Sie unter Erstellen einer Benachrichtigungsregel und Benachrichtigungen löschen.

Erstellen einer Benachrichtigungsregel

Sie können Benachrichtigungsregeln verwenden, um Benutzer über wichtige Änderungen zu informieren, z. B. wenn eine Pull-Anfrage in einem Repository erstellt wird. Die Benachrichtigungsregeln spezifizieren sowohl die Ereignisse als auch das Amazon SNS SNS-Thema, das zum Senden von Benachrichtigungen verwendet wird. Weitere Informationen finden Sie unter Was sind Benachrichtigungen?.

> **Note**
>
> Diese Funktion ist in der Region Europa (Mailand) nicht verfügbar. Informationen zur Konfiguration von Benachrichtigungen in dem in dieser Region verfügbaren Erlebnis finden Sie unter Repository-Benachrichtigungen konfigurieren.

Sie können die Konsole oder die verwenden AWS CLI , um Benachrichtigungsregeln für zu erstellen AWS CodeCommit.

So erstellen Sie eine Benachrichtigungsregel (Konsole):

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codecommit/.
2. Wählen Sie Repositories (Repositorys) und dann ein Repository aus, zu dem Sie Benachrichtigungsregeln hinzufügen möchten.
3. Wählen Sie auf der Repository-Seite Notify (Benachrichtigen) und dann Create notification rule (Benachrichtigungsregel erstellen). Sie können auch zur Seite Settings (Einstellungen) für das Repository navigieren und Create notification rule (Benachrichtigungsregel erstellen) wählen.
4. Geben Sie unter Notification name (Benachrichtigungsname) einen Namen für die Regel ein.
5. Wählen Sie unter Detailtyp die Option Basic aus, wenn Sie möchten, dass nur die Informationen, die Amazon zur Verfügung gestellt wurden, in der Benachrichtigung EventBridge enthalten sind. Wählen Sie Vollständig, wenn Sie Informationen, die Amazon zur Verfügung gestellt wurden, EventBridge und Informationen, die möglicherweise vom CodeCommit oder vom Notification Manager bereitgestellt wurden, einbeziehen möchten.

   Weitere Informationen finden Sie unter Informationen zu Inhalten und Sicherheit von Benachrichtigungen.

6. Wählen Sie unter Events that trigger notifications (Ereignisse, die Benachrichtigungen auslösen) die Ereignisse aus, für die Sie Benachrichtigungen senden möchten. Weitere Informationen finden Sie unter Ereignisse für Benachrichtigungsregeln in Repositories.
7. Führen Sie unter Targets (Ziele) einen der folgenden Schritte aus:

   - Wenn Sie bereits eine Ressource zur Verwendung mit Benachrichtigungen konfiguriert haben, wählen Sie unter Choose target type (Zieltyp wählen) entweder AWS Chatbot (Slack) oder SNS topic (SNS-Thema)aus. Wählen Sie unter Ziel auswählen den Namen des Clients (für einen in konfigurierten Slack-Client AWS Chatbot) oder den Amazon-Ressourcennamen (ARN) des Amazon SNS-Themas (für Amazon SNS SNS-Themen, die bereits mit der für Benachrichtigungen erforderlichen Richtlinie konfiguriert wurden).
   - Wenn Sie keine Ressource für die Verwendung mit Benachrichtigungen konfiguriert haben, wählen Sie Create target (Ziel erstellen) und dann SNS topic (SNS-Thema) aus. Geben Sie nach codestar-notifications- einen Namen für das Thema an und wählen Sie dann Create (Erstellen).

   > **Note**
   >
   > - Wenn Sie das Amazon-SNS-Thema im Rahmen des Erstellens der Benachrichtigungsregel erstellen, wird die Richtlinie, die es ermöglicht, Ereignisse in dem Thema zu veröffentlichen, für Sie angewendet. Durch die Verwendung eines Themas, das für Benachrichtigungsregeln erstellt wurde, kann sichergestellt werden, dass Sie das Thema nur für die Benutzer abonnieren, die Benachrichtigungen zu dieser Ressource erhalten sollen.
   > - Sie können im Rahmen der Erstellung einer AWS Chatbot Benachrichtigungsregel keinen Client erstellen. Wenn du AWS Chatbot (Slack) auswählst, wird dir eine Schaltfläche angezeigt, in der du einen Client konfigurieren kannst. AWS Chatbot Wenn Sie diese Option wählen, wird die AWS Chatbot Konsole geöffnet. Weitere Informationen finden Sie unter Integrationen zwischen Benachrichtigungen konfigurieren und AWS Chatbot.
   > - Wenn Sie ein vorhandenes Amazon SNS SNS-Thema als Ziel verwenden möchten, müssen Sie die erforderliche Richtlinie für AWS CodeStar Benachrichtigungen zusätzlich zu allen anderen Richtlinien hinzufügen, die möglicherweise für dieses Thema existieren. Weitere Informationen finden Sie unter Konfigurieren vorhandener

Amazon SNS-Themen für Benachrichtigungen und Informationen zu Inhalten und Sicherheit von Benachrichtigungen.

8. Um die Erstellung der Regel abzuschließen, wählen Sie Submit (Absenden) aus.
9. Sie müssen das Amazon SNS SNS-Thema für die Regel abonnieren, bevor sie Benachrichtigungen erhalten können. Weitere Informationen finden Sie unter Amazon SNS SNS-Themen, die Ziele sind für Benutzer abonnieren. Sie können auch die Integration zwischen Benachrichtigungen einrichten und Benachrichtigungen AWS Chatbot an Amazon Chime Chime-Chatrooms senden. Weitere Informationen finden Sie unter Konfiguration der Integration zwischen Benachrichtigungen und. AWS Chatbot

So erstellen Sie eine Benachrichtigungsregel (AWS CLI):

1. Führen Sie in einem Terminal oder einer Eingabeaufforderung den Befehl create-notification rule aus, um das JSON-Skelett zu generieren:

```
aws codestar-notifications create-notification-rule --generate-cli-skeleton
 > rule.json
```

Sie können die Datei beliebig benennen. In diesem Beispiel trägt die Datei den Namen *rule.json*.

2. Öffnen Sie die JSON-Datei in einem Texteditor, und bearbeiten Sie sie so, dass sie die Ressource, die Ereignistypen und das gewünschte Ziel für die Regel enthält. Das folgende Beispiel zeigt eine Benachrichtigungsregel, die **MyNotificationRule** nach einem Repository benannt ist, das *MyDemoRepo*in einem AWS Konto mit der ID *123456789012* benannt ist. Benachrichtigungen mit dem vollständigen Detailtyp werden an ein Amazon SNS SNS-Thema mit dem Namen gesendet *MyNotificationTopic*, wenn Branches und Tags erstellt werden:

```
{
    "Name": "MyNotificationRule",
    "EventTypeIds": [
        "codecommit-repository-branches-and-tags-created"
    ],
    "Resource": "arn:aws:codecommit:us-east-1:123456789012:MyDemoRepo",
    "Targets": [
        {
            "TargetType": "SNS",
```

```
            "TargetAddress": "arn:aws:sns:us-
east-1:123456789012:MyNotificationTopic"
        }
    ],
    "Status": "ENABLED",
    "DetailType": "FULL"
}
```

Speichern Sie die Datei.

3. Führen Sie unter Verwendung der soeben bearbeiteten Datei am Terminal oder in der Befehlszeile erneut den Befehl create-notification-rule aus, um die Benachrichtigungsregel zu erstellen:

```
aws codestar-notifications create-notification-rule --cli-input-json
 file://rule.json
```

4. Bei Erfolg gibt der Befehl den ARN der Benachrichtigungsregel zurück, der ähnlich wie im Folgenden dargestellt aussieht:

```
{
    "Arn": "arn:aws:codestar-notifications:us-east-1:123456789012:notificationrule/
dc82df7a-EXAMPLE"
}
```

Benachrichtigungen ändern oder deaktivieren

Sie können die AWS CodeCommit Konsole verwenden, um die Konfiguration von Benachrichtigungen zu ändern, die vor dem 5. November 2019 erstellt wurden, einschließlich der Ereignistypen, mit denen E-Mails an Benutzer gesendet werden, und des Amazon SNS SNS-Themas, das zum Senden von E-Mails über das Repository verwendet wird. Sie können die CodeCommit Konsole auch verwenden, um die Liste der E-Mail-Adressen und Endpunkte zu verwalten, die das Thema abonniert haben, oder um Benachrichtigungen zu deaktivieren.

So ändern Sie Benachrichtigungseinstellungen

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.

2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys, in dem Sie Benachrichtigungen konfigurieren möchten, die vor dem 5. November 2019 erstellt wurden.
3. Wählen Sie im Navigationsbereich Settings und dann Notifications aus. Wenn ein Banner angezeigt wird, das Sie darüber informiert, dass Sie anstelle von Benachrichtigungsregeln über Benachrichtigungen verfügen, wählen Sie Manage existing notifications (Vorhandene Benachrichtigungen verwalten).
4. Wählen Sie Bearbeiten aus.
5. Nehmen Sie die gewünschten Änderungen vor und wählen Sie dann Save.

Das Deaktivieren von Benachrichtigungen ist eine einfache Methode, den Empfang von E-Mails zu Repository-Ereignissen durch Benutzer vorübergehend zu verhindern.

Um eine Benachrichtigung, die vor dem 5. November 2019 erstellt wurde, dauerhaft zu löschen, führen Sie die unter Benachrichtigungen löschen beschriebenen Schritte aus.

So deaktivieren Sie Benachrichtigungen

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Benachrichtigungen deaktivieren möchten.
3. Wählen Sie im Navigationsbereich Settings und dann Notifications aus. Wählen Sie Manage existing notifications (Vorhandene Benachrichtigungen verwalten) aus.
4. Wählen Sie Edit (Bearbeiten) und deaktivieren Sie unter Event status (Ereignisstatus) die Option Enable notifications (Benachrichtigungen aktivieren) mithilfe des Schiebereglers. Wählen Sie Speichern.
5. Der Ereignisstatus ändert sich in Disabled (Deaktiviert). Es werden keine E-Mails zu Ereignissen gesendet. Wenn Sie Benachrichtigungen deaktivieren, wird die CloudWatch Ereignisregel für das Repository automatisch deaktiviert. Ändern Sie ihren Status in der CloudWatch Ereigniskonsole nicht manuell.

Benachrichtigungen löschen

Wenn Sie Benachrichtigungen, die vor dem 5. November 2019 für ein Repository erstellt wurden, nicht mehr verwenden möchten, können Sie die mit der Benachrichtigung verknüpfte

Amazon CloudWatch Events-Regel löschen. Dadurch wird die Benachrichtigung automatisch gelöscht. Es werden keine Abonnements oder das Amazon SNS SNS-Thema gelöscht, das für Benachrichtigungen verwendet wird.

> **Note**
>
> Wenn Sie den Namen eines Repositorys über die Konsole ändern, funktionieren Benachrichtigungen, die vor dem 5. November 2019 erstellt wurden, weiterhin ohne Änderung. Wenn Sie den Namen Ihres Repositorys jedoch über die Befehlszeile oder über die API ändern, funktionieren Benachrichtigungen nicht mehr. Die einfachste Methode zum Wiederherstellen von Benachrichtigungen ist, die Benachrichtigungseinstellungen zu löschen und sie dann erneut zu konfigurieren.

So löschen Sie Benachrichtigungseinstellungen

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, aus dem Benachrichtigungen entfernt werden sollen, die vor dem 5. November 2019 erstellt wurden.
3. Wählen Sie im Navigationsbereich Settings und dann Notifications aus. Wenn ein Banner angezeigt wird, das Sie darüber informiert, dass Sie anstelle von Benachrichtigungsregeln über Benachrichtigungen verfügen, wählen Sie Manage existing notifications (Vorhandene Benachrichtigungen verwalten).
4. Kopieren Sie im Feld CloudWatch Event-Regel den Namen der Regel, die für die Benachrichtigung erstellt wurde.
5. Melden Sie sich bei der an AWS Management Console und öffnen Sie die CloudWatch Konsole unter https://console.aws.amazon.com/cloudwatch/.
6. Wählen Sie unter Events (Ereignisse) die Option Rules (Regeln). Fügen Sie unter Name den Namen der für die Benachrichtigung erstellten Regel ein. Wählen Sie die Regel und anschließend unter Actions (Aktionen) die Option Delete (Löschen).
7. (Optional) Um das für Benachrichtigungen verwendete Amazon SNS SNS-Thema zu ändern oder zu löschen, nachdem Sie die Benachrichtigungseinstellungen gelöscht haben, rufen Sie die Amazon SNS SNS-Konsole unter https://console.aws.amazon.com/sns/v3/home auf. Weitere Informationen finden Sie unter Clean Up im Amazon Simple Notification Service Developer Guide.

Kennzeichnen von Repositorys in AWS CodeCommit

Ein Tag ist eine benutzerdefinierte Attributbezeichnung, die Sie einer AWS Ressource AWS zuweisen oder zuweisen. AWS Tags unterscheiden sich von Git-Tags, die auf Commits angewendet werden können. Jedes AWS Tag besteht aus zwei Teilen:

- einem Tag-Schlüssel (z. B. `CostCenter`, `Environment`, `Project` oder `Secret`). Bei Tag-Schlüsseln wird zwischen Groß- und Kleinschreibung unterschieden.
- einem optionalen Feld, das als Tag-Wert bezeichnet wird (z. B. `111122223333`, `Production` oder ein Team-Name). Ein nicht angegebener Tag-Wert entspricht einer leeren Zeichenfolge. Wie bei Tag-Schlüsseln wird auch bei Tag-Werten zwischen Groß- und Kleinschreibung unterschieden.

Zusammen werden sie als Schlüssel-Wert-Paare bezeichnet. Informationen zu den Limits hinsichtlich der Anzahl von Tags für ein Repository und zu Einschränkungen bei Tag-Schlüsseln und -Werten finden Sie unter Limits.

Mithilfe von Tags können Sie Ihre AWS Ressourcen identifizieren und organisieren. Viele AWS Dienste unterstützen Tagging, sodass Sie Ressourcen aus verschiedenen Diensten dasselbe Tag zuweisen können, um anzuzeigen, dass die Ressourcen miteinander verknüpft sind. Sie können beispielsweise einem CodeCommit Repository dasselbe Tag zuweisen, das Sie einem Amazon S3 S3-Bucket zuweisen. Weitere Informationen zu Tagging-Strategien finden Sie unter Tagging Resources AWS.

In ist CodeCommit die primäre Ressource ein Repository. Sie können die CodeCommit Konsole AWS CLI, die CodeCommit APIs oder AWS SDKs verwenden, um Tags für ein Repository hinzuzufügen, zu verwalten und zu entfernen. Sie können Ihr Repository nicht nur anhand von Tags identifizieren, organisieren und verfolgen, sondern auch Tags in IAM-Richtlinien verwenden, um zu kontrollieren, wer Ihr Repository einsehen und mit ihm interagieren kann. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen.

Themen

- Hinzufügen eines Tags zu einem Repository
- Anzeigen von Tags für ein Repository
- Bearbeiten von Tags für ein Repository
- Entfernen Sie ein Tag aus einem Repository

Hinzufügen eines Tags zu einem Repository

Das Hinzufügen von Tags zu einem Repository kann Ihnen dabei helfen, Ihre AWS Ressourcen zu identifizieren und zu organisieren und den Zugriff darauf zu verwalten. Zuerst fügen Sie ein oder mehrere Tags (Schlüssel-Wert-Paare) zu einem Repository hinzu. Denken Sie daran, dass es hinsichtlich der Anzahl von Tags für ein Repository Grenzen gibt. Es gibt Einschränkungen im Hinblick auf die Zeichen, die Sie in die Felder für Schlüssel und Wert eingeben können. Weitere Informationen finden Sie unter Limits. Sobald Sie über Tags verfügen, können Sie IAM-Richtlinien erstellen, um den Zugriff auf das Repository auf der Grundlage dieser Tags zu verwalten. Sie können die CodeCommit Konsole oder die verwenden AWS CLI , um Tags zu einem Repository hinzuzufügen.

> **Important**
>
> Das Hinzufügen von Tags zu einem Repository kann Auswirkungen auf den Zugriff auf dieses Repository haben. Bevor Sie einem Repository ein Tag hinzufügen, sollten Sie alle IAM-Richtlinien überprüfen, die möglicherweise Tags verwenden, um den Zugriff auf Ressourcen wie Repositorys zu steuern. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen.

Weitere Informationen zum Hinzufügen von Tags zu einem Repository während der Erstellung finden Sie unter Erstellen Sie ein Repository (Konsole).

Themen

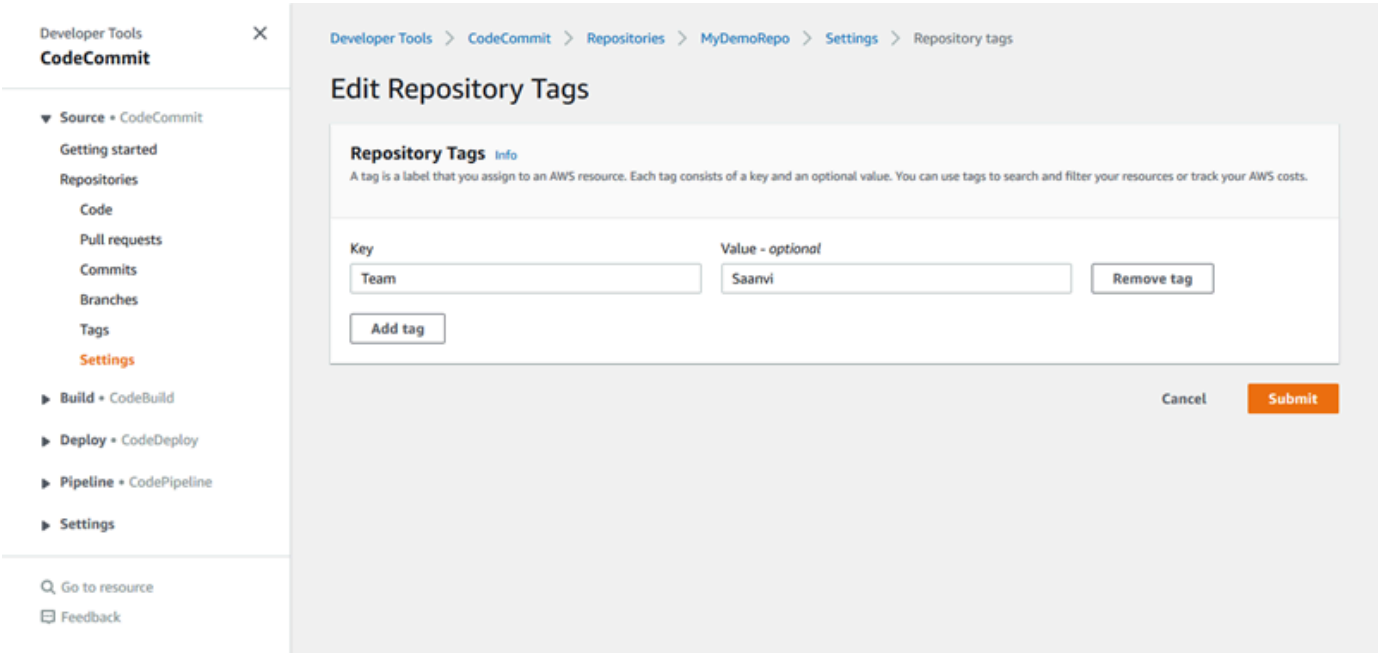
- Fügen Sie einem Repository (Konsole) ein Tag hinzu
- Fügen Sie einem Repository ein Tag hinzu (AWS CLI)

Fügen Sie einem Repository (Konsole) ein Tag hinzu

Sie können die CodeCommit Konsole verwenden, um einem CodeCommit Repository ein oder mehrere Tags hinzuzufügen.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, zu dem Sie Tags hinzufügen möchten.

3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Repository tags (Repository-Tags) aus.
4. Wenn noch keine Tags zum Repository hinzugefügt wurden, wählen Sie Add tag (Tag hinzufügen) aus. Wählen Sie andernfalls Edit (Bearbeiten) und Add tag (Tag hinzufügen) aus.
5. Geben Sie für Schlüssel einen Namen für das Tag ein. Sie können einen optionalen Wert für das Tag unter Wert hinzufügen.



6. (Optional) Zum Hinzufügen eines weiteren Tags wählen Sie Tag hinzufügen erneut aus.
7. Wenn Sie mit dem Hinzufügen von Tags fertig sind, klicken Sie auf Submit (Übermitteln).

Fügen Sie einem Repository ein Tag hinzu (AWS CLI)

Gehen Sie wie folgt vor AWS CLI , um ein Tag zu einem CodeCommit Repository hinzuzufügen. Informationen darüber, wie Sie beim Erstellen eines Repository ein Tag hinzufügen können, finden Sie unter Erstellen Sie ein Repository ()AWS CLI.

Bei diesen Schritten wird davon ausgegangen, dass Sie bereits eine aktuelle Version der AWS CLI installiert oder eine Aktualisierung auf die aktuelle Version vorgenommen haben. Weitere Informationen finden Sie unter Installieren der AWS Command Line Interface.

Führen Sie am Terminal oder über die Befehlszeile den Befehl tag-resource aus und geben Sie dabei den Amazon-Ressourcennamen (ARN) des Repositorys an, für den Sie Tags hinzufügen möchten, sowie den Schlüssel und Wert des hinzuzufügenden Tags. Sie können mehrere Tags zu einem

Repository hinzufügen. Um beispielsweise ein Repository *MyDemoRepo*mit zwei Tags zu taggen, einem Tag-Schlüssel namens *Status* mit dem Tag-Wert *Secret* und einem Tag-Schlüssel namens *Team* mit dem Tag-Wert *Saanvi*:

```
aws codecommit tag-resource --resource-arn arn:aws:codecommit:us-
west-2:111111111111:MyDemoRepo --tags Status=Secret,Team=Saanvi
```

Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.

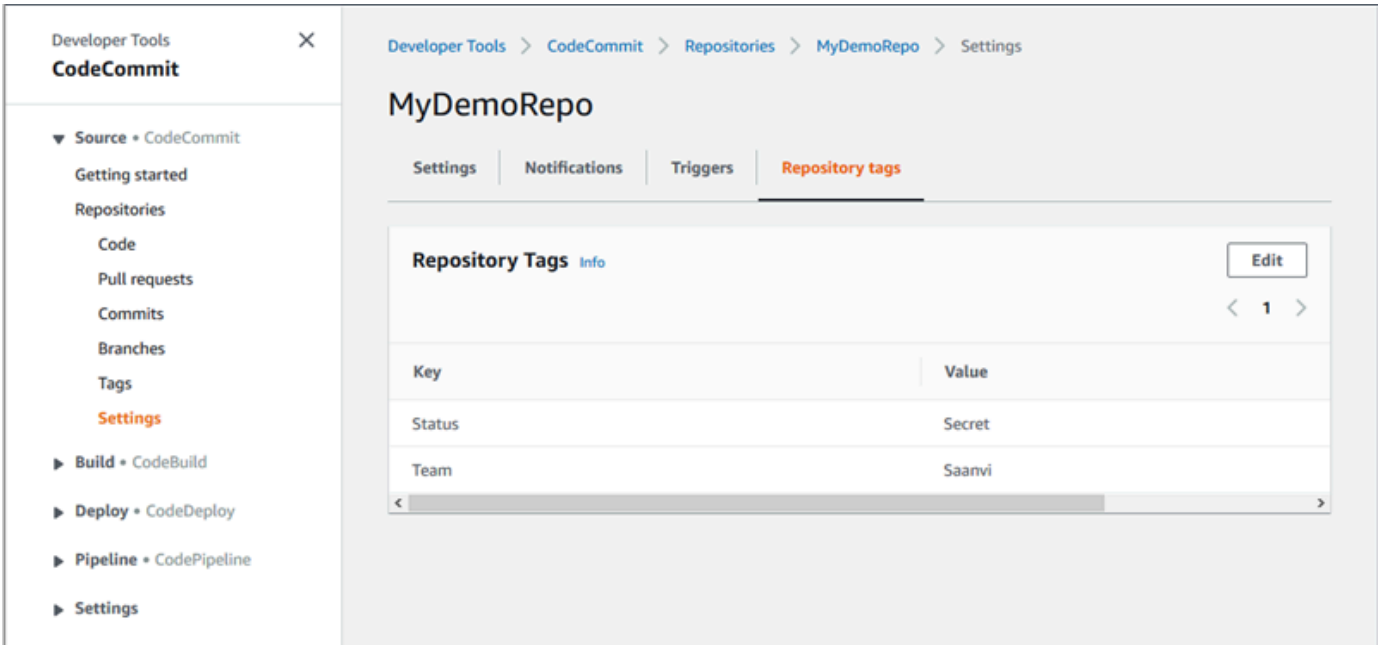
Anzeigen von Tags für ein Repository

Mithilfe von Tags können Sie Ihre AWS Ressourcen identifizieren und organisieren und den Zugriff darauf verwalten. Weitere Informationen zu Tagging-Strategien finden Sie unter Ressourcen taggen AWS. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen.

Tags für ein Repository anzeigen (Konsole)

Sie können die CodeCommit Konsole verwenden, um die mit einem CodeCommit Repository verknüpften Tags anzuzeigen.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Tags anzeigen möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Repository tags (Repository-Tags) aus.



Tags für ein Repository anzeigen (AWS CLI)

Gehen Sie wie folgt vor AWS CLI , um die AWS Tags für ein CodeCommit Repository anzuzeigen. Wenn keine Tags hinzugefügt wurden, ist die zurückgegebene Liste leer.

Führen Sie am Terminal oder über die Befehlszeile den Befehl list-tags-for-resource aus. Um beispielsweise eine Liste von Tag-Schlüsseln und Tag-Werten für ein *MyDemoRepo*mit dem ARN *arn:aws:codecommit:us-east* - 2:111111111111 benanntes Repository anzuzeigen: MyDemoRepo

```
aws codecommit list-tags-for-resource --resource-arn arn:aws:codecommit:us-
west-2:111111111111:MyDemoRepo
```

Bei erfolgreicher Ausführung gibt dieser Befehl etwa wie folgt aussehende Informationen zurück:

```
{
    "tags": {
        "Status": "Secret",
        "Team": "Saanvi"
    }
}
```

Bearbeiten von Tags für ein Repository

Sie können den Wert für ein Tag ändern, das mit dem Repository verknüpft ist. Sie können auch den Namen des Schlüssels ändern. Dies entspricht dem Entfernen des aktuellen Tags und dem Hinzufügen eines anderen Tags mit dem neuen Namen und demselben Wert wie dem des anderen. Denken Sie daran, dass es hinsichtlich der Zeichen, die Sie in die Felder für Schlüssel und Wert eingeben können, Einschränkungen gibt. Weitere Informationen finden Sie unter Limits.

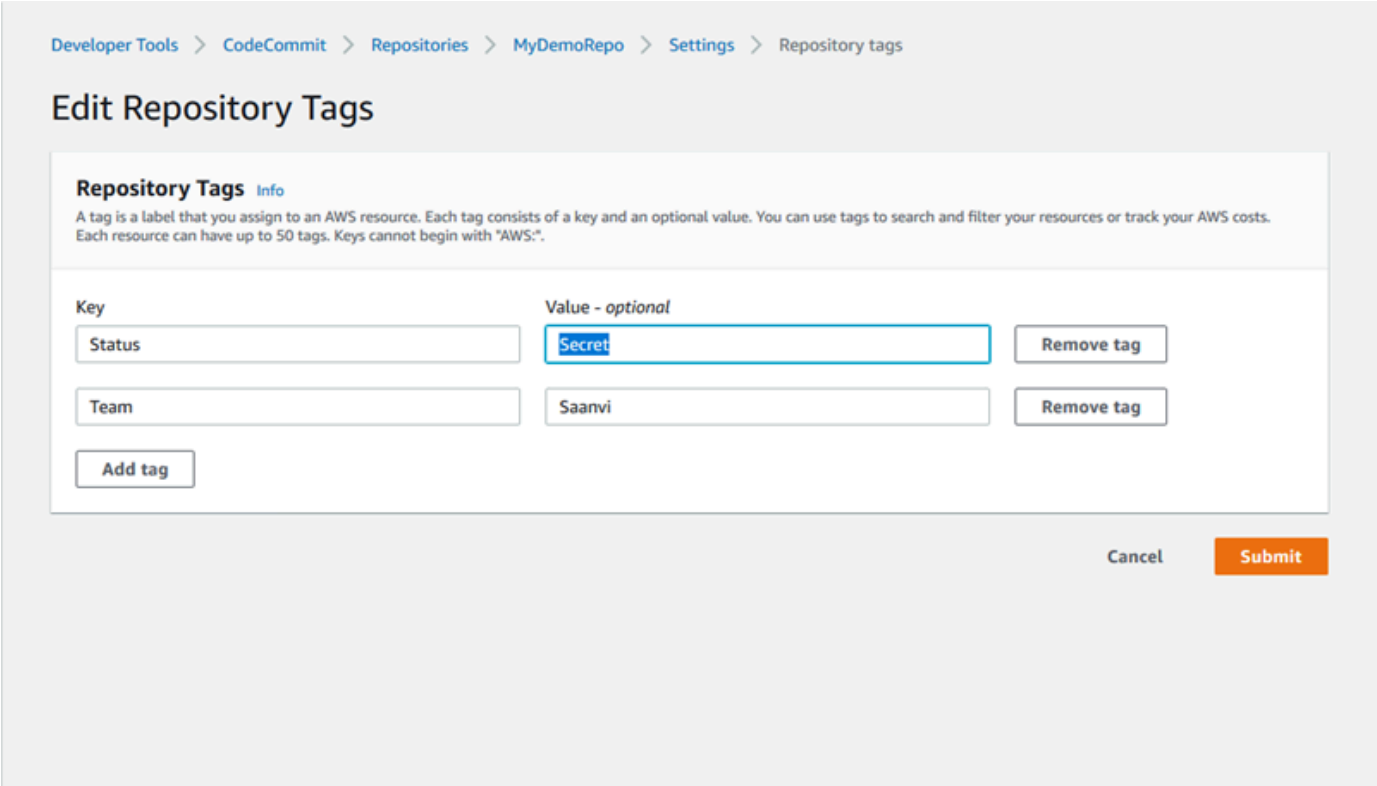
> **Important**
>
> Das Bearbeiten von Tags für ein Repository kann Auswirkungen auf den Repository-Zugriff haben. Bevor Sie den Namen (Schlüssel) oder den Wert eines Tags für ein Repository bearbeiten, sollten Sie alle IAM-Richtlinien überprüfen, die den Schlüssel oder Wert für ein Tag verwenden könnten, um den Zugriff auf Ressourcen wie Repositorys zu steuern. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen.

Bearbeiten Sie ein Tag für ein Repository (Konsole)

Sie können die CodeCommit Konsole verwenden, um die mit einem CodeCommit Repository verknüpften Tags zu bearbeiten.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, für das Sie Tags bearbeiten möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Repository tags (Repository-Tags) aus.
4. Wählen Sie Bearbeiten aus.

5.



Führen Sie eine der folgenden Aktionen aus:

- Zum Ändern des Tags geben Sie einen neuen Namen unter Key (Schlüssel) ein. Das Ändern des Namens eines Tags entspricht dem Entfernen eines Tags und Hinzufügen eines neuen Tags mit dem neuen Schlüsselnamen.
- Geben Sie zum Ändern des Werts eines Tags einen neuen Wert ein. Wenn Sie den Wert in „kein“ ändern möchten, löschen Sie den aktuellen Wert und lassen das Feld leer.

6. Wenn Sie mit dem Bearbeiten der Tags fertig sind, wählen Sie Submit (Übermitteln) aus.

Bearbeiten Sie die Tags für ein Repository (AWS CLI)

Gehen Sie wie folgt vor, AWS CLI um mit dem ein Tag für ein CodeCommit Repository zu aktualisieren. Sie können den Wert für einen vorhandenen Schlüssel ändern oder einen anderen Schlüssel hinzufügen.

Führen Sie im Terminal oder in der Befehlszeile den Befehl tag-resource aus und geben Sie dabei den Amazon-Ressourcennamen (ARN) des Repositorys, für das Sie ein Tag aktualisieren möchten, sowie den Schlüssel und den Wert des Tags an:

```
aws codecommit tag-resource --resource-arn arn:aws:codecommit:us-
west-2:111111111111:MyDemoRepo --tags Team=Li
```

Entfernen Sie ein Tag aus einem Repository

Sie können ein oder mehrere Tags entfernen, die mit einem Repository verknüpft sind. Durch das Entfernen eines Tags wird das Tag nicht aus anderen AWS Ressourcen gelöscht, die mit diesem Tag verknüpft sind.

> **Important**
>
> Das Entfernen von Tags für ein Repository kann Auswirkungen auf den Repository-Zugriff haben. Bevor Sie ein Tag aus einem Repository entfernen, sollten Sie alle IAM-Richtlinien überprüfen, die den Schlüssel oder Wert für ein Tag verwenden könnten, um den Zugriff auf Ressourcen wie Repositorys zu steuern. Beispiele für Tag-basierte Zugriffsrichtlinien finden Sie unter Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen.

Entfernen Sie ein Tag aus einem Repository (Konsole)

Sie können die CodeCommit Konsole verwenden, um die Zuordnung zwischen einem Tag und einem CodeCommit Repository zu entfernen.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, aus dem Sie Tags entfernen möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen). Wählen Sie Repository tags (Repository-Tags) aus.
4. Wählen Sie Bearbeiten aus.
5. Suchen Sie die Tags, die Sie entfernen möchten, und wählen Sie dann Remove tag (Tag entfernen) aus.
6. Wenn Sie die Tags entfernt haben, klicken Sie auf Submit (Übermitteln).

Entferne ein Tag aus einem Repository (AWS CLI)

Gehen Sie wie folgt vor, AWS CLI um mit dem ein Tag aus einem CodeCommit Repository zu entfernen. Wenn ein Tag entfernt wird, wird es nicht gelöscht, sondern lediglich die Zuordnung zwischen dem Tag und dem Repository entfernt.

> **Note**
>
> Wenn Sie ein CodeCommit Repository löschen, werden alle Tag-Verknüpfungen aus dem gelöschten Repository entfernt. Sie müssen vor dem Löschen eines Repositorys keine Tags entfernen.

Führen Sie am Terminal oder über die Befehlszeile den Befehl untag-resource aus und geben Sie dabei den Amazon-Ressourcennamen (ARN) des Repositorys an, für den Sie Tags hinzufügen möchten, sowie den Tag-Schlüssel des zu entfernenden Tags. Um beispielsweise ein Tag in einem Repository zu entfernen, das *MyDemoRepo*mit dem Tag-Schlüssel *Status* benannt ist:

```
aws codecommit untag-resource --resource-arn arn:aws:codecommit:us-
west-2:111111111111:MyDemoRepo --tag-keys Status
```

Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück. Um die dem Repository zugeordneten Tags zu überprüfen, führen Sie den Befehl list-tags-for-resource aus.

Trigger für ein AWS CodeCommit Repository verwalten

Sie können ein CodeCommit Repository so konfigurieren, dass Code-Pushs oder andere Ereignisse Aktionen auslösen, z. B. das Senden einer Benachrichtigung von Amazon Simple Notification Service (Amazon SNS) oder das Aufrufen einer Funktion in. AWS Lambda Sie können bis zu 10 Trigger für jedes Repository erstellen. CodeCommit

Auslöser werden häufig für folgende Aktionen konfiguriert:

- Versenden von E-Mails an Benutzer mit Abonnement bei jedem Push-Vorgang auf das Repository
- Benachrichtigen eines externen Build-Systems, einen Build zu starten, nachdem der Haupt-Branch des Repositorys verschoben wurde

Szenarien wie die Benachrichtigung eines externen Build-Systems erfordern das Schreiben einer Lambda-Funktion, um mit anderen Anwendungen zu interagieren. Das E-Mail-Szenario erfordert lediglich die Erstellung eines Amazon SNS SNS-Themas.

In diesem Thema erfahren Sie, wie Sie Berechtigungen festlegen, mit denen CodeCommit Aktionen in Amazon SNS und Lambda ausgelöst werden können. Außerdem enthält es Links zu Beispielen zum Erstellen, Bearbeiten, Testen und Löschen von Auslösern.

Themen

- Erstellen Sie die Ressource und fügen Sie Berechtigungen hinzu für CodeCommit
- Beispiel: Einen AWS CodeCommit Trigger für ein Amazon SNS SNS-Thema erstellen
- Beispiel: Erstellen Sie einen AWS CodeCommit Trigger für eine Funktion AWS Lambda
- Beispiel: Erstellen Sie einen Trigger AWS CodeCommit für eine bestehende AWS Lambda Funktion
- Trigger für ein Repository bearbeiten AWS CodeCommit
- Testen Sie Trigger für ein AWS CodeCommit Repository
- Trigger aus einem AWS CodeCommit Repository löschen

Erstellen Sie die Ressource und fügen Sie Berechtigungen hinzu für CodeCommit

Sie können Amazon SNS SNS-Themen und Lambda-Funktionen mit Triggern integrieren CodeCommit, aber Sie müssen zuerst Ressourcen mit einer Richtlinie erstellen und dann konfigurieren, CodeCommit die die Berechtigungen für die Interaktion mit diesen Ressourcen gewährt. Sie müssen die Ressource in derselben Umgebung AWS-Region wie das CodeCommit Repository erstellen. Wenn sich das Repository beispielsweise in USA Ost (Ohio) (us-east-2) befindet, muss sich das Amazon SNS SNS-Thema oder die Lambda-Funktion in USA Ost (Ohio) befinden.

- Für Amazon SNS SNS-Themen müssen Sie keine zusätzlichen IAM-Richtlinien oder -Berechtigungen konfigurieren, wenn das Amazon SNS SNS-Thema mit demselben Konto wie das Repository erstellt wurde. CodeCommit Sie können den CodeCommit Auslöser erstellen, sobald Sie das Amazon SNS SNS-Thema erstellt und abonniert haben.
  - Weitere Informationen zum Erstellen von Themen in Amazon SNS finden Sie in der Amazon SNS SNS-Dokumentation.

  - Informationen zur Verwendung von Amazon SNS zum Senden von Nachrichten an Amazon SQS SQS-Warteschlangen finden Sie unter Senden von Nachrichten an Amazon SQS SQS-Warteschlangen im Amazon SNS SNS-Entwicklerhandbuch.
  - Informationen zur Verwendung von Amazon SNS zum Aufrufen einer Lambda-Funktion finden Sie unter Aufrufen von Lambda-Funktionen im Amazon SNS SNS-Entwicklerhandbuch.
- Wenn Sie Ihren Auslöser so konfigurieren möchten, dass er ein Amazon SNS SNS-Thema in einem anderen AWS Konto verwendet, müssen Sie dieses Thema zunächst mit einer Richtlinie konfigurieren, die es ermöglicht, CodeCommit zu diesem Thema zu veröffentlichen. Weitere Informationen finden Sie unter Beispiel 1: Erstellen Sie eine Richtlinie, die den kontoübergreifenden Zugriff auf ein Amazon SNS SNS-Thema ermöglicht.
- Sie können Lambda-Funktionen konfigurieren, indem Sie den Trigger in der Lambda-Konsole als Teil der Funktion erstellen. Dies ist die einfachste Methode, da in der Lambda-Konsole erstellte Trigger automatisch die Berechtigungen enthalten, die für CodeCommit den Aufruf der Lambda-Funktion erforderlich sind. Wenn Sie den Trigger in erstellen CodeCommit, müssen Sie eine Richtlinie angeben, die das Aufrufen der CodeCommit Funktion ermöglicht. Weitere Informationen finden Sie unter Erstellen Sie einen Trigger für eine bestehende Lambda-Funktion und Beispiel 3: Erstellen Sie eine Richtlinie für die AWS Lambda Integration mit einem Trigger CodeCommit .

Beispiel: Einen AWS CodeCommit Trigger für ein Amazon SNS SNS-Thema erstellen

Sie können einen Trigger für ein CodeCommit Repository erstellen, sodass Ereignisse in diesem Repository Benachrichtigungen von einem Amazon Simple Notification Service (Amazon SNS) -Thema auslösen. Möglicherweise möchten Sie einen Auslöser für ein Amazon SNS SNS-Thema erstellen, damit Benutzer Benachrichtigungen über Repository-Ereignisse wie das Löschen von Branches abonnieren können. Sie können auch die Vorteile der Integration von Amazon SNS SNS-Themen mit anderen Diensten wie Amazon Simple Queue Service (Amazon SQS) und nutzen. AWS Lambda

> **Note**
>
> - Sie müssen den Auslöser auf ein vorhandenes Amazon SNS SNS-Thema verweisen, bei dem es sich um die Aktion handelt, die als Reaktion auf Repository-Ereignisse ergriffen wurde. Weitere Informationen zum Erstellen und Abonnieren von Amazon SNS SNS-Themen finden Sie unter Erste Schritte mit Amazon Simple Notification Service.

- Amazon SNS FIFO-Themen (first in, first out) werden für CodeCommit Trigger nicht unterstützt.

Themen

- Einen Auslöser für ein Amazon SNS SNS-Thema für ein CodeCommit Repository (Konsole) erstellen
- Einen Auslöser für ein Amazon SNS SNS-Thema für ein CodeCommit Repository erstellen ()AWS CLI

Einen Auslöser für ein Amazon SNS SNS-Thema für ein CodeCommit Repository (Konsole) erstellen

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie Auslöser für Repository-Ereignisse erstellen möchten.
3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Wählen Sie Create trigger (Auslöser erstellen) aus und gehen Sie folgendermaßen vor:
   - Geben Sie im Feld Triggername einen Namen für den Trigger ein (z. B. *MyFirstTrigger*).
   - Wählen Sie unter Ereignisse die Repository-Ereignisse aus, die das Amazon SNS SNS-Thema zum Senden von Benachrichtigungen veranlassen.

     Wenn Sie All repository events wählen, können Sie keine anderen Ereignisse auswählen. Wenn Sie nur einen Teil der Ereignisse auswählen möchten, löschen Sie All repository events und wählen Sie dann die gewünschten Ereignisse in der Liste aus. Wenn Sie beispielsweise möchten, dass der Trigger nur ausgeführt wird, wenn ein Benutzer einen Branch oder ein Tag im CodeCommit Repository erstellt, entfernen Sie Alle Repository-Ereignisse und wählen Sie dann Branch oder Tag erstellen.
   - Wenn Sie möchten, dass der Auslöser für alle Branches des Repositorys unter Branches gilt, lassen Sie die Auswahl leer, da diese Standardoption den Auslöser automatisch auf alle Branches anwendet. Wenn Sie möchten, dass dieser Auslöser nur für bestimmte Branches gilt, wählen Sie bis zu 10 Branch-Namen aus der Liste der Repository-Branches aus.

- Wählen Sie unter Wählen Sie den zu verwendenden Service die Option Amazon SNS aus.
- Wählen Sie in Amazon SNS einen Themennamen aus der Liste aus oder geben Sie den ARN für das Thema ein.

> **Note**
>
> Amazon SNS FIFO-Themen (first in, first out) werden für CodeCommit Trigger nicht unterstützt. Sie müssen ein Amazon SNS SNS-Thema auswählen, dessen Typ auf Standard gesetzt ist. Wenn Sie ein Amazon SNS FIFO-Thema verwenden möchten, müssen Sie eine Amazon Eventbridge-Regel für CodeCommit Ereignisse konfigurieren, für die das SNS-FIFO-Thema als Ziel konfiguriert ist.

- Geben Sie unter Benutzerdefinierte Daten alle optionalen Informationen ein, die Sie in die vom Amazon SNS SNS-Thema gesendete Benachrichtigung aufnehmen möchten (z. B. einen IRC-Kanalnamen, den Entwickler verwenden, wenn sie über Entwicklungen in diesem Repository sprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden.

5. (Optional) Wählen Sie Test trigger (Auslöser testen). Mit diesem Schritt können Sie überprüfen, ob der Zugriff zwischen CodeCommit und dem Amazon SNS SNS-Thema korrekt konfiguriert wurde. Es verwendet das Amazon SNS SNS-Thema, um eine Testbenachrichtigung mit Daten aus Ihrem Repository zu senden, sofern verfügbar. Falls keine echten Daten verfügbar sind, enthält die Testbenachrichtigung Beispieldaten.
6. Wählen Sie Create trigger (Auslöser erstellen), um die Erstellung des Auslösers abzuschließen.

Einen Auslöser für ein Amazon SNS SNS-Thema für ein CodeCommit Repository erstellen ()AWS CLI

Sie können auch die Befehlszeile verwenden, um einen Trigger für ein Amazon SNS SNS-Thema als Reaktion auf CodeCommit Repository-Ereignisse zu erstellen, z. B. wenn jemand einen Commit in Ihr Repository überträgt.

Um einen Auslöser für ein Amazon SNS SNS-Thema zu erstellen

1. Öffnen Sie einen Texteditor und erstellen Sie eine JSON-Datei mit folgenden Angaben:

   - Der Name des Amazon SNS SNS-Themas.

> **Note**
>
> Amazon SNS FIFO-Themen (first in, first out) werden für CodeCommit Trigger nicht unterstützt. Sie müssen ein Amazon SNS SNS-Thema auswählen, dessen Typ auf Standard gesetzt ist. Wenn Sie ein Amazon SNS FIFO-Thema verwenden möchten, müssen Sie eine Amazon Eventbridge-Regel für CodeCommit Ereignisse konfigurieren, für die das SNS-FIFO-Thema als Ziel konfiguriert ist.

- Repository und Branches, die mit diesem Auslöser überwacht werden sollen. (Wenn keine Branches angegeben werden, gilt der Auslöser für alle Branches im Repository.)
- Ereignisse, die diesen Auslöser aktivieren.

Speichern Sie die Datei.

Um beispielsweise einen Trigger für ein Repository mit dem MyDemoRepoNamen zu erstellen, der alle Repository-Ereignisse in einem Amazon SNS SNS-Thema namens mySNSTopic für zwei Zweige veröffentlicht, main und preprod:

```
{
    "repositoryName": "MyDemoRepo",
    "triggers": [
        {
            "name": "MyFirstTrigger",
            "destinationArn": "arn:aws:sns:us-east-2:111122223333:MySNSTopic",
            "customData": "",
            "branches": [
                "main", "preprod"
            ],
            "events": [
                "all"
            ]
        }
    ]
}
```

In der JSON-Datei muss für jeden Auslöser eines Repositorys ein Auslöserblock vorhanden sein. Wenn Sie für ein Repository mehr als einen Auslöser erstellen möchten, fügen Sie

mehrere Auslöserblöcke zur JSON-Datei hinzu. Beachten Sie, dass alle in dieser Datei erstellten Auslöser für das angegebene Repository gelten. Es ist nicht möglich, Auslöser für verschiedene Repositorys in einer JSON-Datei zu erstellen. Wenn Sie beispielsweise zwei Auslöser für ein Repository erstellen möchten, können Sie eine JSON-Datei mit zwei Auslöserblöcken erstellen. Im folgenden Beispiel werden für den zweiten Auslöser keine Branches angegeben, daher gilt dieser Auslöser für alle Branches:

```
{
    "repositoryName": "MyDemoRepo",
    "triggers": [
        {
            "name": "MyFirstTrigger",
            "destinationArn": "arn:aws:sns:us-east-2:111122223333:MySNSTopic",
            "customData": "",
            "branches": [
                "main", "preprod"
            ],
            "events": [
                "all"
            ]
        },
        {
            "name": "MySecondTrigger",
            "destinationArn": "arn:aws:sns:us-east-2:111122223333:MySNSTopic2",
            "customData": "",
            "branches": [],
            "events": [
                "updateReference", "deleteReference"
            ]
        }
    ]
}
```

Sie können Auslöser für bestimmte Ereignisse erstellen, z. B. wenn ein Commit per Push an ein Repository übertragen wird. Zu den Ereignistypen zählen:

- `all` für alle Ereignisse im angegebenen Repository und den angegebenen Branches
- `updateReference` für Fälle, in denen Commits per Push an das angegebene Repository und die angegebenen Branches übertragen werden

- `createReference` für Fälle, in denen in dem angegebenen Repository ein neuer Branch oder ein neues Tag erstellt wird
- `deleteReference` für Fälle, in denen in dem angegebenen Repository ein Branch oder ein Tag gelöscht wird

> **Note**
>
> Sie können in einem Auslöser mehrere Ereignistypen verwenden. Wenn Sie `all` angeben, können Sie allerdings keine anderen Ereignisse angeben.

Geben Sie im Terminal oder in der Eingabeaufforderung aws codecommit put-repository-triggers help ein, um eine vollständige Liste der gültigen Ereignistypen anzuzeigen.

Darüber hinaus können Sie in `customData` eine Zeichenfolge einschließen (z. B. den Namen eines IRC-Kanal, den Entwickler verwenden, wenn Sie die Entwicklung in diesem Repository besprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden. Diese Zeichenfolge wird als Attribut an die CodeCommit JSON-Datei angehängt, die als Antwort auf den Trigger zurückgegeben wird.

2. (Optional) Führen Sie an einem Terminal oder an einer Eingabeaufforderung den Befehl test-repository-triggers aus. Dieser Test verwendet Beispieldaten aus dem Repository (oder generiert Beispieldaten, falls keine Daten verfügbar sind), um eine Benachrichtigung an die Abonnenten des Amazon SNS SNS-Themas zu senden. Mit dem Folgenden wird beispielsweise getestet, ob der JSON-Code in der Trigger-Datei mit dem Namen *`trigger.json`* gültig ist und im Amazon SNS SNS-Thema veröffentlicht CodeCommit werden kann:

```
aws codecommit test-repository-triggers --cli-input-json file://trigger.json
```

Bei erfolgreicher Ausführung gibt dieser Befehl etwa wie folgt aussehende Informationen zurück:

```
{
    "successfulExecutions": [
        "MyFirstTrigger"
    ],
    "failedExecutions": []
}
```

3. Führen Sie in einem Terminal oder einer Befehlszeile den Befehl aus, in dem der put-repository-triggers Trigger erstellt werden soll. CodeCommit Sie können beispielsweise eine JSON-Datei mit dem Namen *trigger.json* verwenden, um den Auslöser zu erstellen:

   ```
   aws codecommit put-repository-triggers --cli-input-json
   file://trigger.json
   ```

   Die vom Befehl zurückgegebene Konfigurations-ID sollte wie folgt aussehen:

   ```
   {
       "configurationId": "0123456-I-AM-AN-EXAMPLE"
   }
   ```

4. Führen Sie zum Anzeigen der Konfiguration des Auslösers den Befehl get-repository-triggers aus und geben Sie dabei den Namen des Repositorys an:

   ```
   aws codecommit get-repository-triggers --repository-name MyDemoRepo
   ```

   Dieser Befehl gibt die Struktur aller Auslöser zurück, die für das Repository konfiguriert sind, ähnlich folgendem Beispiel:

   ```
   {
       "configurationId": "0123456-I-AM-AN-EXAMPLE",
       "triggers": [
           {
               "events": [
                   "all"
               ],
               "destinationArn": "arn:aws:sns:us-east-2:111122223333:MySNSTopic",
               "branches": [
                   "main",
                   "preprod"
               ],
               "name": "MyFirstTrigger",
               "customData": "Project ID 12345"
           }
       ]
   }
   ```

5. Erstellen Sie einen Commit und übertragen Sie ihn per Push an das Repository, für das Sie den Auslöser konfiguriert haben, um die Funktion des Auslösers zu prüfen. Sie sollten eine Antwort vom Amazon SNS SNS-Thema sehen. Wenn Sie beispielsweise das Amazon SNS-Thema so

konfiguriert haben, dass eine E-Mail gesendet wird, sollte in dem E-Mail-Konto, das das Thema abonniert hat, eine E-Mail von Amazon SNS angezeigt werden.

Im Folgenden finden Sie ein Beispiel für die Ausgabe einer E-Mail, die von Amazon SNS als Antwort auf einen Push an ein CodeCommit Repository gesendet wurde:

```
{
  "Records":[
     {
        "awsRegion":"us-east-2",
        "codecommit":{
           "references" : [
              {
                    "commit":"317f8570EXAMPLE",
                    "created":true,
                    "ref":"refs/heads/NewBranch"
              },
              {
                    "commit":"4c925148EXAMPLE",
                    "ref":"refs/heads/preprod",
              }
            ]
          },
        "eventId":"11111-EXAMPLE-ID",
        "eventName":"ReferenceChange",
        "eventPartNumber":1,
        "eventSource":"aws:codecommit",
        "eventSourceARN":"arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
        "eventTime":"2016-02-09T00:08:11.743+0000",
        "eventTotalParts":1,
        "eventTriggerConfigId":"0123456-I-AM-AN-EXAMPLE",
        "eventTriggerName":"MyFirstTrigger",
        "eventVersion":"1.0",
        "customData":"Project ID 12345",
        "userIdentityARN":"arn:aws:iam::111122223333:user/JaneDoe-CodeCommit",
     }
  ]
}
```

Beispiel: Erstellen Sie einen AWS CodeCommit Trigger für eine Funktion AWS Lambda

Sie können einen Trigger für ein CodeCommit Repository erstellen, sodass Ereignisse im Repository eine Lambda-Funktion aufrufen. In diesem Beispiel erstellen Sie eine Lambda-Funktion, die die URL zurückgibt, die zum Klonen des Repositorys in ein CloudWatch Amazon-Protokoll verwendet wurde.

Themen

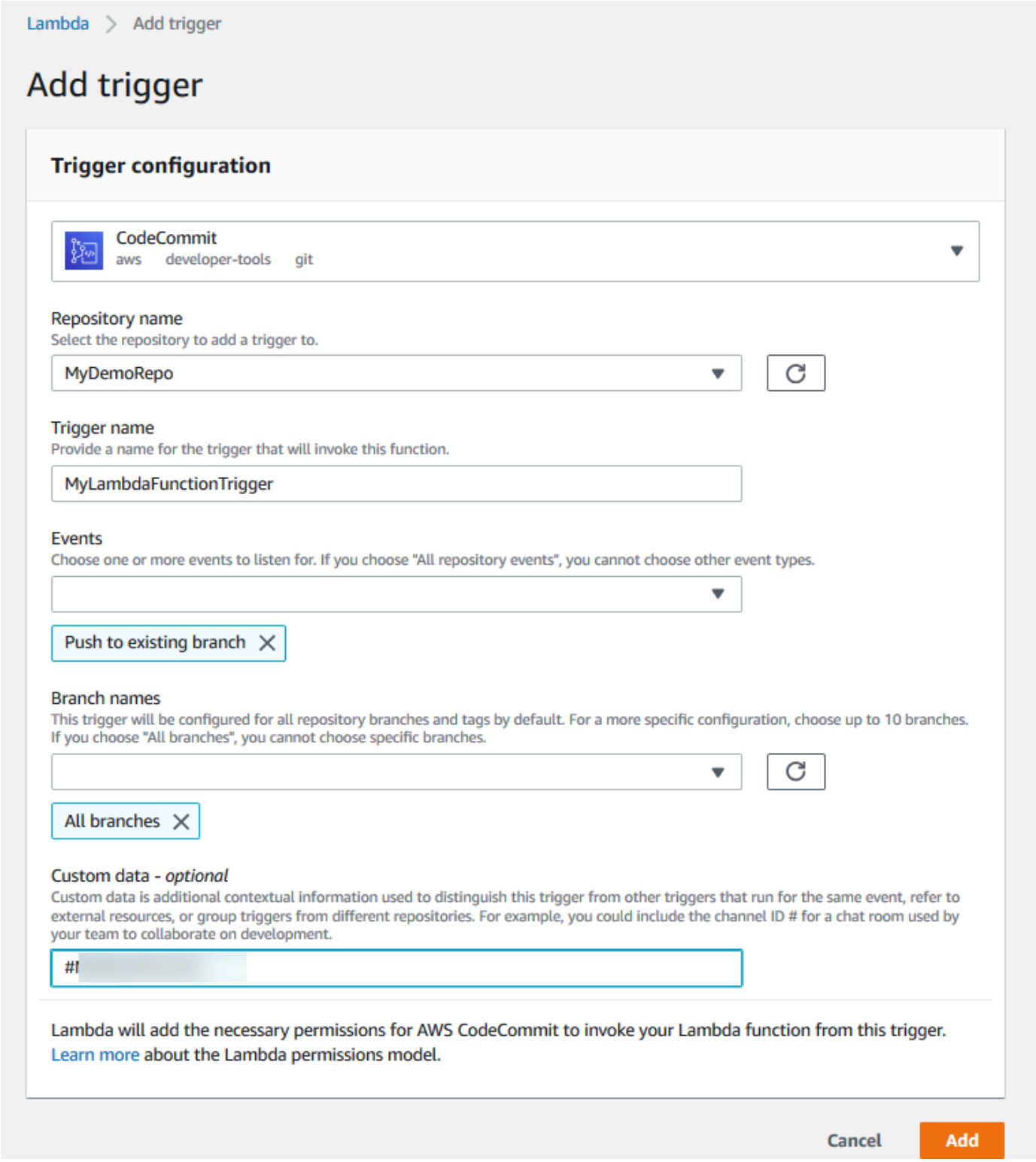
- So erstellen Sie die Lambda-Funktion:
- Den Trigger für die Lambda-Funktion im AWS CodeCommit Repository anzeigen

So erstellen Sie die Lambda-Funktion:

Wenn Sie die Lambda-Konsole verwenden, um die Funktion zu erstellen, können Sie auch einen CodeCommit Trigger für die Lambda-Funktion erstellen. Die folgenden Schritte beinhalten ein Beispiel für eine Lambda-Funktion. Das Beispiel ist in zwei Sprachen verfügbar: JavaScript und Python. Die Funktion gibt die URLs, die zum Klonen eines Repositorys verwendet wurden, in ein CloudWatch Protokoll zurück.

So erstellen Sie eine Lambda-Funktion mithilfe eines Lambda-Blueprints

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die AWS Lambda Konsole unter https://console.aws.amazon.com/lambda/.
2. Wählen Sie auf der Seite Lambda-Funktionen die Option Create function aus. (Wenn Sie Lambda noch nicht verwendet haben, wählen Sie Jetzt starten.)
3. Wählen Sie auf der Seite Create function die Option Author from scratch. Geben Sie im Feld Funktionsname beispielsweise *MyLambdaFunctionforCodeCommit*einen Namen für die Funktion ein. Wählen Sie unter Runtime (Laufzeit) die Sprache aus, die Sie verwenden möchten, um Ihre Funktion zu schreiben, und wählen Sie dann Create function (Funktion erstellen).
4. Wählen Sie auf der Registerkarte Configuration (Konfiguration) die Option Add trigger (Auslöser hinzufügen) aus.
5. Wählen Sie in der Trigger-Konfiguration eine Option CodeCommitaus der Dropdownliste Dienste aus.



- Wählen Sie unter Repository-Name den Namen des Repositorys aus, für das Sie einen Trigger konfigurieren möchten, der die Lambda-Funktion als Reaktion auf Repository-Ereignisse verwendet.

- Geben Sie im Feld Triggername einen Namen für den Trigger ein (z. B. *MyLambdaFunctionTrigger*).
- Wählen Sie unter Ereignisse die Repository-Ereignisse aus, die die Lambda-Funktion auslösen. Wenn Sie All repository events wählen, können Sie keine anderen Ereignisse auswählen. Wenn Sie nur einen Teil der Ereignisse auswählen möchten, löschen Sie All repository events und wählen Sie dann die gewünschten Ereignisse in der Liste aus. Wenn Sie beispielsweise möchten, dass der Trigger nur ausgeführt wird, wenn ein Benutzer ein Tag oder einen Branch im AWS CodeCommit Repository erstellt, entfernen Sie Alle Repository-Ereignisse und wählen Sie dann Branch oder Tag erstellen aus.
- Wenn der Auslöser für alle Branches des Repositorys gelten soll, wählen Sie unter Branches die Option All branches aus. Wählen Sie andernfalls Specific branches aus. Der Standard-Branch für das Repository wird standardmäßig hinzugefügt. Sie können diesen Branch behalten oder aus der Liste entfernen. Aus der Liste der Repository-Branches können Sie bis zu 10 Branch-Namen auswählen.
- (Optional) Geben Sie unter Benutzerdefinierte Daten Informationen ein, die Sie in die Lambda-Funktion aufnehmen möchten (z. B. den Namen des IRC-Kanals, der von Entwicklern verwendet wird, um die Entwicklung im Repository zu besprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden.

Wählen Sie Hinzufügen aus.

6. Wählen Sie auf der Seite Configuration (Konfiguration) unter Function Code (Funktionscode) und Code entry type (Codeeingabetyp) die Option Edit Code Inline (Code inline bearbeiten) aus. Wählen Sie unter Runtime (Laufzeit)die Option Node.js aus. Wenn Sie ein Python-Funktionsbeispiel erstellen möchten, wählen Sie Python aus.
7. Wählen Sie unter Code entry type die Option Edit code inline aus und ersetzen Sie den hello-world-Code durch eines der folgenden Beispiele.

   Für Node.js:

```
import {
  CodeCommitClient,
  GetRepositoryCommand,
} from "@aws-sdk/client-codecommit";

const codecommit = new CodeCommitClient({ region: "your-region" });

/**
```

```
 * @param {{ Records: { codecommit: { references: { ref: string }[] },
 eventSourceARN: string  }[]} event
 */
export const handler = async (event) => {
  // Log the updated references from the event
  const references = event.Records[0].codecommit.references.map(
    (reference) => reference.ref,
  );
  console.log("References:", references);

  // Get the repository from the event and show its git clone URL
  const repository = event.Records[0].eventSourceARN.split(":")[5];
  const params = {
    repositoryName: repository,
  };

  try {
    const data = await codecommit.send(new GetRepositoryCommand(params));
    console.log("Clone URL:", data.repositoryMetadata.cloneUrlHttp);
    return data.repositoryMetadata.cloneUrlHttp;
  } catch (error) {
    console.error("Error:", error);
    throw new Error(
      `Error getting repository metadata for repository ${repository}`,
    );
  }
};
```

Für Python:

```
import json
import boto3

codecommit = boto3.client("codecommit")


def lambda_handler(event, context):
    # Log the updated references from the event
    references = {
        reference["ref"]
        for reference in event["Records"][0]["codecommit"]["references"]
    }
```

```
    print("References: " + str(references))

    # Get the repository from the event and show its git clone URL
    repository = event["Records"][0]["eventSourceARN"].split(":")[5]
    try:
        response = codecommit.get_repository(repositoryName=repository)
        print("Clone URL: " + response["repositoryMetadata"]["cloneUrlHttp"])
        return response["repositoryMetadata"]["cloneUrlHttp"]
    except Exception as e:
        print(e)
        print(
            "Error getting repository {}. Make sure it exists and that your
repository is in the same region as this function.".format(
                repository
            )
        )
        raise e
```

8. Wählen Sie auf der Registerkarte Berechtigungen unter Ausführungsrolle die Rolle aus, um sie in der IAM-Konsole zu öffnen. Bearbeiten Sie die angefügte Richtlinie, um dem Repository, das für den Auslöser verwendet werden soll, eine `GetRepository`-Berechtigung zu erteilen.

Den Trigger für die Lambda-Funktion im AWS CodeCommit Repository anzeigen

Nachdem Sie die Lambda-Funktion erstellt haben, können Sie den Trigger in AWS CodeCommit anzeigen und testen. Beim Testen des Auslösers wird die Funktion als Reaktion auf die von Ihnen angegebenen Repository-Ereignisse ausgeführt.

So zeigen Sie den Trigger für die Lambda-Funktion an und testen ihn

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie die Auslöser anzeigen möchten.
3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Überprüfen Sie die Liste der Auslöser für das Repository. Sie sollten den Trigger sehen, den Sie in der Lambda-Konsole erstellt haben. Wählen Sie ihn aus der Liste aus und wählen Sie

Test trigger (Trigger testen). Damit wird die Funktion mit Beispieldaten zu Ihrem Repository, einschließlich der neuesten Commit-ID für das Repository, aufgerufen. (Falls kein Commit-Verlauf vorhanden ist, werden stattdessen aus Nullen bestehende Beispielwerte generiert.) Auf diese Weise können Sie überprüfen, ob Sie den Zugriff zwischen AWS CodeCommit und der Lambda-Funktion korrekt konfiguriert haben.

5. Erstellen Sie einen Commit und übertragen Sie ihn per Push an das Repository, für das Sie den Auslöser konfiguriert haben, um die Funktion des Auslösers weiter zu überprüfen. Sie sollten eine Antwort der Lambda-Funktion auf der Registerkarte Überwachung für diese Funktion in der Lambda-Konsole sehen. Wählen Sie auf der Registerkarte Überwachung die Option Anmeldungen anzeigen aus. CloudWatch Die CloudWatch Konsole wird in einer neuen Registerkarte geöffnet und zeigt Ereignisse für Ihre Funktion an. Wählen Sie aus der Liste den Protokollstream mit dem Zeitpunkt aus, zu dem Sie den Commit per Push übertragen haben. Die angezeigten Ereignisdaten sollten wie folgt aussehen:

```
START RequestId: 70afdc9a-EXAMPLE Version: $LATEST
2015-11-10T18:18:28.689Z 70afdc9a-EXAMPLE References: [ 'refs/heads/main' ]
2015-11-10T18:18:29.814Z 70afdc9a-EXAMPLE Clone URL: https://git-codecommit.us-
east-2.amazonaws.com/v1/repos/MyDemoRepo
END RequestId: 70afdc9a-EXAMPLE
REPORT RequestId: 70afdc9a-EXAMPLE Duration: 1126.87 ms Billed Duration: 1200 ms
 Memory Size: 128 MB Max Memory Used: 14 MB
```

Beispiel: Erstellen Sie einen Trigger AWS CodeCommit für eine bestehende AWS Lambda Funktion

Der einfachste Weg, einen Trigger zu erstellen, der eine Lambda-Funktion aufruft, besteht darin, diesen Trigger in der Lambda-Konsole zu erstellen. Diese integrierte Integration stellt sicher, dass Sie CodeCommit über die zum Ausführen der Funktion erforderlichen Berechtigungen verfügen. Um einen Trigger für eine bestehende Lambda-Funktion hinzuzufügen, rufen Sie die Lambda-Konsole auf und wählen Sie die Funktion aus. Führen Sie auf der Registerkarte Triggers (Auslöser) zu der Funktion die Schritte unter Add trigger (Auslöser hinzufügen) aus. Diese Schritte gleichen denen unter So erstellen Sie die Lambda-Funktion:.

Sie können auch einen Trigger für eine Lambda-Funktion in einem CodeCommit Repository erstellen. Dazu müssen Sie eine vorhandene Lambda-Funktion zum Aufrufen auswählen. Außerdem müssen Sie die für CodeCommit die Ausführung der Funktion erforderlichen Berechtigungen manuell konfigurieren.

Themen

- Manuelles Konfigurieren von Berechtigungen, um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen
- Erstellen Sie einen Trigger für die Lambda-Funktion in einem CodeCommit Repository (Konsole)
- Erstellen Sie einen Trigger für eine Lambda-Funktion für ein CodeCommit Repository ()AWS CLI

Manuelles Konfigurieren von Berechtigungen, um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen

Wenn Sie einen Trigger erstellen CodeCommit , der eine Lambda-Funktion aufruft, müssen Sie die Berechtigungen, mit denen CodeCommit die Lambda-Funktion ausgeführt werden kann, manuell konfigurieren. Um diese manuelle Konfiguration zu vermeiden, sollten Sie den Trigger für die Funktion stattdessen in der Lambda-Konsole erstellen.

Um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen

1. Öffnen Sie einen Klartext-Editor und erstellen Sie eine JSON-Datei, die den Namen der Lambda-Funktion, die Details des CodeCommit Repositorys und die Aktionen angibt, die Sie in Lambda zulassen möchten, ähnlich der folgenden:

   ```
   {
       "FunctionName": "MyCodeCommitFunction",
       "StatementId": "1",
       "Action": "lambda:InvokeFunction",
       "Principal": "codecommit.amazonaws.com",
       "SourceArn": "arn:aws:codecommit:us-east-1:111122223333:MyDemoRepo",
       "SourceAccount": "111122223333"
   }
   ```

2. Speichern Sie die Datei als JSON-Datei mit einem Namen, den Sie sich leicht merken können (z. B.. json). *AllowAccessfromMyDemoRepo*

3. Führen Sie mithilfe der JSON-Datei, die Sie gerade erstellt haben, im Terminal (Linux, macOS oder Unix) oder in der Befehlszeile (Windows) den aws lambda add-permissions Befehl aus, um der mit Ihrer Lambda-Funktion verknüpften Ressourcenrichtlinie eine Berechtigung hinzuzufügen:

   ```
   aws lambda add-permission --cli-input-json file://AllowAccessfromMyDemoRepo.json
   ```

Dieser Befehl gibt den JSON-Wert der soeben hinzugefügten Richtlinienanweisung zurück, ähnlich folgendem Beispiel:

```
{
    "Statement": "{\"Condition\":{\"StringEquals\":{\"AWS:SourceAccount
\":\"111122223333\"},\"ArnLike\":{\"AWS:SourceArn\":
\"arn:aws:codecommit:us-east-1:111122223333:MyDemoRepo\"}},\"Action
\":[\"lambda:InvokeFunction\"],\"Resource\":\"arn:aws:lambda:us-
east-1:111122223333:function:MyCodeCommitFunction\",\"Effect\":\"Allow\",
\"Principal\":{\"Service\":\"codecommit.amazonaws.com\"},\"Sid\":\"1\"}"
}
```

Weitere Informationen zu Ressourcenrichtlinien für Lambda-Funktionen finden Sie unter AddPermissionund The Pull/Push Event Models im AWS Lambda Benutzerhandbuch.

4. Melden Sie sich unter https://console.aws.amazon.com/iam/ bei der IAM-Konsole an AWS Management Console und öffnen Sie sie.
5. Wählen Sie im Navigationsbereich Dashboard die Option Roles und in der Liste der Rollen *lambda_basic_execution* aus.
6. Wählen Sie auf der Zusammenfassungsseite für die Rolle die Registerkarte Permissions (Berechtigungen) aus und wählen Sie unter Inline Policies (Eingebundene Richtlinien) die Option Create Role Policy (Rollenrichtlinie erstellen) aus.
7. Wählen Sie auf der Seite Set Permissions (Berechtigungen festlegen) die Option Policy Generator (Richtliniengenerator) und dann Select (Auswählen) aus.
8. Führen Sie auf der Seite Edit Permissions die folgenden Schritte aus:

   - Wählen Sie unter Effect die Option Allow aus.
   - Wählen Sie AWS unter Service die Option AWS CodeCommit.
   - Wählen Sie unter Aktionen die Option aus GetRepository.
   - Geben Sie unter Amazon Resource Name (ARN) (Amazon-Ressourcenname (ARN)) den ARN für das Repository ein (z. B. `arn:aws:codecommit:us-east-1:111122223333:MyDemoRepo`).

   Wählen Sie Add Statement und anschließend Next Step aus.
9. Wählen Sie auf der Seite Review Policy (Richtlinie überprüfen) die Option Apply Policy (Richtlinie anwenden) aus.

Die Richtlinienanweisung sollte dann dem folgenden Beispiel ähneln:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "Stmt11111111",
            "Effect": "Allow",
            "Action": [
                "codecommit:GetRepository"
            ],
            "Resource": [
                "arn:aws:codecommit:us-east-1:111122223333:MyDemoRepo"
            ]
        }
    ]
}
```

Erstellen Sie einen Trigger für die Lambda-Funktion in einem CodeCommit Repository (Konsole)

Nachdem Sie die Lambda-Funktion erstellt haben, können Sie einen Trigger erstellen, der CodeCommit die Funktion als Reaktion auf die von Ihnen angegebenen Repository-Ereignisse ausführt.

> **Note**
>
> Bevor Sie den Trigger für das Beispiel erfolgreich testen oder ausführen können, müssen Sie die Richtlinien konfigurieren, die es ermöglichen CodeCommit , die Funktion und die Lambda-Funktion aufzurufen, um Informationen über das Repository abzurufen. Weitere Informationen finden Sie unter Um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen.

Um einen Trigger für eine Lambda-Funktion zu erstellen

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.

2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie Auslöser für Repository-Ereignisse erstellen möchten.
3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Wählen Sie Create trigger.
5. Gehen Sie in Create trigger (Auslöser erstellen) wie folgt vor:

   - Geben Sie im Feld Triggername einen Namen für den Trigger ein (z. B. *MyLambdaFunctionTrigger*).
   - Wählen Sie unter Ereignisse die Repository-Ereignisse aus, die die Lambda-Funktion auslösen.

     Wenn Sie All repository events wählen, können Sie keine anderen Ereignisse auswählen. Wenn Sie nur einen Teil der Ereignisse auswählen möchten, löschen Sie All repository events und wählen Sie dann die gewünschten Ereignisse in der Liste aus. Wenn Sie beispielsweise möchten, dass der Trigger nur ausgeführt wird, wenn ein Benutzer ein Tag oder einen Branch im CodeCommit Repository erstellt, entfernen Sie Alle Repository-Ereignisse und wählen Sie dann Branch oder Tag erstellen aus.
   - Wenn Sie möchten, dass der Auslöser für alle Branches des Repositorys unter Branches gilt, lassen Sie die Auswahl leer, da diese Standardoption den Auslöser automatisch auf alle Branches anwendet. Wenn Sie möchten, dass dieser Auslöser nur für bestimmte Branches gilt, wählen Sie bis zu 10 Branch-Namen aus der Liste der Repository-Branches aus.
   - Wählen Sie unter Choose the service to use (Zu verwendende Services auswählen) die AWS Lambda aus.
   - Wählen Sie Lambda Lambda-Funktion den Funktionsnamen aus der Liste aus, oder geben Sie den ARN für die Funktion ein.
   - (Optional) Geben Sie unter Benutzerdefinierte Daten Informationen ein, die Sie in die Lambda-Funktion aufnehmen möchten (z. B. den Namen des IRC-Kanals, der von Entwicklern verwendet wird, um die Entwicklung im Repository zu besprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden.
6. (Optional) Wählen Sie Test trigger (Auslöser testen). Damit wird die Funktion mit Beispieldaten zu Ihrem Repository, einschließlich der neuesten Commit-ID für das Repository, aufgerufen. (Falls kein Commit-Verlauf vorhanden ist, werden stattdessen aus Nullen bestehende Beispielwerte generiert.) Auf diese Weise können Sie überprüfen, ob Sie den Zugriff zwischen CodeCommit und der Lambda-Funktion korrekt konfiguriert haben.

7. Wählen Sie Create trigger (Auslöser erstellen), um die Erstellung des Auslösers abzuschließen.
8. Erstellen Sie einen Commit und übertragen Sie ihn per Push an das Repository, für das Sie den Auslöser konfiguriert haben, um die Funktion des Auslösers zu überprüfen. Sie sollten eine Antwort der Lambda-Funktion auf der Registerkarte Überwachung für diese Funktion in der Lambda-Konsole sehen.

Erstellen Sie einen Trigger für eine Lambda-Funktion für ein CodeCommit Repository ()AWS CLI

Sie können auch die Befehlszeile verwenden, um einen Trigger für eine Lambda-Funktion als Reaktion auf CodeCommit Repository-Ereignisse zu erstellen, z. B. wenn jemand einen Commit in Ihr Repository überträgt.

Um einen Trigger für eine Lambda-Funktion zu erstellen

1. Öffnen Sie einen Texteditor und erstellen Sie eine JSON-Datei mit folgenden Angaben:

   - Der Name der Lambda-Funktion.
   - Repository und Branches, die mit diesem Auslöser überwacht werden sollen. (Wenn keine Branches angegeben werden, gilt der Auslöser für alle Branches im Repository.)
   - Ereignisse, die diesen Auslöser aktivieren.

   Speichern Sie die Datei.

   Wenn Sie beispielsweise einen Trigger für ein Repository mit dem Namen erstellen möchten *MyDemoRepo*, der alle Repository-Ereignisse in einer Lambda-Funktion veröffentlicht, die *MyCodeCommitFunction*nach zwei Zweigen benannt ist, *main* und *preprod*:

```
{
    "repositoryName": "MyDemoRepo",
    "triggers": [
        {
            "name": "MyLambdaFunctionTrigger",
            "destinationArn": "arn:aws:lambda:us-
east-1:111122223333:function:MyCodeCommitFunction",
            "customData": "",
            "branches": [
                "main", "preprod"
```

```
            ],
            "events": [
                "all"
            ]
        }
    ]
}
```

In der JSON-Datei muss für jeden Auslöser eines Repositorys ein Auslöserblock vorhanden sein. Wenn Sie für ein Repository mehr als einen Auslöser erstellen möchten, fügen Sie in der JSON-Datei weitere Blöcke hinzu. Beachten Sie, dass alle in dieser Datei erstellten Auslöser für das angegebene Repository gelten. Es ist nicht möglich, Auslöser für verschiedene Repositorys in einer JSON-Datei zu erstellen. Wenn Sie beispielsweise zwei Auslöser für ein Repository erstellen möchten, können Sie eine JSON-Datei mit zwei Auslöserblöcken erstellen. Im folgenden Beispiel sind im zweiten Auslöserblock keine Branches angegeben, daher gilt dieser Auslöser für alle Branches:

```
{
    "repositoryName": "MyDemoRepo",
    "triggers": [
        {
            "name": "MyLambdaFunctionTrigger",
            "destinationArn": "arn:aws:lambda:us-
east-1:111122223333:function:MyCodeCommitFunction",
            "customData": "",
            "branches": [
                "main", "preprod"
            ],
            "events": [
                "all"
            ]
        },
        {
            "name": "MyOtherLambdaFunctionTrigger",
            "destinationArn": "arn:aws:lambda:us-
east-1:111122223333:function:MyOtherCodeCommitFunction",
            "customData": "",
            "branches": [],
            "events": [
                "updateReference", "deleteReference"
            ]
```

```
        }
    ]
}
```

Sie können Auslöser für bestimmte Ereignisse erstellen, z. B. wenn ein Commit per Push an ein Repository übertragen wird. Zu den Ereignistypen zählen:

- `all` für alle Ereignisse im angegebenen Repository und den angegebenen Branches
- `updateReference` für Fälle, in denen Commits per Push an das angegebene Repository und die angegebenen Branches übertragen werden
- `createReference` für Fälle, in denen in dem angegebenen Repository ein neuer Branch oder ein neues Tag erstellt wird
- `deleteReference` für Fälle, in denen in dem angegebenen Repository ein Branch oder ein Tag gelöscht wird

> **Note**
>
> Sie können in einem Auslöser mehrere Ereignistypen verwenden. Wenn Sie `all` angeben, können Sie allerdings keine anderen Ereignisse angeben.

Geben Sie im Terminal oder in der Eingabeaufforderung aws codecommit put-repository-triggers help ein, um eine vollständige Liste der gültigen Ereignistypen anzuzeigen.

Darüber hinaus können Sie in `customData` eine Zeichenfolge einschließen (z. B. den Namen eines IRC-Kanal, den Entwickler verwenden, wenn Sie die Entwicklung in diesem Repository besprechen). Dies ist ein Zeichenfolgenfeld. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden. Diese Zeichenfolge wird als Attribut an die CodeCommit JSON-Datei angehängt, die als Antwort auf den Trigger zurückgegeben wird.

2. (Optional) Führen Sie an einem Terminal oder an einer Eingabeaufforderung den Befehl test-repository-triggers aus. Mit dem Folgenden wird beispielsweise getestet, ob die JSON-Datei mit dem Namen *`trigger.json`* gültig ist und die Lambda-Funktion auslösen CodeCommit kann. Dieser Test verwendet Beispieldaten, um die Funktion auszulösen, wenn keine echten Daten vorhanden sind.

```
aws codecommit test-repository-triggers --cli-input-json file://trigger.json
```

Bei erfolgreicher Ausführung gibt dieser Befehl etwa wie folgt aussehende Informationen zurück:

```
{
    "successfulExecutions": [
        "MyLambdaFunctionTrigger"
    ],
    "failedExecutions": []
}
```

3. Führen Sie in einem Terminal oder einer Befehlszeile den Befehl aus, in dem der put-repository-triggers Trigger erstellt werden soll. CodeCommit Sie können beispielsweise eine JSON-Datei mit dem Namen *trigger.json* verwenden, um den Auslöser zu erstellen:

   ```
   aws codecommit put-repository-triggers --cli-input-json
   file://trigger.json
   ```

   Dieser Befehl gibt eine Konfigurations-ID wie die folgende zurück:

   ```
   {
       "configurationId": "0123456-I-AM-AN-EXAMPLE"
   }
   ```

4. Führen Sie zum Anzeigen der Konfiguration des Auslösers den Befehl get-repository-triggers aus und geben Sie dabei den Namen des Repositorys an:

   ```
   aws codecommit get-repository-triggers --repository-name MyDemoRepo
   ```

   Dieser Befehl gibt die Struktur aller Auslöser zurück, die für das Repository konfiguriert sind, ähnlich folgendem Beispiel:

   ```
   {
       "configurationId": "0123456-I-AM-AN-EXAMPLE",
       "triggers": [
           {
               "events": [
                   "all"
               ],
               "destinationArn": "arn:aws:lambda:us-
   east-1:111122223333:MyCodeCommitFunction",
               "branches": [
                   "main",
   ```

```
                "preprod"
            ],
            "name": "MyLambdaFunctionTrigger",
            "customData": "Project ID 12345"
        }
    ]
}
```

5. Erstellen Sie einen Commit und übertragen Sie ihn per Push an das Repository, für das Sie den Auslöser konfiguriert haben, um die Funktion des Auslösers zu prüfen. Sie sollten eine Antwort der Lambda-Funktion auf der Registerkarte Überwachung für diese Funktion in der Lambda-Konsole sehen.

Trigger für ein Repository bearbeiten AWS CodeCommit

Sie können die Trigger bearbeiten, die für ein CodeCommit Repository erstellt wurden. Sie können die Ereignisse und Branches für den Auslöser, die als Reaktion auf das Ereignis auszuführende Aktion und andere Einstellungen bearbeiten.

Themen

- Bearbeiten Sie einen Trigger für ein Repository (Konsole)
- Bearbeiten Sie einen Trigger für ein Repository (AWS CLI)

Bearbeiten Sie einen Trigger für ein Repository (Konsole)

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, bei dem Sie einen Auslöser für Repository-Ereignisse bearbeiten möchten.
3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Wählen Sie aus der Liste der Auslöser für das Repository den Auslöser aus, den Sie bearbeiten möchten, und klicken Sie auf Edit (Bearbeiten).
5. Nehmen Sie die gewünschten Änderungen am Auslöser vor und wählen Sie anschließend Save (Speichern) aus.

Bearbeiten Sie einen Trigger für ein Repository (AWS CLI)

1. Führen Sie an einem Terminal (Linux, macOS oder Unix) oder einer Befehlszeile (Windows) den get-repository-triggers Befehl aus, um eine JSON-Datei mit der Struktur aller Trigger zu erstellen, die für Ihr Repository konfiguriert sind. Um beispielsweise eine JSON-Datei namens *MyTriggers.json* mit der Struktur aller Trigger zu erstellen, die für ein Repository mit dem Namen *MyDemoRepo*konfiguriert sind:

   ```
   aws codecommit get-repository-triggers --repository-name MyDemoRepo
    >MyTriggers.json
   ```

   Dieser Befehl gibt nichts zurück, aber eine Datei namens *MyTriggers.json* wird in dem Verzeichnis erstellt, in dem Sie den Befehl ausgeführt haben.

2. Bearbeiten Sie die JSON-Datei in einem Texteditor und ändern Sie den Auslöserblock für den zu bearbeitenden Auslöser. Ersetzen Sie das `configurationId`-Paar durch ein `repositoryName`-Paar. Speichern Sie die Datei.

   Wenn Sie beispielsweise einen Trigger bearbeiten möchten, der *MyFirstTrigger*im Repository `configurationId` mit `repositoryName` dem Namen *MyDemoRepo*so benannt ist, dass er für alle Zweige gilt, ersetzen Sie ihn durch und entfernen Sie die angegebenen `preprod` Verzweigungen in *rotem kursivem Text main* und entfernen Sie sie. Wenn keine Branches angegeben werden, gilt der Auslöser standardmäßig für alle Branches im Repository:

   ```
   {
       "repositoryName": "MyDemoRepo",
       "triggers": [
           {
               "destinationArn": "arn:aws:sns:us-
   east-2:111122223333:MyCodeCommitTopic",
               "branches": [
                   "main",
                   "preprod"
               ],
               "name": "MyFirstTrigger",
               "customData": "",
               "events": [
                   "all"
               ]
           }
       ]
   ```

```
}
```

3. Führen Sie am Terminal oder über die Befehlszeile den Befehl put-repository-triggers aus. Dadurch werden alle Trigger für das Repository aktualisiert, einschließlich der Änderungen, die Sie am *MyFirstTrigger*Trigger vorgenommen haben:

```
aws codecommit put-repository-triggers --repository-name MyDemoRepo
 file://MyTriggers.json
```

Dieser Befehl gibt eine Konfigurations-ID wie die folgende zurück:

```
{
    "configurationId": "0123456-I-AM-AN-EXAMPLE"
}
```

Testen Sie Trigger für ein AWS CodeCommit Repository

Sie können die Trigger testen, die für ein CodeCommit Repository erstellt wurden. Beim Testen wird der Auslöser mit Beispieldaten aus Ihrem Repository ausgeführt, einschließlich der aktuellen Commit-ID. Wenn für das Repository kein Commit-Verlauf existiert, werden stattdessen aus Nullen bestehende Beispielwerte generiert. Durch das Testen von Triggern können Sie überprüfen, ob Sie den Zugriff zwischen CodeCommit und dem Ziel des Triggers korrekt konfiguriert haben, unabhängig davon, ob es sich um eine AWS Lambda Funktion oder eine Amazon Simple Notification Service-Benachrichtigung handelt.

Themen

- Testen Sie einen Trigger für ein Repository (Konsole)
- Testen Sie einen Trigger für ein Repository ()AWS CLI

Testen Sie einen Trigger für ein Repository (Konsole)

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, bei dem Sie einen Auslöser für Repository-Ereignisse testen möchten.

3. Wählen Sie im Navigationsbereich für das Repository Settings (Einstellungen) und anschließend Triggers (Auslöser).
4. Wählen Sie den Auslöser aus, den Sie testen möchten, und klicken Sie auf Test trigger (Auslöser testen). Es sollte eine Erfolgs- oder Fehlermeldung angezeigt werden. Bei Erfolg sollte Ihnen auch eine entsprechende Aktionsantwort von der Lambda-Funktion oder dem Amazon SNS SNS-Thema angezeigt werden.

Testen Sie einen Trigger für ein Repository ()AWS CLI

1. Führen Sie an einem Terminal (Linux, macOS oder Unix) oder einer Befehlszeile (Windows) den get-repository-triggers Befehl aus, um eine JSON-Datei mit der Struktur aller Trigger zu erstellen, die für Ihr Repository konfiguriert sind. Um beispielsweise eine JSON-Datei namens *TestTrigger.json* mit der Struktur aller Trigger zu erstellen, die für ein Repository mit dem Namen MyDemoRepo konfiguriert sind:

```
aws codecommit get-repository-triggers --repository-name MyDemoRepo
 >TestTrigger.json
```

   Dieser Befehl erstellt eine Datei mit dem Namen *TestTriggers.json* in dem Verzeichnis, in dem Sie den Befehl ausgeführt haben.
2. Bearbeiten Sie die JSON-Datei in einem Texteditor und nehmen Sie die gewünschten Änderungen an der Auslöseranweisung vor. Ersetzen Sie das `configurationId`-Paar durch ein `repositoryName`-Paar. Speichern Sie die Datei.

   Wenn Sie beispielsweise einen Trigger testen möchten, der *MyFirstTrigger*in dem *MyDemoRepo*so benannten Repository benannt ist, dass er für alle Branches gilt, ersetzen Sie den `configurationId` durch `repositoryName` und speichern Sie dann eine Datei, die der folgenden ähnelt, *TestTriggerals.json*:

```
{
    "repositoryName": "MyDemoRepo",
    "triggers": [
        {
            "destinationArn": "arn:aws:sns:us-
east-2:111122223333:MyCodeCommitTopic",
            "branches": [
                "main",
                "preprod"
```

```
            ],
            "name": "MyFirstTrigger",
            "customData": "",
            "events": [
                "all"
            ]
        }
    ]
}
```

3. Führen Sie am Terminal oder über die Befehlszeile den Befehl test-repository-triggers aus. Dadurch werden alle Trigger für das Repository aktualisiert, einschließlich der Änderungen, die Sie am *MyFirstTrigger*Trigger vorgenommen haben:

```
aws codecommit test-repository-triggers --cli-input-json file://TestTrigger.json
```

Dieser Befehl gibt eine Reaktion wie die folgende zurück:

```
{
    "successfulExecutions": [
        "MyFirstTrigger"
    ],
    "failedExecutions": []
}
```

Trigger aus einem AWS CodeCommit Repository löschen

Sie können Auslöser, die nicht mehr benötigt werden, löschen. Dieser Löschvorgang kann nicht rückgängig gemacht werden, aber Sie können die Auslöser erneut erstellen.

> **Note**
>
> Wenn Sie einen oder mehrere Trigger für Ihr Repository konfiguriert haben, werden beim Löschen des Repositorys nicht die Amazon SNS SNS-Themen oder Lambda-Funktionen gelöscht, die Sie als Ziele dieser Trigger konfiguriert haben. Achten Sie darauf, diese Ressourcen ebenfalls zu löschen, sofern sie nicht mehr benötigt werden.

Themen

- Löscht einen Trigger aus einem Repository (Konsole)
- Löscht einen Trigger aus einem Repository (AWS CLI)

Löscht einen Trigger aus einem Repository (Konsole)

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys)das Repository aus, in dem Sie Auslöser für Repository-Ereignisse löschen möchten.
3. Wählen Sie im Navigationsbereich des Repositorys die Option Settings aus. Klicken Sie unter Settings auf die Option Triggers.
4. Wählen Sie den zu löschenden Auslöser in der Liste aus und wählen Sie dann Delete (Löschen).
5. Geben Sie zur Bestätigung im Dialogfeld delete ein.

Löscht einen Trigger aus einem Repository (AWS CLI)

1. Führen Sie an einem Terminal (Linux, macOS oder Unix) oder einer Befehlszeile (Windows) den get-repository-triggers Befehl aus, um eine JSON-Datei mit der Struktur aller Trigger zu erstellen, die für Ihr Repository konfiguriert sind. Um beispielsweise eine JSON-Datei namens *MyTriggers.json* mit der Struktur aller Trigger zu erstellen, die für ein Repository mit dem Namen MyDemoRepo konfiguriert sind:

   ```
   aws codecommit get-repository-triggers --repository-name MyDemoRepo
    >MyTriggers.json
   ```

   Dieser Befehl erstellt eine Datei mit dem Namen *MyTriggers.json* in dem Verzeichnis, in dem Sie den Befehl ausgeführt haben.
2. Bearbeiten Sie die JSON-Datei in einem Texteditor und entfernen Sie den Auslöserblock für den zu löschenden Auslöser. Ersetzen Sie das `configurationId`-Paar durch ein `repositoryName`-Paar. Speichern Sie die Datei.

   Wenn Sie beispielsweise einen Trigger mit dem Namen *MyFirstTrigger*aus dem Repository `configurationId` mit `repositoryName` dem Namen entfernen möchten *MyDemoRepo*, ersetzen Sie ihn durch und entfernen Sie die Anweisung in *rotem kursivem Text*:

   ```
   {
   ```

```
    "repositoryName": "MyDemoRepo",
    "triggers": [
        {
            "destinationArn": "arn:aws:sns:us-
east-2:111122223333:MyCodeCommitTopic",
            "branches": [
                "main",
                "preprod"
            ],
            "name": "MyFirstTrigger",
            "customData": "",
            "events": [
                "all"
            ]
        },
        {
            "destinationArn": "arn:aws:lambda:us-
east-2:111122223333:function:MyCodeCommitJSFunction",
            "branches": [],
            "name": "MyLambdaTrigger",
            "events": [
                "all"
            ]
        }
    ]
}
```

3. Führen Sie am Terminal oder über die Befehlszeile den Befehl put-repository-triggers aus. Dadurch werden die Trigger für das Repository aktualisiert und der *MyFirstTrigger*Trigger gelöscht:

```
aws codecommit put-repository-triggers --repository-name MyDemoRepo
 file://MyTriggers.json
```

Dieser Befehl gibt eine Konfigurations-ID wie die folgende zurück:

```
{
    "configurationId": "0123456-I-AM-AN-EXAMPLE"
}
```

> **Note**
>
> Um alle Trigger für ein Repository mit dem Namen zu löschen *MyDemoRepo*, würde Ihre JSON-Datei etwa so aussehen:
>
> ```
> {
>     "repositoryName": "MyDemoRepo",
>     "triggers": []
> }
> ```

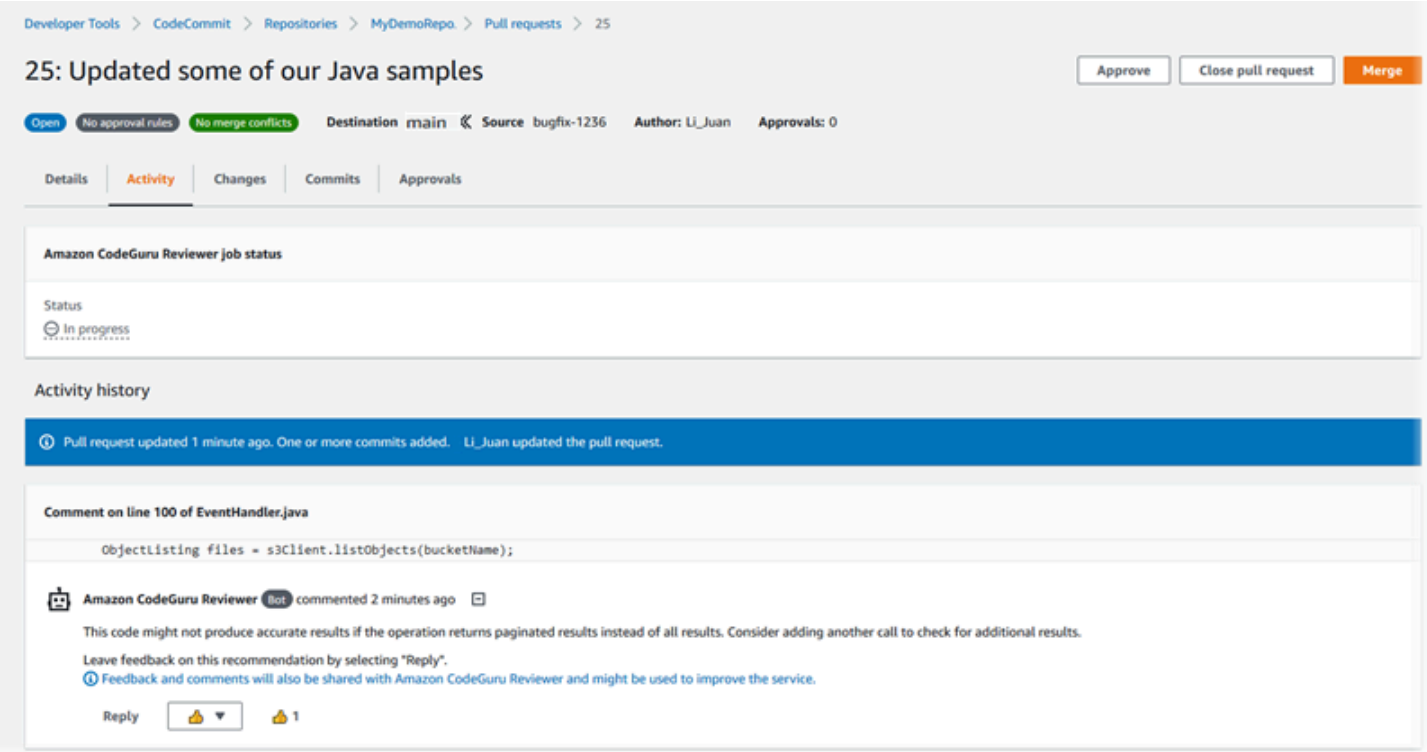
Ein AWS CodeCommit Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben

Amazon CodeGuru Reviewer ist ein automatisierter Code-Review-Service, der mithilfe von Programmanalyse und maschinellem Lernen häufig auftretende Probleme erkennt und Korrekturen in Ihrem Java- oder Python-Code empfiehlt. Sie können Repositorys in Ihrem Amazon Web Services Services-Konto mit CodeGuru Reviewer verknüpfen. Wenn Sie dies tun, erstellt der CodeGuru Prüfer eine servicebezogene Rolle, die es dem CodeGuru Prüfer ermöglicht, den Code in allen Pull-Requests zu analysieren, die nach der Zuordnung erstellt wurden.

Nachdem Sie ein Repository zugeordnet haben, analysiert und kommentiert CodeGuru Reviewer alle Probleme, die bei der Erstellung von Pull Requests festgestellt werden. Jeder Kommentar ist eindeutig als von einem CodeGuru Rezensenten mit der Bezeichnung Amazon CodeGuru Reviewer stammend gekennzeichnet. Sie können auf diese Kommentare wie auf alle anderen Kommentare in einer Pull-Anforderung reagieren und auch Feedback zur Qualität des Vorschlags übermitteln. Dieses Feedback wird an den CodeGuru Rezensenten weitergegeben und kann dazu beitragen, den Service und seine Vorschläge zu verbessern.

> **Note**
>
> In Pull-Requests, die erstellt wurden, bevor das Repository mit dem Repository verknüpft wurde, werden dir keine Kommentare von CodeGuru Reviewer angezeigt. Aus folgenden Gründen werden Kommentare in Pull-Anforderungen, die nach der Zuordnung erstellt wurden, möglicherweise nicht angezeigt:
>
> - Die Pull-Anfrage enthält keinen Java- oder Python-Code.

- CodeGuru Der Prüfer hatte nicht genug Zeit, um den Code in der Pull-Anfrage auszuführen und zu überprüfen. Dieser Vorgang kann bis zu 30 Minuten dauern. Kommentare können im Verlauf der Überprüfung erscheinen, aber das Kommentieren ist erst abgeschlossen, wenn der Jobstatus als Abgeschlossen angezeigt wird.
- CodeGuru Der Prüfer hat in der Pull-Anfrage keine Probleme im Java- oder Python-Code festgestellt.
- Der Code-Prüfungsauftrag konnte nicht ausgeführt werden. Den Status einer Überprüfung für einen Pull Request findest du auf der Registerkarte Aktivität des Pull-Requests.
- Sie sehen die Änderungen am Pull Request auf der Registerkarte Änderungen, der Pull Request wurde aktualisiert und Amazon CodeGuru Reviewer hat bei den Änderungen keine Probleme festgestellt. Kommentare von Amazon CodeGuru Reviewer werden nur dann auf der Registerkarte „Änderungen“ angezeigt, wenn sie zur letzten Version des Pull-Requests abgegeben wurden. Sie werden immer auf der Registerkarte Aktivität angezeigt.



Weitere Informationen finden Sie unter Verwenden von Pull-Anforderungen inAWS CodeCommitRepositoriesÜberprüfen einer Pull-Anforderung, und im Amazon CodeGuru Reviewer-Benutzerhandbuch.

> **Note**
>
> Sie müssen mit einem IAM-Benutzer oder einer IAM-Rolle angemeldet sein, der über ausreichende Berechtigungen verfügt, um ein Repository mit CodeGuru Reviewer zu verknüpfen oder die Zuordnung aufzuheben. Informationen zu den verwalteten Richtlinien, die diese Berechtigungen beinhalten CodeCommit , finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit und. AWS CodeCommitverwaltete Richtlinien und Amazon CodeGuru Reviewer Informationen zu den Berechtigungen und zur Sicherheit von CodeGuru Rezensenten finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.

Themen

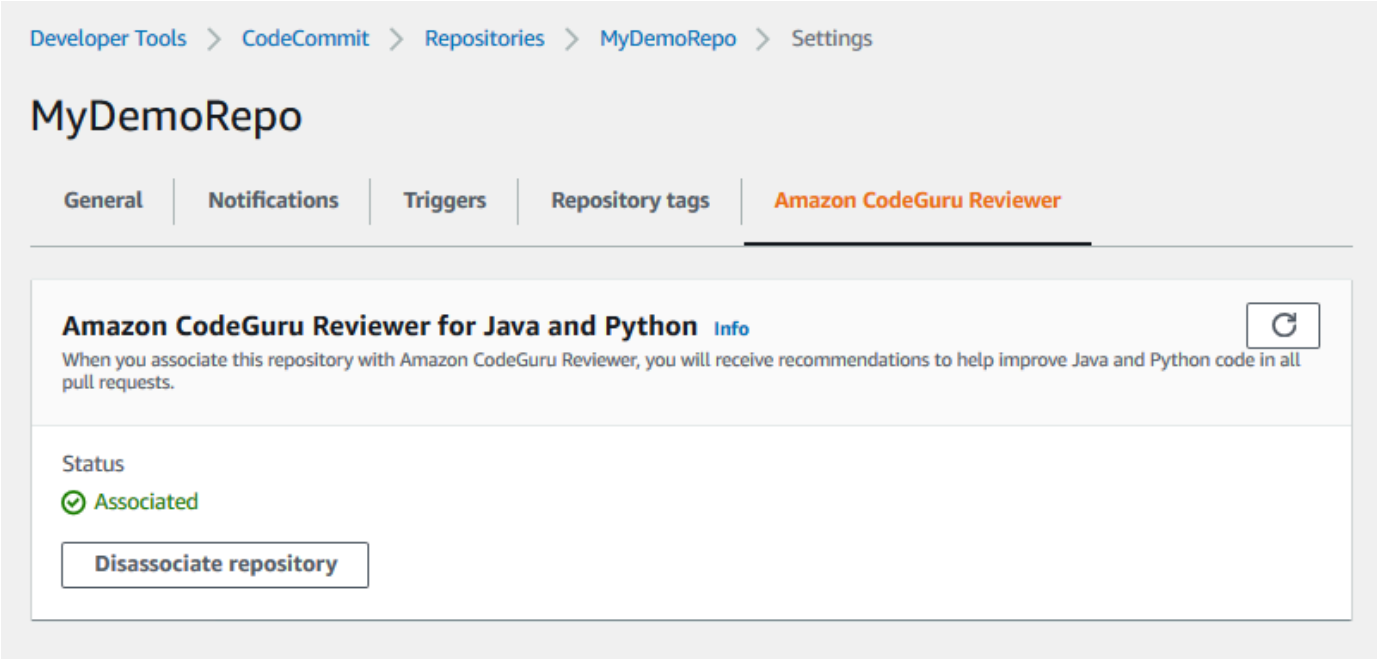
- Ordnen Sie dem CodeGuru Rezensenten ein Repository zu
- Trennen Sie die Zuordnung eines Repositorys zu Reviewer CodeGuru

Ordnen Sie dem CodeGuru Rezensenten ein Repository zu

Verwenden Sie die AWS CodeCommit Konsole, um CodeGuru Reviewer schnell ein Repository zuzuordnen. Weitere Methoden finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories den Namen des Repositorys aus, das CodeGuru Reviewer zugeordnet werden soll.
3. Wählen Sie Einstellungen und dann Amazon CodeGuru Reviewer aus.
4. Wählen Sie Associate repository (Repository zuordnen) aus.

> **Note**
>
> Es kann bis zu 10 Minuten dauern, bis ein Repository vollständig mit CodeGuru Reviewer verknüpft ist. Der Status wird nicht automatisch aktualisiert. Um den aktuellen Status anzuzeigen, wählen Sie die Schaltfläche "Refresh (Aktualisieren)".



Trennen Sie die Zuordnung eines Repositorys zu Reviewer CodeGuru

Verwenden Sie die AWS CodeCommit Konsole, um ein Repository schnell von CodeGuru Reviewer zu trennen. Weitere Methoden finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories den Namen des Repositorys aus, das Sie von CodeGuru Reviewer trennen möchten.
3. Wählen Sie Einstellungen und dann Amazon CodeGuru Reviewer aus.
4. Wählen Sie Disassociate repository (Repository-Zuordnung aufheben).

CodeCommit Repository-Details anzeigen

Sie können die AWS CodeCommit Konsole oder Git von einem lokalen Repo aus verwenden AWS CLI, das mit dem CodeCommit Repository verbunden ist, um Informationen über verfügbare Repositorys anzuzeigen.

Bevor Sie diese Anweisungen befolgen, führen Sie die Schritte unter Einrichtung aus.

Themen

- Repository-Details anzeigen (Konsole)
- CodeCommit Repository-Details anzeigen (Git)
- CodeCommit Repository-Details anzeigen (AWS CLI)

Repository-Details anzeigen (Konsole)

Verwenden Sie die AWS CodeCommit Konsole, um schnell alle Repositorys anzuzeigen, die mit Ihrem Amazon Web Services Services-Konto erstellt wurden.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Sehen Sie sich unter Repositorys die Details zu den Repositorys in der Region AWS-Region an, in der Sie angemeldet sind. Verwenden Sie die Regionsauswahl, um ein anderes AWS-Region T auszuwählen, um Repositorys in dieser Region anzuzeigen.
3. Wählen Sie den Namen des Repositorys, für das Sie weitere Details anzeigen möchten, und führen Sie dann einen der folgenden Schritte aus:

   - Wählen Sie zum Anzeigen der URL für das Klonen des Repositorys Clone URL (URL klonen) und anschließend das Protokoll aus, das Sie beim Klonen des Repositorys verwenden möchten. Dadurch wird die Klon-URL kopiert. Um sie zu überprüfen, fügen Sie sie in einen Texteditor ein.
   - Um konfigurierbare Optionen für das Repository sowie Details wie den Repository-ARN und die Repository-ID anzuzeigen, wählen Sie im Navigationsbereich Einstellungen aus.

> **Note**
>
> Wenn Sie als IAM-Benutzer angemeldet sind, können Sie Ihre Einstellungen für die Anzeige von Code und anderen Konsoleneinstellungen konfigurieren und speichern. Weitere Informationen finden Sie unter Arbeiten mit Benutzereinstellungen.

CodeCommit Repository-Details anzeigen (Git)

Um Git von einem lokalen Repo aus zu verwenden, um Details zu CodeCommit Repositorys anzuzeigen, führen Sie den git remote show Befehl aus.

Bevor du diese Schritte ausführst, verbinde das lokale Repository mit dem Repository. CodeCommit Anweisungen finden Sie unter Herstellen einer Verbindung mit einem Repository.

1. Führen Sie den git remote show ***remote-name*** Befehl aus, wobei *remote-name* der Alias des CodeCommit Repositorys ist (standardmäßig). `origin`

   > **Tip**
   >
   > Um eine Liste der CodeCommit Repository-Namen und ihrer URLs zu erhalten, führen Sie den git remote -v Befehl aus.

   Um beispielsweise Details über das CodeCommit Repository mit dem Alias anzuzeigen`origin`:

   ```
   git remote show origin
   ```

2. Für HTTPS:

   ```
   * remote origin
     Fetch URL: https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
     Push  URL: https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
     HEAD branch: (unknown)
     Remote branches:
       MyNewBranch tracked
       main tracked
     Local ref configured for 'git pull':
       MyNewBranch merges with remote MyNewBranch (up to date)
     Local refs configured for 'git push':
       MyNewBranch pushes to MyNewBranch (up to date)
       main pushes to main (up to date)
   ```

   Für SSH:

   ```
   * remote origin
     Fetch URL: ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
     Push  URL: ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
     HEAD branch: (unknown)
     Remote branches:
       MyNewBranch tracked
       main tracked
     Local ref configured for 'git pull':
   ```

```
    MyNewBranch merges with remote MyNewBranch (up to date)
  Local refs configured for 'git push':
    MyNewBranch pushes to MyNewBranch (up to date)
    main pushes to main (up to date)
```

> **Tip**
>
> Um die SSH-Schlüssel-ID für Ihren IAM-Benutzer nachzuschlagen, öffnen Sie die IAM-Konsole und erweitern Sie auf der Seite mit den IAM-Benutzerdetails die Option Sicherheitsanmeldedaten. Die SSH-Schlüssel-ID finden Sie unter SSH-Schlüssel für. AWS CodeCommit

Weitere Optionen findest du in deiner Git-Dokumentation.

CodeCommit Repository-Details anzeigen (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

Führen Sie die folgenden Befehle aus, AWS CLI um die Repository-Details anzuzeigen:

- Um eine Liste der CodeCommit Repository-Namen und der entsprechenden IDs anzuzeigen, führen Sie list-repositories aus.
- Um Informationen zu einem einzelnen CodeCommit Repository anzuzeigen, führen Sie get-repository aus.
- Um Informationen über mehrere Repositorys in anzuzeigen CodeCommit, führen Sie den Befehl aus. batch-get-repositories

So zeigen Sie eine Liste der CodeCommit-Repositorys an

1. Führen Sie den Befehl list-repositories aus:

   ```
   aws codecommit list-repositories
   ```

   Sie können den optionalen Parameter `--sort-by` oder `--order` verwenden, um die Sortierreihenfolge der zurückgegebenen Informationen zu ändern.

2. Bei Erfolg gibt dieser Befehl ein `repositories` Objekt aus, das die Namen und IDs aller Repositorys enthält, die dem Amazon Web Services Services-Konto CodeCommit zugeordnet sind.

   Es folgt eine Beispielausgabe basierend auf dem vorangehenden Befehl:

```
{
    "repositories": [
        {
           "repositoryName": "MyDemoRepo",
           "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE"
        },
        {
           "repositoryName": "MyOtherDemoRepo",
           "repositoryId": "cfc29ac4-b0cb-44dc-9990-f6f51EXAMPLE"
        }
    ]
}
```

Um Details zu einem einzelnen CodeCommit Repository anzuzeigen

1. Führen Sie den get-repository Befehl aus und geben Sie den Namen des CodeCommit Repositorys mit der `--repository-name` Option an.

   > **Tip**
   >
   > Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl list-repositories aus.

   Um beispielsweise Details zu einem CodeCommit Repository mit dem Namen anzuzeigen: `MyDemoRepo`

```
aws codecommit get-repository --repository-name MyDemoRepo
```

2. Ist der Befehl erfolgreich, wird ein `repositoryMetadata`-Objekt mit den folgenden Informationen ausgegeben:

   - Name des Repositorys (`repositoryName`)

- Beschreibung des Repositorys (`repositoryDescription`)
- Eindeutige, systemgenerierte ID des Repositorys (`repositoryId`)
- Die ID des Amazon Web Services Services-Kontos, das dem Repository zugeordnet ist (`accountId`).

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
        "repositoryMetadata": {
            "creationDate": 1429203623.625,
            "defaultBranch": "main",
            "repositoryName": "MyDemoRepo",
            "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/
MyDemoRepo",
            "lastModifiedDate": 1430783812.0869999,
            "repositoryDescription": "My demonstration repository",
            "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/repos/
MyDemoRepo",
            "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
            "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
            "accountId": "111111111111"
        }
}
```

So zeigen Sie Details zu mehreren CodeCommit -Repositorys; an

1. Führen Sie den Befehl batch-get-repositories mit der Option `--repository-names` aus. Fügen Sie zwischen jedem CodeCommit Repository-Namen ein Leerzeichen ein.

> **Tip**
>
> Um die Namen der Repositorys abzurufen CodeCommit, führen Sie den Befehl list-repositories aus.

Um beispielsweise Details zu zwei CodeCommit Repositorys mit dem Namen und anzuzeigen: `MyDemoRepo` `MyOtherDemoRepo`

```
aws codecommit batch-get-repositories --repository-names MyDemoRepo MyOtherDemoRepo
```

2. Ist der Befehl erfolgreich, wird ein Objekt mit den folgenden Informationen ausgegeben:

   - Eine Liste aller CodeCommit Repositorys, die nicht gefunden werden konnten ()`repositoriesNotFound`.
   - Eine Liste von CodeCommit Repositorien ()`repositories`. Auf jeden CodeCommit Repository-Namen folgt:
     - Beschreibung des Repositorys (`repositoryDescription`)
     - Eindeutige, systemgenerierte ID des Repositorys (`repositoryId`)
     - Die ID des Amazon Web Services Services-Kontos, das dem Repository zugeordnet ist (`accountId`).

   Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
        "repositoriesNotFound": [],
        "repositories": [
             {
                "creationDate": 1429203623.625,
                "defaultBranch": "main",
                "repositoryName": "MyDemoRepo",
                "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/
repos/MyDemoRepo",
                "lastModifiedDate": 1430783812.0869999,
                "repositoryDescription": "My demonstration repository",
                "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/
repos/MyDemoRepo",
                "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE",
                "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
                "accountId": "111111111111"
            },
            {
                "creationDate": 1429203623.627,
                "defaultBranch": "main",
                "repositoryName": "MyOtherDemoRepo",
                "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/
repos/MyOtherDemoRepo",
                "lastModifiedDate": 1430783812.0889999,
```

```
                "repositoryDescription": "My other demonstration repository",
                "cloneUrlHttp": "https://codecommit.us-east-2.amazonaws.com/v1/
repos/MyOtherDemoRepo",
                "repositoryId": "cfc29ac4-b0cb-44dc-9990-f6f51EXAMPLE",
                "Arn": "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo",
                "accountId": "111111111111"
            }
        ],
        "repositoriesNotFound": []
    }
```

AWS CodeCommit Repository-Einstellungen ändern

Sie können die Konsole AWS CLI und die AWS CodeCommit Konsole verwenden, um die Einstellungen eines CodeCommit Repositorys zu ändern, z. B. dessen Beschreibung oder seinen Namen.

> **Important**
>
> Wenn der Name eines Repositorys geändert wird, können möglicherweise lokale Repos nicht mehr aufgerufen werden, die den alten Namen in ihrer Remote-URL verwenden. Führen Sie den Befehl git remote set-url aus, um die Remote-URL für die Verwendung des neuen Repository-Namens zu aktualisieren.

Themen

- Repository-Einstellungen ändern (Konsole)
- AWS CodeCommit Repository-Einstellungen ändern ()AWS CLI

Repository-Einstellungen ändern (Konsole)

Gehen Sie wie folgt vor, um mit der AWS CodeCommit Konsole die Einstellungen eines CodeCommit Repositorys zu ändern. AWS CodeCommit

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.

2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, bei dem Sie Einstellungen ändern möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Um den Namen des Repositorys zu ändern, geben Sie unter Repository name (Repository-Name) in das Textfeld Name einen neuen Namen ein und klicken Sie auf Save (Speichern). Wenn Sie dazu aufgefordert werden, überprüfen Sie Ihre Wahl.

   Important

   Wenn Sie den Namen des AWS CodeCommit Repositorys ändern, ändern sich auch die SSH- und HTTPS-URLs, die Benutzer benötigen, um sich mit dem Repository zu verbinden. Benutzer können erst eine Verbindung mit diesem Repository herstellen, wenn sie ihre Verbindungseinstellungen aktualisiert haben. Da sich der ARN des Repositorys ändert, werden durch eine Änderung des Repository-Namens auch alle IAM-Benutzerrichtlinien ungültig, die auf dem ARN dieses Repositorys basieren.
   Um nach einer Namensänderung eine Verbindung mit dem Repository herzustellen, muss der Benutzer mit dem git remote set-url die neue URL angeben. Wenn Sie beispielsweise den Namen des Repositorys von MyDemoRepo in ändern würden, würden Benutzer MyRenamedDemoRepo, die HTTPS verwenden, um eine Verbindung zum Repository herzustellen, den folgenden Git-Befehl ausführen:

   ```
   git remote set-url origin https://git-codecommit.us-east-2.amazonaws.com/v1/
   repos/MyRenamedDemoRepo
   ```

   Benutzer, die mit SSH eine Verbindung mit dem Repository herstellen, führen den folgenden Git-Befehl aus:

   ```
   git remote set-url origin ssh://git-codecommit.us-east-2.amazonaws.com/v1/
   repos/MyRenamedDemoRepo
   ```

   Weitere Optionen findest du in deiner Git-Dokumentation.

5. Um die Beschreibung des Repositorys zu ändern, geben Sie den Text in das Textfeld Description (Beschreibung) ein und klicken Sie auf Save (Speichern).

> **Note**
>
> Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie ein Anwendungsentwickler sind, der die `BatchGetRepositories` APIs `GetRepository` oder verwendet und planen, das Repository-Beschreibungsfeld in einem Webbrowser anzuzeigen, finden Sie weitere Informationen in der CodeCommit API-Referenz.

6. Zum Ändern des Standard-Branches wählen Sie unter Default branch (Standard-Branch) die Branch-Dropdown-Liste und wählen Sie einen anderen Branch aus. Wählen Sie Speichern.
7. Um den AWS KMS Verschlüsselungsschlüssel zu ändern, der zum Verschlüsseln und Entschlüsseln von Daten im Repository verwendet wird, wählen Sie unter Repository-Verschlüsselungsschlüssel entweder Von AWS verwalteter Schlüsseloder Vom Kunden verwalteter Schlüssel aus, um den Typ des zu verwendenden Schlüssels anzugeben. Wenn Sie einen vom Kunden verwalteten Schlüssel wählen, geben Sie den ARN des Schlüssels ein. Wählen Sie Speichern.
8. Um das Repository zu löschen, klicken Sie auf Delete repository. Geben Sie in das Feld neben Type the name of the repository to confirm deletion (Namen des Repositorys eingeben, um das Löschen zu bestätigen) die Zeichenfolge **`delete`** ein und klicken Sie dann auf Delete (Löschen).

> **Important**
>
> Nachdem Sie dieses Repository gelöscht haben AWS CodeCommit, können Sie es nicht mehr in ein lokales oder gemeinsam genutztes Repository klonen. Sie werden auch nicht mehr in der Lage sein, Daten aus einem lokalen oder gemeinsam genutzten Repo daraus abzurufen oder Daten dorthin zu übertragen. Diese Aktion kann nicht rückgängig gemacht werden.

AWS CodeCommit Repository-Einstellungen ändern ()AWS CLI

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

Um die Einstellungen eines CodeCommit Repositorys in AWS CLI zu ändern AWS CodeCommit, führen Sie einen oder mehrere der folgenden Befehle aus:

- update-repository-descriptionum die Beschreibung eines CodeCommit Repositorys zu ändern.
- update-repository-nameum den Namen eines CodeCommit Repositorys zu ändern.

Um die Beschreibung eines CodeCommit Repositorys zu ändern

1. Führen Sie den Befehl update-repository-description aus und geben Sie Folgendes an:

   - Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).

     > **Tip**
     >
     > Führen Sie den list-repositories Befehl aus, um den Namen des CodeCommit Repositorys abzurufen.

   - Neue Beschreibung des Repositorys (mit der Option `--repository-description`).

     > **Note**
     >
     > Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie ein Anwendungsentwickler sind, der die `BatchGetRepositories` APIs `GetRepository` oder verwendet und planen, das Repository-Beschreibungsfeld in einem Webbrowser anzuzeigen, finden Sie weitere Informationen in der CodeCommit API-Referenz.

   Um beispielsweise die Beschreibung für das CodeCommit Repository mit dem Namen wie folgt `MyDemoRepo` zu ändern`This description was changed`:

   ```
   aws codecommit update-repository-description --repository-name MyDemoRepo --
   repository-description "This description was changed"
   ```

   Dieser Befehl liefert nur eine Ausgabe, wenn Fehler aufgetreten sind.

2. Um die geänderte Beschreibung zu überprüfen, führen Sie den get-repository Befehl aus und geben Sie den Namen des CodeCommit Repositorys an, dessen Beschreibung Sie mit der `--repository-name` Option geändert haben.

   Die Ausgabe des Befehls zeigt den geänderten Text in `repositoryDescription`.

Um den Namen eines CodeCommit Repositorys zu ändern

1. Führen Sie den Befehl update-repository-name aus und geben Sie Folgendes an:

   - Der aktuelle Name des CodeCommit Repositorys (mit der `--old-name` Option).

     > **Tip**
     >
     > Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl list-repositories aus.

   - Der neue Name des CodeCommit Repositorys (mit der `--new-name` Option).

   So ändern Sie z. B. das Repository mit dem Namen `MyDemoRepo` in `MyRenamedDemoRepo`:

   ```
   aws codecommit update-repository-name --old-name MyDemoRepo --new-name
    MyRenamedDemoRepo
   ```

   Dieser Befehl liefert nur eine Ausgabe, wenn Fehler aufgetreten sind.

   > **Important**
   >
   > Wenn Sie den Namen des AWS CodeCommit Repositorys ändern, werden die SSH- und HTTPS-URLs geändert, die Benutzer benötigen, um eine Verbindung zum Repository herzustellen. Benutzer können erst eine Verbindung mit diesem Repository herstellen, nachdem sie ihre Verbindungseinstellungen aktualisiert haben. Da sich der ARN des Repositorys ändert, macht eine Änderung des Repository-Namens außerdem alle IAM-Benutzerrichtlinien ungültig, die auf dem ARN dieses Repositorys basieren.

2. Um den geänderten Namen zu überprüfen, führen Sie den Befehl list-repositories aus und prüfen Sie die Liste mit den Repository-Namen.

Synchronisieren Sie Änderungen zwischen einem lokalen Repository und einem Repository AWS CodeCommit

Sie verwenden Git, um Änderungen zwischen einem lokalen Repo und dem mit dem lokalen Repo verbundenen CodeCommit Repository zu synchronisieren.

Um Änderungen vom lokalen Repo in das CodeCommit Repository zu übertragen, führe den Befehl aus. git push ***remote-name branch-name***

Um Änderungen am lokalen Repo aus dem CodeCommit Repository abzurufen, führen Sie den Befehl aus. git pull ***remote-name branch-name***

Sowohl beim Push als auch beim Pull ist *remote-name* der Spitzname, den das lokale Repo für das Repository verwendet. CodeCommit *Branch-Name ist der Name* des Branches im CodeCommit Projektarchiv, zu dem ein Push oder ein Pull ausgeführt werden soll.

> **Tip**
>
> Um den Spitznamen zu erhalten, den das lokale Repo für das CodeCommit Repository verwendet, führen Sie den Befehl aus. git remote Um eine Liste der Branch-Namen zu erhalten, führen Sie den Befehl git branch aus. Ein Sternchen (*) wird neben dem aktuell geöffneten Branch angezeigt. (Sie können auch git status ausführen, um den aktuellen Branch-Namen anzuzeigen.)

> **Note**
>
> Wenn Sie das Repository geklont haben, ist *remote-name aus der Sicht des lokalen Repositorys nicht der Name* des Repositorys. CodeCommit Beim Klonen eines Repositorys wird *remote-name* automatisch auf `origin` festgelegt.

Um beispielsweise Änderungen vom lokalen Repo in den `main` Zweig im Repository mit dem Spitznamen zu übertragen: CodeCommit `origin`

```
git push origin main
```

Um auf ähnliche Weise Änderungen am lokalen Repo aus dem `main` Branch im CodeCommit Repository mit dem Spitznamen abzurufen: `origin`

```
git pull origin main
```

> **Tip**
>
> Durch Hinzufügen der Option `-u` zum Befehl git push legen Sie Upstream-Nachverfolgungsdaten fest. Wenn Sie beispielsweise git push -u origin main ausführen, können Sie zukünftig git push und git pull ohne *`remote-name branch-name`* ausführen. Führen Sie git remote show ***`remote-name`*** aus (z. B. git remote show origin), um Upstream-Nachverfolgungsdaten abzurufen.

Weitere Optionen findest du in deiner Git-Dokumentation.

Push Commits in ein zusätzliches Git-Repository

Sie können Ihr lokales Repository so konfigurieren, dass Änderungen mithilfe von Push an zwei Remote-Repositorys übertragen werden. Sie könnten beispielsweise Ihre vorhandene Git-Repository-Lösung weiterhin verwenden, während Sie AWS CodeCommit testen. Folgen Sie diesen grundlegenden Schritten, um Änderungen in Ihrem lokalen Repository in ein separates Git-Repository zu CodeCommit übertragen.

> **Tip**
>
> Wenn Sie kein Git-Repository haben, können Sie ein leeres in einem anderen Dienst als diesem erstellen CodeCommit und dann Ihr CodeCommit Repository dorthin migrieren. Hierzu führen Sie Schritte aus, die den Anweisungen unter Migration zu CodeCommit weitestgehend entsprechen.

1. Wechseln Sie über das Befehlszeilen- oder Terminalfenster zu Ihrem lokalen Repository-Verzeichnis und führen Sie den Befehl git remote -v aus. Die Ausgabe sollte folgendermaßen oder ähnlich aussehen:

   Für HTTPS:

   ```
   origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
   origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
   ```

   Für SSH:

```
origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
```

2. Führen Sie den git remote set-url --add --push origin ***git-repository-name*** Befehl aus, in dem *git-repository-name*die URL und der Name des Git-Repositorys stehen, in dem Sie Ihren Code hosten möchten. Hierdurch wird das Push-Ziel von `origin` in dieses Git-Repository geändert.

> **Note**
>
> Mit dem Befehl git remote set-url --add --push wird die Standard-URL für Pushes überschrieben, sodass Sie diesen Befehl zweimal ausführen müssen, wie weiter unten erläutert.

Der folgende Befehl ändert beispielsweise den Push of origin in *some-URL/:* MyDestinationRepo

```
git remote set-url --add --push origin some-URL/MyDestinationRepo
```

Mit diesem Befehl wird kein Inhalt zurückgegeben.

> **Tip**
>
> Wenn Sie einen Push an ein Git-Repository ausführen, für das Anmeldeinformationen erforderlich sind, achten Sie darauf, dass Sie diese in einem Hilfsprogramm für Anmeldeinformationen oder in der Konfiguration der Zeichenfolge *some-URL* konfigurieren. Andernfalls schlagen Ihre Pushes an dieses Repository fehl.

3. Führen Sie den Befehl git remote -v erneut aus. Die Befehlsausgabe sollte der folgenden ähneln:

Für HTTPS:

```
origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin  some-URL/MyDestinationRepo (push)
```

Für SSH:

```
origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
origin  some-URL/MyDestinationRepo (push)
```

4. Fügen Sie jetzt das CodeCommit Repository hinzu. Führen Sie git remote set-url --add --push origin es erneut aus, diesmal mit der URL und dem Repository-Namen Ihres CodeCommit Repositorys.

   Der folgende Befehl fügt beispielsweise den Push of origin zu https://git-codecommit.us-east-2.amazonaws.com/v1/repos/ hinzuMyDemoRepo:

   Für HTTPS:

   ```
   git remote set-url --add --push origin https://git-codecommit.us-
   east-2.amazonaws.com/v1/repos/MyDemoRepo
   ```

   Für SSH:

   ```
   git remote set-url --add --push origin ssh://git-codecommit.us-
   east-2.amazonaws.com/v1/repos/MyDemoRepo
   ```

   Mit diesem Befehl wird kein Inhalt zurückgegeben.

5. Führen Sie den Befehl git remote -v erneut aus. Die Befehlsausgabe sollte der folgenden ähneln:

   Für HTTPS:

   ```
   origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
   origin  some-URL/MyDestinationRepo (push)
   origin  https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
   ```

   Für SSH:

   ```
   origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (fetch)
   origin  some-URL/MyDestinationRepo (push)
   origin  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo (push)
   ```

Sie haben jetzt zwei Git-Repositorys als Ziel für Ihre Pushs, aber Ihre Pushs gehen zuerst zu *some-URL*/. MyDestinationRepo Wenn der Push an dieses Repository fehlschlägt, werden Ihre Commits nicht mithilfe von Push an die Repositorys übertragen.

> **Tip**
>
> Wenn für das andere Repository Anmeldeinformationen erforderlich sind, die Sie manuell eingeben möchten, sollten Sie die Reihenfolge der Pushs so ändern, dass Sie zuerst zu pushen beginnen. CodeCommit Führen Sie den Befehl git remote set-url --delete aus, um das Repository zu löschen, an das Pushes zuerst ausgeführt werden. Führen Sie anschließend den Befehl git remote set-url --add aus, um das Repository erneut hinzuzufügen, sodass es zum zweiten Push-Ziel in der Liste wird.
> Weitere Optionen findest du in deiner Git-Dokumentation.

6. Um zu überprüfen, ob Sie nun einen Push an beide Remote-Repositorys ausführen, erstellen Sie mit einem Texteditor die folgende Textdatei in Ihrem lokalen Repository:

```
bees.txt
-------
Bees are flying insects closely related to wasps and ants, and are known for their
 role in pollination and for producing honey and beeswax.
```

7. Führen Sie git add aus, um die Änderung per Stage an Ihr lokales Repository zu übertragen:

```
git add bees.txt
```

8. Führen Sie git commit aus, um die Änderung per Commit an Ihr lokales Repository zu übertragen:

```
git commit -m "Added bees.txt"
```

9. Um den Commit vom lokalen Repository mithilfe von Push in Ihre Remote-Repositorys zu übertragen, führen Sie den Befehl git push -u ***remote-name branch-name*** aus, wobei *remote-name* der vom lokalen Repository für die Remote-Repositorys verwendete Name ist und *branch-name* für den Namen des Branches steht, der in das Repository übertragen werden soll.

> **Tip**
>
> Sie müssen die Option `-u` nur für die erste Push-Ausführung verwenden. Dann werden die Upstream-Nachverfolgungsdaten festgelegt.

Beispielsweise wird durch die Ausführung von git push -u origin main deutlich, dass der Push an beide Remote-Repositorys in den erwarteten Branches übertragen wurde. Die Ausgabe entspricht weitgehend der folgenden:

Für HTTPS:

```
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 5.61 KiB | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To some-URL/MyDestinationRepo
   a5ba4ed..250f6c3  main -> main
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 5.61 KiB | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote:
To https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
   a5ba4ed..250f6c3  main -> main
```

Für SSH:

```
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 5.61 KiB | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
To some-URL/MyDestinationRepo
   a5ba4ed..250f6c3  main -> main
Counting objects: 5, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (3/3), done.
```

```
Writing objects: 100% (3/3), 5.61 KiB | 0 bytes/s, done.
Total 3 (delta 1), reused 0 (delta 0)
remote:
To ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
   a5ba4ed..250f6c3  main -> main
```

Weitere Optionen findest du in deiner Git-Dokumentation.

Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen

Sie können den Zugriff auf CodeCommit Repositorys für IAM-Benutzer und -Gruppen in einem anderen Konto konfigurieren. AWS Dieser Vorgang wird oft als kontoübergreifender Zugriff bezeichnet. Dieser Abschnitt enthält Beispiele und step-by-step Anweisungen für die Konfiguration des kontoübergreifenden Zugriffs auf ein Repository mit dem Namen *MySharedDemoRepo*in der Region USA Ost (Ohio) in einem AWS Konto (als AccountA bezeichnet) für IAM-Benutzer, die zu einer IAM-Gruppe gehören, die *DevelopersWithCrossAccountRepositoryAccess*in einem anderen AWS Konto benannt ist (als AccountB bezeichnet).

Dieser Abschnitt ist in drei Teile untergliedert:

- Aktionen für den Administrator in Konto A
- Aktionen für den Administrator in Konto B
- Aktionen für den Repository-Benutzer in Konto B

So konfigurieren den kontoübergreifenden Zugriff:

- Der Administrator in AccountA meldet sich als IAM-Benutzer mit den erforderlichen Berechtigungen an, um Repositorys in IAM zu erstellen und zu verwalten und Rollen in CodeCommit IAM zu erstellen. Wenn Sie verwaltete Richtlinien verwenden, wenden Sie IAM FullAccess und AWSCodeCommitFullAccess auf diesen IAM-Benutzer an.

  Die Beispielkonto-ID für AccountA lautet *111122223333*.
- Der Administrator in AccountB meldet sich als IAM-Benutzer mit den erforderlichen Berechtigungen an, um IAM-Benutzer und -Gruppen zu erstellen und zu verwalten und Richtlinien für Benutzer und Gruppen zu konfigurieren. Wenn Sie verwaltete Richtlinien verwenden, wenden Sie IAM auf diesen IAM-Benutzer FullAccess an.

Die Beispielkonto-ID für AccountB ist *888888888888*.

- Um die Aktivitäten eines Entwicklers nachzuahmen, meldet sich der Repository-Benutzer in AccountB als IAM-Benutzer an, der Mitglied der IAM-Gruppe ist, die erstellt wurde, um den Zugriff auf das CodeCommit Repository in AccountA zu ermöglichen. Für dieses Konto müssen folgende Elemente konfiguriert sein:
  - AWS Zugriff auf die Managementkonsole.
  - Ein Zugriffsschlüssel und ein geheimer Schlüssel, die verwendet werden, wenn eine Verbindung zu AWS Ressourcen hergestellt wird, sowie der ARN der Rolle, die beim Zugriff auf Repositorys in AccountA verwendet werden soll.
  - Das git-remote-codecommit-Dienstprogramm auf dem lokalen Computer, auf dem das Repository geklont wird. Dieses Dienstprogramm erfordert Python und dessen Installationsprogramm „pip“. Sie können das Dienstprogramm von git-remote-codecommit auf der Python Package Index-Website herunterladen.

  Weitere Informationen finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit und IAM-Benutzer.

Themen

- Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountA
- Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountB
- Kontoübergreifender Repository-Zugriff: Aktionen für den Repository-Benutzer in AccountB

Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountA

Um Benutzern oder Gruppen in Konto B den Zugriff auf ein Repository in Konto A zu erlauben, muss der Administrator von Konto A folgende Schritte ausführen:

- Erstellen einer Richtlinie in Konto A, die Zugriff auf das Repository gewährt
- Erstellen Sie eine Rolle in AccountA, die von IAM-Benutzern und -Gruppen in AccountB übernommen werden kann.
- Fügen Sie der Rolle die -Richtlinie an.

In den folgenden Abschnitten werden die Schritte dazu mit Beispielen beschrieben.

Themen

- Schritt 1: Erstellen Sie eine Richtlinie für den Repository-Zugriff in AccountA
- Schritt 2: Erstellen Sie eine Rolle für den Repository-Zugriff in AccountA

Schritt 1: Erstellen Sie eine Richtlinie für den Repository-Zugriff in AccountA

Sie können eine Richtlinie in Konto A erstellen, die Zugriff auf das Repository in Konto B gewährt. Abhängig von der Zugriffsebene, die Sie zulassen möchten, führen Sie einen der folgenden Schritte aus:

- Konfigurieren Sie die Richtlinie so, dass Benutzern in Konto B Zugriff auf ein bestimmtes Repository gewährt wird. Erlauben Sie ihnen aber nicht, eine Liste aller Repositorys in Konto A anzuzeigen.
- Konfigurieren Sie zusätzlichen Zugriff, um Benutzern in Konto B die Auswahl des Repositorys aus einer Liste aller Repositorys in Konto A zu erlauben.

So erstellen Sie eine Richtlinie für den Repository-Zugriff

1. Melden Sie sich bei der AWS Management Console als IAM-Benutzer mit Berechtigungen zum Erstellen von Richtlinien in AccountA an.
2. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
3. Wählen Sie im Navigationsbereich Policies aus.
4. Wählen Sie Richtlinie erstellen aus.
5. Wählen Sie die Registerkarte JSON aus und kopieren Sie das folgende JSON-Richtliniendokument in das JSON-Textfeld. Ersetzen Sie *us-east-2* durch das AWS-Region für das Repository, *111122223333* durch die Konto-ID für AccountA und *MySharedDemoRepo*durch den Namen für Ihr Repository in AccountA: CodeCommit

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

```
            "codecommit:Create*",
            "codecommit:DeleteBranch",
            "codecommit:Get*",
            "codecommit:List*",
            "codecommit:Describe*",
            "codecommit:Put*",
            "codecommit:Post*",
            "codecommit:Merge*",
            "codecommit:Test*",
            "codecommit:Update*",
            "codecommit:GitPull",
            "codecommit:GitPush"
        ],
        "Resource": [
            "arn:aws:codecommit:us-east-2:111122223333:MySharedDemoRepo"
        ]
    }
]
}
```

Wenn Sie möchten, dass Benutzer, die diese Rolle übernehmen, eine Liste von Repositorys auf der CodeCommit Konsolen-Startseite einsehen können, fügen Sie der Richtlinie eine zusätzliche Erklärung hinzu, die wie folgt lautet:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codecommit:BatchGet*",
                "codecommit:Create*",
                "codecommit:DeleteBranch",
                "codecommit:Get*",
                "codecommit:List*",
                "codecommit:Describe*",
                "codecommit:Put*",
                "codecommit:Post*",
                "codecommit:Merge*",
                "codecommit:Test*",
                "codecommit:Update*",
                "codecommit:GitPull",
                "codecommit:GitPush"
```

```
            ],
            "Resource": [
                "arn:aws:codecommit:us-east-2:111122223333:MySharedDemoRepo"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "codecommit:ListRepositories",
            "Resource": "*"
        }
    ]
}
```

Dieser Zugriff macht es für Benutzer, die diese Rolle übernehmen, mit dieser Richtlinie einfacher, das Repository zu finden, auf das sie Zugriff haben. Sie können den Namen des Repositorys aus der Liste auswählen und gelangen dann auf die Startseite des freigegebenen Repositorys (Code). Die Benutzer können nicht auf andere in der Liste angezeigten Repositorys zugreifen, sie können aber die Repositorys in Konto A auf der Seite Dashboard anzeigen.

Wenn Sie nicht möchten, dass Benutzer, die diese Rolle übernehmen, eine Liste aller Repositorys in AccountA einsehen können, verwenden Sie das erste Richtlinienbeispiel, stellen Sie jedoch sicher, dass Sie diesen Benutzern einen direkten Link zur Startseite des gemeinsam genutzten Repositorys in der CodeCommit Konsole senden.

6. Wählen Sie Richtlinie prüfen. Der Richtlinienvalidator meldet Syntaxfehler (wenn Sie beispielsweise vergessen, die Amazon Web Services Services-Konto-ID und den Repository-Namen durch Ihre Amazon Web Services Services-Konto-ID und Ihren Repository-Namen zu ersetzen).
7. Geben Sie auf der Seite „Richtlinie überprüfen“ einen Namen für die Richtlinie ein (z. B. *CrossAccountAccessForMySharedDemoRepo*). Sie können optional auch eine Beschreibung für die Richtlinie eingeben. Wählen Sie Richtlinie erstellen aus.

Schritt 2: Erstellen Sie eine Rolle für den Repository-Zugriff in AccountA

Nachdem Sie eine Richtlinie konfiguriert haben, erstellen Sie eine Rolle, die IAM-Benutzer und -Gruppen in AccountB annehmen können, und fügen Sie die Richtlinie dieser Rolle hinzu.

So erstellen Sie eine Rolle für den Repository-Zugriff

1. Wählen Sie in der IAM-Konsole Roles (Rollen) aus.

2. Wählen Sie Rolle erstellen aus.
3. Wählen Sie ein anderes Amazon Web Services Services-Konto.
4. Geben Sie unter Konto-ID die Amazon Web Services Services-Konto-ID für AccountB ein (z. B. *888888888888*). Wählen Sie Weiter: Berechtigungen aus.
5. Wählen Sie unter Zugriffsrichtlinien anhängen die Richtlinie aus, die Sie im vorherigen Verfahren erstellt haben (). *CrossAccountAccessForMySharedDemoRepo* Wählen Sie Weiter: Prüfen aus.
6. Geben Sie im Feld Rollenname einen Namen für die Rolle ein (z. B. *MyCrossAccountRepositoryContributorRole*). Sie können optional auch eine Beschreibung eingeben, um anderen den Zweck dieser Rolle mitzuteilen.
7. Wählen Sie Rolle erstellen aus.
8. Öffnen Sie die Rolle, die Sie gerade erstellt haben, und kopieren Sie den ARN der Rolle (z. B. `arn:aws:iam::`*111122223333*`:role/`*MyCrossAccountRepositoryContributorRole*). Sie müssen diesen ARN dem Administrator von Konto B mitteilen.

Kontoübergreifender Zugriff auf das Repository: Aktionen für den Administrator in AccountB

Um Benutzern oder Gruppen in Konto B den Zugriff auf ein Repository in Konto A zu erlauben, muss der Administrator von Konto B eine Gruppe in Konto B erstellen. Diese Gruppe muss mit einer Richtlinie konfiguriert werden, die es den Gruppenmitgliedern erlaubt, die vom Administrator von Konto A erstellte Rolle zu übernehmen.

In den folgenden Abschnitten werden die Schritte dazu mit Beispielen beschrieben.

Themen

- Schritt 1: Erstellen Sie eine IAM-Gruppe für den Repository-Zugriff für AccountB-Benutzer
- Schritt 2: Erstellen Sie eine Richtlinie und fügen Sie Benutzer zur IAM-Gruppe hinzu

Schritt 1: Erstellen Sie eine IAM-Gruppe für den Repository-Zugriff für AccountB-Benutzer

Die einfachste Methode, um zu verwalten, welche IAM-Benutzer in AccountB auf das AccountA-Repository zugreifen können, besteht darin, eine IAM-Gruppe in AccountB zu erstellen, die über die

Berechtigung verfügt, die Rolle in AccountA anzunehmen, und dann die IAM-Benutzer zu dieser Gruppe hinzuzufügen.

So erstellen Sie eine Gruppe für den kontoübergreifenden Repository-Zugriff

1. Melden Sie sich bei der AWS Management Console als IAM-Benutzer mit den erforderlichen Berechtigungen an, um IAM-Gruppen und -Richtlinien zu erstellen und IAM-Benutzer in AccountB zu verwalten.
2. Öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
3. Wählen Sie in der IAM-Konsole Gruppen aus.
4. Wählen Sie Create New Group (Neue Gruppe erstellen).
5. Geben Sie im Feld Gruppenname einen Namen für die Gruppe ein (z. B. *DevelopersWithCrossAccountRepositoryAccess*). Wählen Sie Next Step (Weiter) aus.
6. In Richtlinien anfügen wählen Sie Next Stepaus. Sie erstellen die kontoübergreifende Richtlinie im nächsten Verfahren. Schließen Sie die Erstellung der Gruppe ab.

Schritt 2: Erstellen Sie eine Richtlinie und fügen Sie Benutzer zur IAM-Gruppe hinzu

Nachdem Sie nun über eine Gruppe verfügen, erstellen Sie die Richtlinie, die es den Mitgliedern dieser Gruppe erlaubt, die Rolle zu übernehmen, die ihnen Zugriff auf das Repository in Konto A gewährt. Fügen Sie dann der Gruppe die IAM-Benutzer in AccountB hinzu, denen Sie den Zugriff in AccountA gewähren möchten.

So erstellen Sie eine Richtlinie für die Gruppe und fügen dieser Benutzer hinzu

1. Wählen Sie in der IAM-Konsole Gruppen und dann den Namen der Gruppe aus, die Sie gerade erstellt haben (z. B.). *DevelopersWithCrossAccountRepositoryAccess*
2. Wählen Sie die Registerkarte Berechtigungen. Erweitern Sie Inline Policies (Eingebundene Richtlinien) und wählen Sie den Link, um eine eingebundene Richtlinie zu erstellen. (Wenn Sie eine Gruppe konfigurieren, die bereits über eine eingebundene Richtlinie verfügt, wählen Sie Create Group Policy (Gruppenrichtlinie erstellen).)
3. Wählen Sie Custom Policy und dann Select aus.
4. Geben Sie im Feld Richtlinienname einen Namen für die Richtlinie ein (z. B. *AccessPolicyForSharedRepository*).
5. Fügen Sie die folgende Richtlinie unter Policy Document (Richtliniendokument) ein. Ersetzen Sie in den ARN durch den ARN der Richtlinie**`Resource`**, die vom Administrator in AccountA erstellt

wurde (z. B. arn:aws:iam:: ***111122223333:role/***), und wählen Sie dann Apply Policy aus. ***MyCrossAccountRepositoryContributorRole*** Weitere Informationen zu der Richtlinie, die vom Administrator von Konto A erstellt wurde, finden Sie unter Schritt 1: Erstellen Sie eine Richtlinie für den Repository-Zugriff in AccountA.

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource":
 "arn:aws:iam::111122223333:role/MyCrossAccountRepositoryContributorRole"
  }
}
```

6. Wählen Sie die Registerkarte Users. Wählen Sie Benutzer zur Gruppe hinzufügen und fügen Sie dann die AccountB IAM-Benutzer hinzu. Sie können der Gruppe beispielsweise einen IAM-Benutzer mit dem Benutzernamen *Saanvi_Sarkar* hinzufügen.

> **Note**
>
> Benutzer in AccountB benötigen programmatischen Zugriff, einschließlich eines Zugriffsschlüssels und eines geheimen Schlüssels, um ihre lokalen Computer für den Zugriff auf das gemeinsam genutzte CodeCommit Repository zu konfigurieren. Wenn Sie IAM-Benutzer erstellen, achten Sie darauf, den Zugriffsschlüssel und den geheimen Schlüssel zu speichern. Damit die Sicherheit des AWS -Kontos gewährleistet ist, kann auf den geheimen Zugriffsschlüssel nur während seiner Erstellung zugegriffen werden.

Kontoübergreifender Repository-Zugriff: Aktionen für den Repository-Benutzer in AccountB

Um auf das Repository in Konto A zugreifen zu können, müssen die Benutzer in der Gruppe von Konto B ihre lokalen Computer für den Repository-Zugriff konfigurieren. In den folgenden Abschnitten werden die Schritte dazu mit Beispielen beschrieben.

Themen

- Schritt 1: Konfiguration von AWS CLI und Git für einen AccountB-Benutzer, um auf das Repository in AccountA zuzugreifen

- Schritt 2: Klonen Sie das CodeCommit Repository in AccountA und greifen Sie darauf zu

Schritt 1: Konfiguration von AWS CLI und Git für einen AccountB-Benutzer, um auf das Repository in AccountA zuzugreifen

Sie können SSH-Schlüssel oder Git-Anmeldeinformationen nicht verwenden, um auf Repositorys in einem anderen Amazon Web Services Services-Konto zuzugreifen. AccountB-Benutzer müssen ihre Computer so konfigurieren, dass sie entweder git-remote-codecommit (empfohlen) oder den Credential Helper verwenden, um auf das gemeinsame CodeCommit Repository in AccountA zuzugreifen. Sie können jedoch weiterhin SSH-Schlüssel oder Git-Anmeldeinformationen verwenden, wenn Sie auf Repositorys in Konto B zugreifen.

Gehen Sie folgendermaßen vor, um den Zugriff mithilfe von git-remote-codecommit zu konfigurieren. Falls Sie es noch nicht installiert habengit-remote-codecommit, laden Sie es von git-remote-codecommitder Python Package Index-Website herunter.

Um Git AWS CLI und Git für den kontoübergreifenden Zugriff zu konfigurieren

1. Installieren Sie das AWS CLI auf dem lokalen Computer. Anleitungen für Ihr Betriebssystem finden Sie unter Installieren der AWS CLI.
2. Installieren Sie Git auf dem lokalen Computer. Für die Installation von Git empfehlen wir Websites wie Git Downloads oder Git for Windows.

   > **Note**
   >
   > CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git. Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter anFehlerbehebung.

3. Führen Sie am Terminal oder über die Befehlszeile an dem Speicherort, an dem Sie das Repository klonen möchten, die Befehle git config --local user.name und git config --local user.email aus, um den Benutzernamen und die E-Mail-Adresse für die Commits für das Repository anzugeben. Beispielsweise:

```
git config --local user.name "Saanvi Sarkar"
git config --local user.email saanvi_sarkar@example.com
```

Diese Befehle geben nichts zurück. Die E-Mail-Adresse und der Benutzername, die Sie angeben, werden aber mit den Commits verknüpft, die Sie für das Repository in Konto A vornehmen.

4. Führen Sie den Befehl aws configure --profile aus, um ein Standardprofil für das Herstellen einer Verbindung zu Ressourcen in Konto B zu konfigurieren. Wenn Sie dazu aufgefordert werden, geben Sie den Zugriffsschlüssel und den geheimen Schlüssel für Ihren IAM-Benutzer ein.

> **Note**
>
> Wenn Sie das Profil bereits installiert AWS CLI und konfiguriert haben, können Sie diesen Schritt überspringen.

Führen Sie beispielsweise den folgenden Befehl aus, um ein AWS CLI Standardprofil zu erstellen, das Sie für den Zugriff auf AWS Ressourcen in AccountB in US East (Ohio) (us-east-2) verwenden:

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie die folgenden Informationen an:

```
AWS Access Key ID [None]: Your-IAM-User-Access-Key
AWS Secret Access Key ID [None]: Your-IAM-User-Secret-Access-Key
Default region name ID [None]: us-east-2
Default output format [None]: json
```

5. Führen Sie den Befehl aws configure --profile erneut aus, um ein Profil zu konfigurieren, das für die Verbindung mit dem Repository in Konto A verwendet werden soll. Wenn Sie dazu aufgefordert werden, geben Sie den Zugriffsschlüssel und den geheimen Schlüssel für Ihren IAM-Benutzer ein. Führen Sie beispielsweise den folgenden Befehl aus, um ein AWS CLI Profil mit dem Namen zu erstellen *MyCrossAccountAccessProfile*, das Sie für den Zugriff auf ein Repository in AccountA in US East (Ohio) (us-east-2) verwenden:

```
aws configure --profile MyCrossAccountAccessProfile
```

Wenn Sie dazu aufgefordert werden, geben Sie die folgenden Informationen an:

```
AWS Access Key ID [None]: Your-IAM-User-Access-Key
AWS Secret Access Key ID [None]: Your-IAM-User-Secret-Access-Key
Default region name ID [None]: us-east-2
Default output format [None]: json
```

6. Öffnen Sie in einem Texteditor die Datei `config` (die AWS CLI -Konfigurationsdatei). Je nach Betriebssystem befindet sich diese Datei unter Linux, macOS oder Unix oder `~/.aws/config` unter Windows unter *Laufwerk*:\ Users\ *USERNAME*\ .aws\ config.
7. Suchen Sie in der Datei den Eintrag, der dem Standardprofil entspricht, das Sie für den Zugriff auf Repositorys in Konto B konfiguriert haben. Das sollte bei Ihnen ähnlich wie im folgenden Bild aussehen:

   ```
   [default]
   region = us-east-2
   output = json
   ```

   Fügen Sie `account` der Profilkonfiguration hinzu. Geben Sie die AWS Konto-ID von AccountB an. Beispielsweise:

   ```
   [default]
   account = 888888888888
   region = us-east-2
   output = json
   ```

8. Suchen Sie in der Datei den Eintrag, der dem *MyCrossAccountAccessProfile*Profil entspricht, das Sie gerade erstellt haben. Das sollte bei Ihnen ähnlich wie im folgenden Bild aussehen:

   ```
   [profile MyCrossAccountAccessProfile]
   region = us-east-2
   output = json
   ```

   Fügen Sie `account`, `role_arn` und `source_profile` der Profilkonfiguration hinzu. Geben Sie die Amazon Web Services Services-Konto-ID von AccountA, den ARN der Rolle in AccountA, von der Sie annehmen, dass sie auf das Repository in dem anderen Konto zugreift, und den Namen Ihres AWS CLI Standardprofils in AccountB an. Beispielsweise:

```
[profile MyCrossAccountAccessProfile]
region = us-east-2
account = 111122223333
role_arn = arn:aws:iam::111122223333:role/MyCrossAccountRepositoryContributorRole
source_profile = default
output = json
```

Speichern Sie Ihre Änderungen und schließen Sie den Texteditor.

Schritt 2: Klonen Sie das CodeCommit Repository in AccountA und greifen Sie darauf zu

Führen Sie git clonegit push, und aus, git pull um das kontenübergreifende CodeCommit Repository zu klonen, zu pushen und Daten daraus abzurufen. Sie können sich auch bei der AWS Management Console anmelden, die Rollen wechseln und die CodeCommit Konsole verwenden, um mit dem Repository des anderen Kontos zu interagieren.

> **Note**
>
> Je nachdem, wie die IAM-Rolle konfiguriert wurde, können Sie Repositorys möglicherweise auf der Standardseite für anzeigen. CodeCommit Wenn Sie die Repositorys nicht einsehen können, bitten Sie den Repository-Administrator, Ihnen per E-Mail einen URL-Link zur Codepage für das gemeinsam genutzte Repository in der Konsole zu senden. CodeCommit Die URL sollte in etwa wie folgt aussehen:
>
> ```
> https://console.aws.amazon.com/codecommit/home?region=us-east-2#/
> repository/MySharedDemoRepo/browse/HEAD/--/
> ```

So klonen Sie das kontoübergreifende Repository auf Ihrem lokalen Computer

1. Führen Sie über die Befehlszeile oder am Terminal in dem Verzeichnis, in dem Sie das Repository klonen möchten, den Befehl git clone mit der HTTPS (GRC)-Klon-URL aus. Beispielsweise:

   ```
   git clone codecommit://MyCrossAccountAccessProfile@MySharedDemoRepo
   ```

Sofern Sie nichts anderes angeben, wird das Repository in ein Unterverzeichnis mit demselben Namen wie das Repository geklont.

2. Ändern Sie die Verzeichnisse zum geklonten Repository und fügen Sie eine Datei hinzu oder nehmen Sie eine Änderung an einer Datei vor. Sie können beispielsweise eine Datei mit dem Namen *NewFile.txt* hinzufügen.
3. Fügen Sie die Datei zu den verfolgten Änderungen für das lokale Repository hinzu, übertragen Sie die Änderung und übertragen Sie die Datei per Push in das CodeCommit Repository. Beispielsweise:

```
git add NewFile.txt
git commit -m "Added a file to test cross-account access to this repository"
git push
```

Weitere Informationen finden Sie unter Erste Schritte mit Git und AWS CodeCommit.

Nachdem du eine Datei hinzugefügt hast, kannst du in der CodeCommit Konsole deinen Commit ansehen, die Änderungen anderer Benutzer am Repo überprüfen, an Pull-Requests teilnehmen und vieles mehr.

So greifen Sie in der Konsole auf das kontoübergreifende Repository zu CodeCommit

1. Melden Sie sich AWS Management Console bei AccountB (*888888888888*) als der IAM-Benutzer an, dem kontoübergreifender Zugriff auf das Repository in AccountA gewährt wurde.
2. Wählen Sie in der Navigationsleiste Ihren Benutzernamen aus. Wählen Sie dann in der Dropdown-Liste Rollen wechseln aus.

> **Note**
>
> Wenn Sie diese Option zum ersten Mal auswählen, überprüfen Sie die Informationen auf der Seite und klicken Sie dann erneut auf Rollen wechseln.

3. Führen Sie auf der Seite Rollen wechseln die folgenden Schritte aus:
   - Geben Sie unter Konto die Konto-ID für AccountA ein (z. B. *111122223333*).
   - Geben Sie im Feld Rolle den Namen der Rolle ein, die Sie für den Zugriff auf das Repository in AccountA übernehmen möchten (z. B. *MyCrossAccountRepositoryContributorRole*).

- Geben Sie unter Display Name (Anzeigename) einen Anzeigenamen für die Rolle ein. Dieser Name wird in der Konsole angezeigt, wenn Sie diese Rolle übernehmen. Er wird auch in der Liste der angenommenen Rollen angezeigt, wenn Sie in der Konsole das nächste Mal die Rolle wechseln möchten.
- (Optional) Wählen Sie unter Farbe eine Farbkennzeichnung für den Anzeigenamen aus.
- Wählen Sie Switch Role.

Weitere Informationen finden Sie unter Wechseln zu einer Rolle (AWS Management Console).

4. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.

   Wenn die angenommene Rolle berechtigt ist, die Namen von Repositorys in Konto A anzuzeigen, sehen Sie eine Liste der Repositorys sowie eine Fehlermeldung, die Sie darüber informiert, dass Sie keine Berechtigungen zum Anzeigen der zugehörigen Statuswerte haben. Dieses Verhalten wird erwartet. Wählen Sie den Namen des freigegebenen Repositorys aus der Liste aus.

   Wenn die angenommene Rolle nicht berechtigt ist, die Namen von Repositorys in Konto A anzuzeigen, sehen Sie eine Fehlermeldung und eine leere Liste ohne Repositorys. Fügen Sie den URL-Link zum Repository ein oder ändern Sie den Konsolenlink und ändern Sie `/list` in den Namen des freigegebenen Repositorys (z. B. `/MySharedDemoRepo`).

5. Suchen Sie unter Code den Namen der Datei, die Sie vom lokalen Computer hinzugefügt haben. Wählen Sie die Datei aus und durchsuchen Sie den Code in der Datei. Durchsuchen Sie dann das restliche Repository und beginnen Sie mit der Verwendung der Funktionen.

   Weitere Informationen finden Sie unter Erste Schritte mit AWS CodeCommit.

Löschen Sie ein AWS CodeCommit Repository

Sie können die CodeCommit Konsole oder die verwenden AWS CLI , um ein CodeCommit Repository zu löschen.

> **Note**
>
> Beim Löschen eines Repositorys werden lokale Kopien dieses Repositorys (lokale Repos) nicht gelöscht. Um ein lokales Repo zu löschen, verwenden Sie die Verzeichnis- und Dateiverwaltungstools Ihres lokalen Computers.

Themen

- Löschen Sie ein CodeCommit Repository (Konsole)
- Löscht ein lokales Repo
- Löscht ein CodeCommit Repository ()AWS CLI

Löschen Sie ein CodeCommit Repository (Konsole)

Gehen Sie wie folgt vor, um mit der CodeCommit Konsole ein CodeCommit Repository zu löschen.

> **Important**
>
> Nachdem Sie ein CodeCommit Repository gelöscht haben, können Sie es nicht mehr in ein lokales oder gemeinsam genutztes Repository klonen. Sie sind auch nicht mehr in der Lage, Daten aus einem lokalen oder gemeinsam genutzten Repo daraus abzurufen oder Daten dorthin zu übertragen. Diese Aktion kann nicht rückgängig gemacht werden.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, das Sie löschen möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Wählen Sie auf der Registerkarte General (Allgemein) unter Delete repository (Repository löschen) die Option Delete repository (Repository löschen) aus. Geben Sie **`delete`** ein und klicken Sie auf Delete (Löschen). Das Repository wird dauerhaft gelöscht.

> **Note**
>
> Durch das Löschen des Repositorys in CodeCommit werden keine lokalen Repos gelöscht.

Löscht ein lokales Repo

Verwenden Sie die Verzeichnis- und Dateiverwaltungstools Ihres lokalen Computers, um das Verzeichnis zu löschen, das das lokale Repository enthält.

Durch das Löschen eines lokalen Repos wird kein CodeCommit Repository gelöscht, mit dem es möglicherweise verbunden ist.

Löscht ein CodeCommit Repository ()AWS CLI

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

AWS CLI Um das zum Löschen eines CodeCommit Repositorys zu verwenden, führen Sie den delete-repository Befehl aus und geben Sie den Namen des zu löschenden CodeCommit Repositorys an (mit der `--repository-name` Option).

> **Important**
>
> Nachdem Sie ein CodeCommit Repository gelöscht haben, können Sie es nicht mehr in ein lokales oder gemeinsam genutztes Repository klonen. Sie sind auch nicht mehr in der Lage, Daten aus einem lokalen oder gemeinsam genutzten Repo daraus abzurufen oder Daten dorthin zu übertragen. Diese Aktion kann nicht rückgängig gemacht werden.

> **Tip**
>
> Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl list-repositories aus.

Beispielsweise löschen Sie das Repository `MyDemoRepo` wie folgt:

```
aws codecommit delete-repository --repository-name MyDemoRepo
```

Bei Erfolg erscheint die ID des CodeCommit Repositorys, das dauerhaft gelöscht wurde, in der Ausgabe:

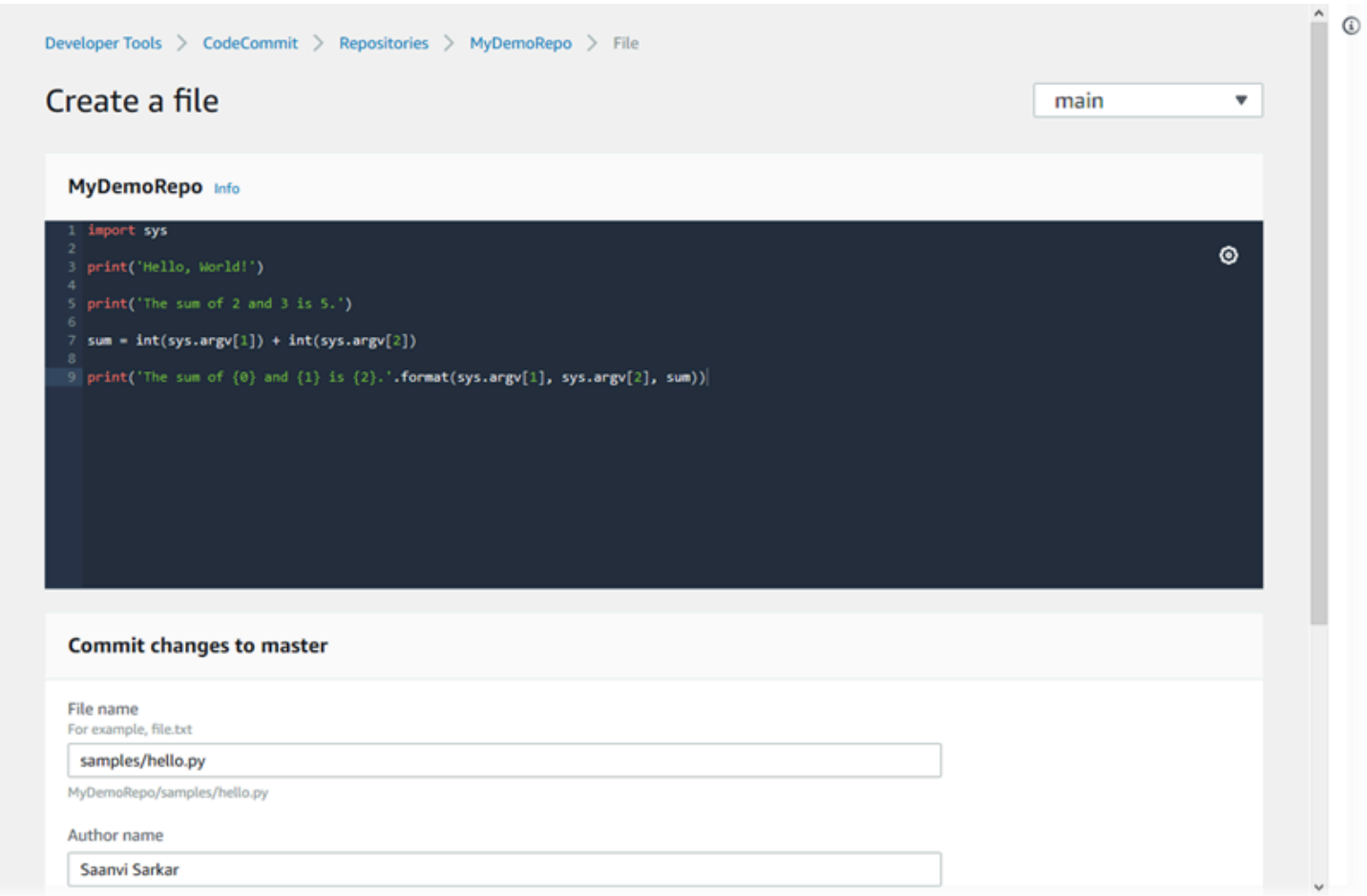
```
{
    "repositoryId": "f7579e13-b83e-4027-aaef-650c0EXAMPLE"
}
```

Durch das Löschen eines CodeCommit Repositorys werden keine lokalen Repos gelöscht, die möglicherweise damit verbunden sind.

Arbeiten mit Dateien inAWS CodeCommitRepositories

In CodeCommit ist eine Datei eine der Versionskontrolle unterliegende, unabhängige Information, die Ihnen und anderen Benutzern des Repositorys und Branchs, in dem die Datei gespeichert ist, zur Verfügung steht. Sie können Ihre Repository-Dateien mithilfe einer Verzeichnisstruktur wie auf einem Computer strukturieren. Im Gegensatz zu Ihrem Computer verfolgt CodeCommit jedoch automatisch jede Änderung einer Datei. Sie können die Versionen einer Datei vergleichen und verschiedene Versionen einer Datei in verschiedenen Branches des Repositorys speichern.

Um eine Datei in einem Repository hinzuzufügen oder zu bearbeiten, können Sie einen Git-Client verwenden. Sie können auch die CodeCommit-Konsole verwenden,AWS CLIoder die CodeCommit-API.



Weitere Informationen zur Verwendung anderer Repository's in CodeCommit finden Sie unterArbeiten mit Repositorien,Verwenden von Pull-Anforderungen,Arbeiten mit Zweigen,Mit Commits arbeiten, undArbeiten mit Benutzereinstellungenaus.

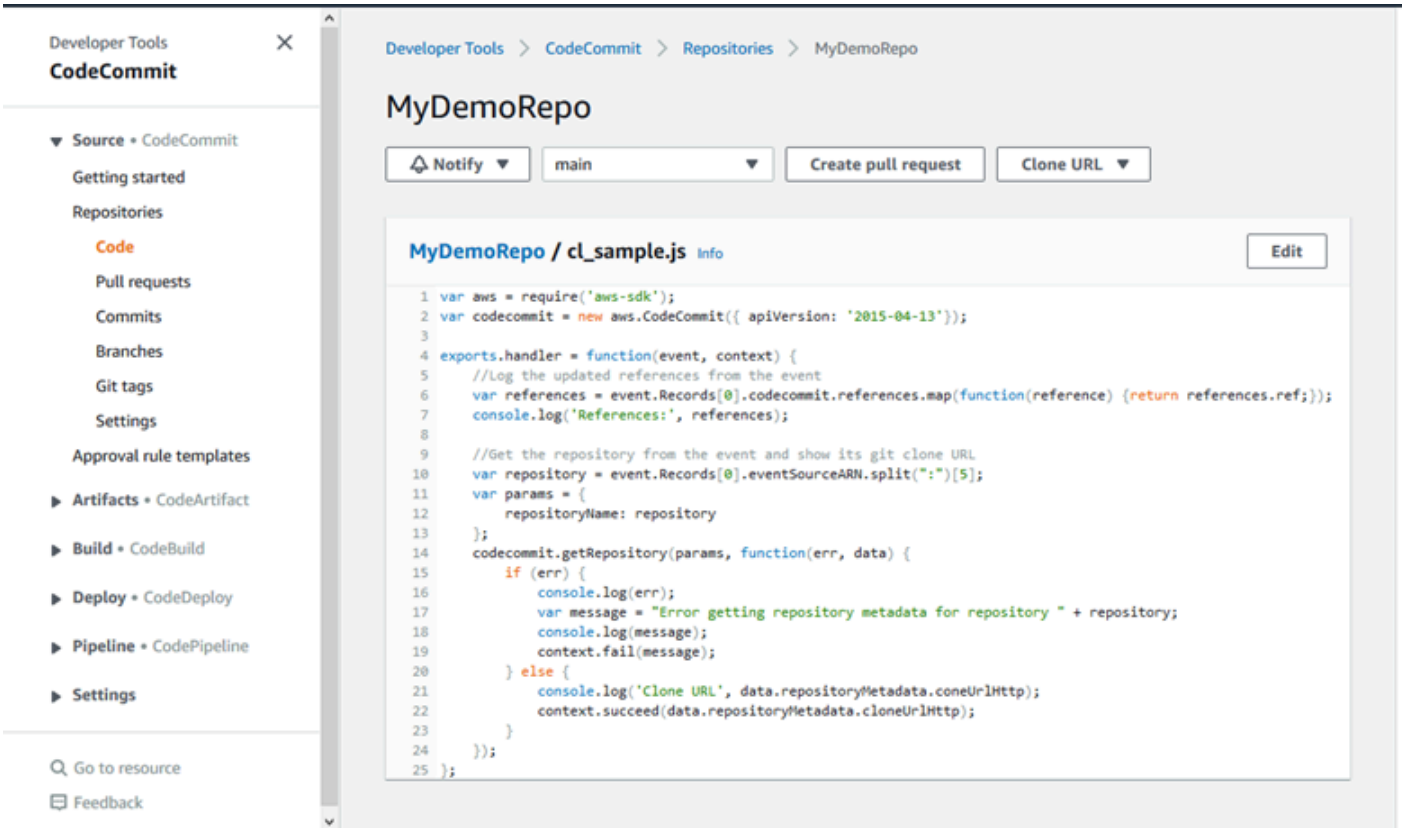
Themen

- Durchsuchen von Dateien inAWS CodeCommitEndlager
- Erstellen oder Hinzufügen einer Datei zu einemAWS CodeCommitEndlager
- Bearbeiten Sie den Inhalt einer Datei in einemAWS CodeCommit Repository

Durchsuchen von Dateien inAWS CodeCommitEndlager

Nachdem Sie eine Verbindung mit einem CodeCommit-Repository hergestellt haben, können Sie es in einem lokalen Repository klonen oder dessen Inhalte mithilfe der CodeCommit-Konsole durchsuchen. In diesem Thema wird die Verwendung vonCodeCommit-Konsole zum Durchsuchen des InhaltsCodeCommit-Repository.

> **Note**
>
> Für aktive CodeCommit-Benutzer wird keine Gebühr für das Durchsuchen von Code über die CodeCommit-Konsole erhoben. Weitere Informationen zu den Fällen, wann Gebühren anfallen können, finden Sie unter Preise.



Durchsuchen einesCodeCommitEndlager

Sie können mithilfe der CodeCommit-Konsole die Dateien in einem Repository prüfen oder schnell den Inhalt einer Datei lesen.

So durchsuchen Sie den Inhalt eines Repositorys

1. Öffnen SieCodeCommit-Konsolehttps://console.aws.amazon.com/codesuite/codecommit/homeaus.
2. Wählen Sie auf der Seite Repositories (Repositorys) in der Liste der Repositorys das Repository aus, das Sie durchsuchen möchten.
3. Durchsuchen Sie in der Codeansicht die Inhalte des Standard-Branches für Ihr Repository.

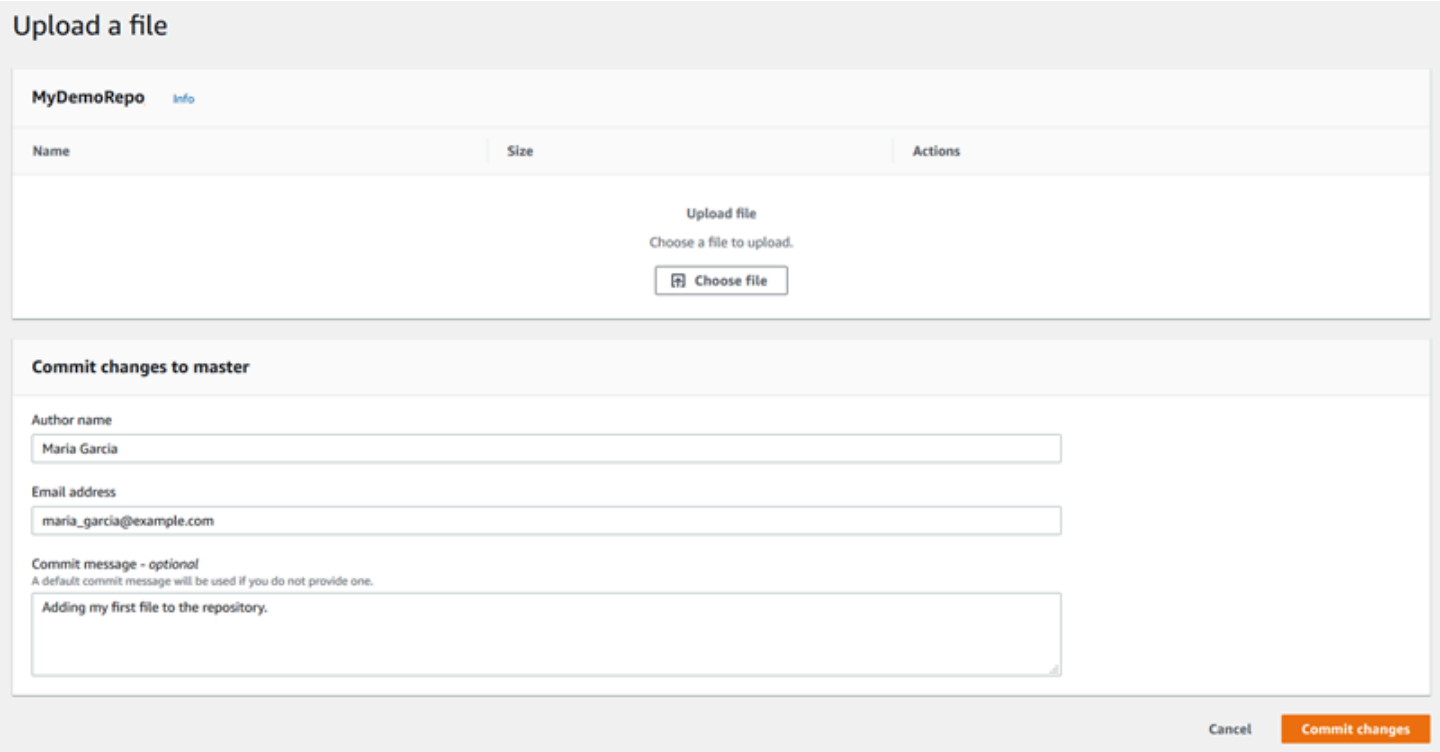
   Um einen anderen Branch oder Tag in der Ansicht anzuzeigen, klicken Sie auf die Schaltfläche der Ansichtsauswahl. Wählen Sie entweder in der Dropdown-Liste einen Branch- oder Tag-Namen aus oder geben Sie diesen im Filterfeld ein und wählen Sie ihn dann in der Liste aus.
4. Führen Sie eine der folgenden Aufgaben aus:

- Um den Inhalt eines Verzeichnisses anzuzeigen, wählen Sie es in der Liste aus. Sie können in der Navigationsliste ein beliebiges Verzeichnis auswählen, um zu dieser Verzeichnisansicht zurückzukehren. Sie können auch den Aufwärtspfeil oben in der Verzeichnisliste verwenden.
- Um den Inhalt einer Datei anzuzeigen, wählen Sie sie in der Liste aus. Wenn die Größe der Datei den Grenzwert des Commit-Objekts überschreitet, kann sie nicht in der Konsole angezeigt werden. Stattdessen muss sie in einem lokalen Repository aufgerufen werden. Weitere Informationen finden Sie unter Kontingente . Zum Beenden der Dateiansicht wählen Sie in der Code-Navigationsleiste das anzuzeigende Verzeichnis aus.

> **Note**
>
> Nicht alle Binärdateien sind in der Konsole sichtbar. Wenn Sie eine Binärdatei wählenund es ist potenziell sichtbarwird eine Warnmeldung eingeblendet, in der Sie zur Bestätigung aufgefordert werden, ob Sie den Inhalt anzeigen möchten. Um die Datei anzuzeigen, wählen Sie die Option Show file contents aus. Wenn Sie die Datei nicht anzeigen möchten, wählen Sie in der Code-Navigationsleiste das anzuzeigende Verzeichnis aus. Wenn Sie eine Markdown-Datei (.md) auswählen, klicken Sie aufRendered MarkdownundMarkdown-QuelleSchaltflächen zum Umschalten zwischen der gerenderten und der Syntaxansicht. Weitere Informationen finden Sie unterVerwenden von Markdown in der Konsoleaus.

Erstellen oder Hinzufügen einer Datei zu einemAWS CodeCommitEndlager

Sie können die CodeCommit-Konsole verwenden,AWS CLIoder einen Git-Client, um eine Datei zu einem -Repository hinzuzufügen. Sie können eine Datei vom lokalen Computer in das Repository hochladen oder den Code-Editor in der Konsole zum Erstellen der Datei verwenden. Der Editor ist eine schnelle und einfache Möglichkeit zum Hinzufügen einer einfachen Datei (z. B. einer readme.md-Datei) zu einem Branch in einem Repository.



Themen

- Erstellen oder Hochladen einer Datei (Konsole)
- Hinzufügen einer Datei (AWS CLI)
- Hinzufügen einer Datei (Git)

Erstellen oder Hochladen einer Datei (Konsole)

Sie können die CodeCommit-Konsole verwenden, um eine Datei zu erstellen und zu einem Branch in einem CodeCommit-Repository hinzuzufügen. Beim Erstellen der Datei können Sie Ihren Benutzernamen und eine E-Mail-Adresse angeben. Sie können auch eine Commit-Nachricht hinzufügen, sodass andere Benutzer wissen, wer die Datei hinzugefügt hat und aus welchem Grund. Sie können eine Datei auch direkt von Ihrem lokalen Computer in einen Branch in einem Repository hochladen.

So fügen Sie eine Datei zu einem Repository hinzu

1. Öffnen Sie die CodeCommit-Konsole unter.https://console.aws.amazon.com/codesuite/codecommit/homeaus.

2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie eine Datei hinzufügen möchten.
3. Wählen Sie in der Codeansicht den Branch aus, in dem Sie die Datei hinzufügen möchten. Standardmäßig werden beim Öffnen der Codeansicht die Inhalte der Standardverzweigung angezeigt.

   Um einen anderen Branch anzuzeigen, wählen Sie die Schaltfläche der Ansichtsauswahl. Wählen Sie entweder in der Dropdown-Liste einen Branch-Namen aus oder geben Sie diesen im Filterfeld ein und wählen Sie ihn dann aus der Liste aus.
4. Wählen Sie Add file (Datei hinzufügen) und dann eine der folgenden Optionen:

   - Um den Code-Editor zum Erstellen des Inhalts einer Datei und zum Hinzufügen der Datei zum Repository zu verwenden, wählen Sie Create file (Datei erstellen).
   - Um eine Datei vom lokalen Computer in das Repository hochzuladen, wählen Sie Upload file (Datei hochladen).
5. Stellen Sie den anderen Benutzern Informationen darüber bereit, wer diese Datei zum Repository hinzugefügt hat und warum.

   - Geben Sie im Feld Author name (Name des Autors) Ihren Namen ein. Dieser Name wird als Name des Autors und Committers in den Commit-Informationen verwendet. Standardmäßig wird von CodeCommit Ihr IAM-Benutzername oder eine Ableitung Ihrer Konsolenanmeldung als Name des Autors verwendet.
   - In :E-Mail-AdresseGeben Sie eine E-Mail-Adresse ein, sodass sich andere Repository-Benutzer bei Fragen zu dieser Änderung an Sie wenden können.
   - Geben Sie in das Feld Commit message (Commit-Nachricht) eine kurze Beschreibung ein. Dies ist zwar optional, wird aber ausdrücklich empfohlen. Andernfalls wird eine Standard-Commit-Nachricht verwendet.
6. Führen Sie eine der folgenden Aufgaben aus:

   - Wenn Sie eine Datei hochladen, wählen Sie sie auf dem lokalen Computer aus.
   - Wenn Sie eine Datei erstellen, geben Sie den gewünschten Inhalt im Code-Editor ein und geben Sie einen Namen für die Datei an.
7. Wählen Sie Commit changes (Änderungen übernehmen) aus.

Hinzufügen einer Datei (AWS CLI)

Sie können dasAWS CLIund dieput-fileBefehl, um eine Datei in einem CodeCommit-Repository hinzuzufügen. Sie können den Befehl put-file auch nutzen, um ein Verzeichnis oder eine Pfadstruktur für die Datei hinzuzufügen.

> **Note**
>
> Um zu verwendenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLIaus. Weitere Informationen finden Sie unter Befehlszeilenreferenz .

So fügen Sie eine Datei zu einem Repository hinzu

1. Erstellen Sie auf dem lokalen Computer die Datei, die Sie dem CodeCommit-Repository hinzufügen möchten.
2. Führen Sie am Terminal oder in der Befehlszeile den Befehl put-file unter Angabe der folgenden Informationen aus:

   - Repository, dem Sie die Datei hinzufügen möchten
   - Branch, dem Sie die Datei hinzufügen möchten
   - Vollständige Commit-ID des letzten Commits an diesem Branch (auch bezeichnet als oberster oder HEAD-Commit)
   - Der lokale Speicherort der Datei Die Syntax für diesen Standort hängt von Ihrem lokalen Betriebssystem ab.
   - Der Name der Datei, die Sie hinzufügen möchten, einschließlich des Pfads, in sich dem die aktualisierte Datei im Repository befindet (falls vorhanden)
   - Der Benutzername und die E-Mail-Adresse, den/die Sie mit dieser Datei verknüpfen möchten
   - Eine Commit-Nachricht, die erklärt, warum Sie diese Datei hinzugefügt haben

   Der Benutzername, die E-Mail-Adresse und die Commit-Nachricht sind optional. Teilen Sie aber anderen Benutzern mit, wer die Änderung warum vorgenommen hat. Wenn Sie keinen Benutzernamen angeben, wird von CodeCommit standardmäßig Ihr IAM-Benutzername oder eine Ableitung Ihrer Konsolenanmeldung als Name des Autors verwendet.

Beispiel: Sie möchten eine Datei namens *ExampleSolution.py* dem Repository *MyDemoRepo* für einen Branch mit der Bezeichnung *feature-randomizationfeature* hinzufügen, dessen letzter Commit die ID *4c925148EXAMPLE* hat:

```
aws codecommit put-file --repository-name MyDemoRepo --branch-name feature-
randomizationfeature --file-content file://MyDirectory/ExampleSolution.py --file-
path /solutions/ExampleSolution.py --parent-commit-id 4c925148EXAMPLE --name "María
 García" --email "maría_garcía@example.com" --commit-message "I added a third
 randomization routine."
```

> **Note**
>
> Wenn Sie Binärdateien hinzufügen, stellen Sie sicher, dass Sie für die Angabe des lokalen Speicherorts der Datei `fileb://` verwenden.

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

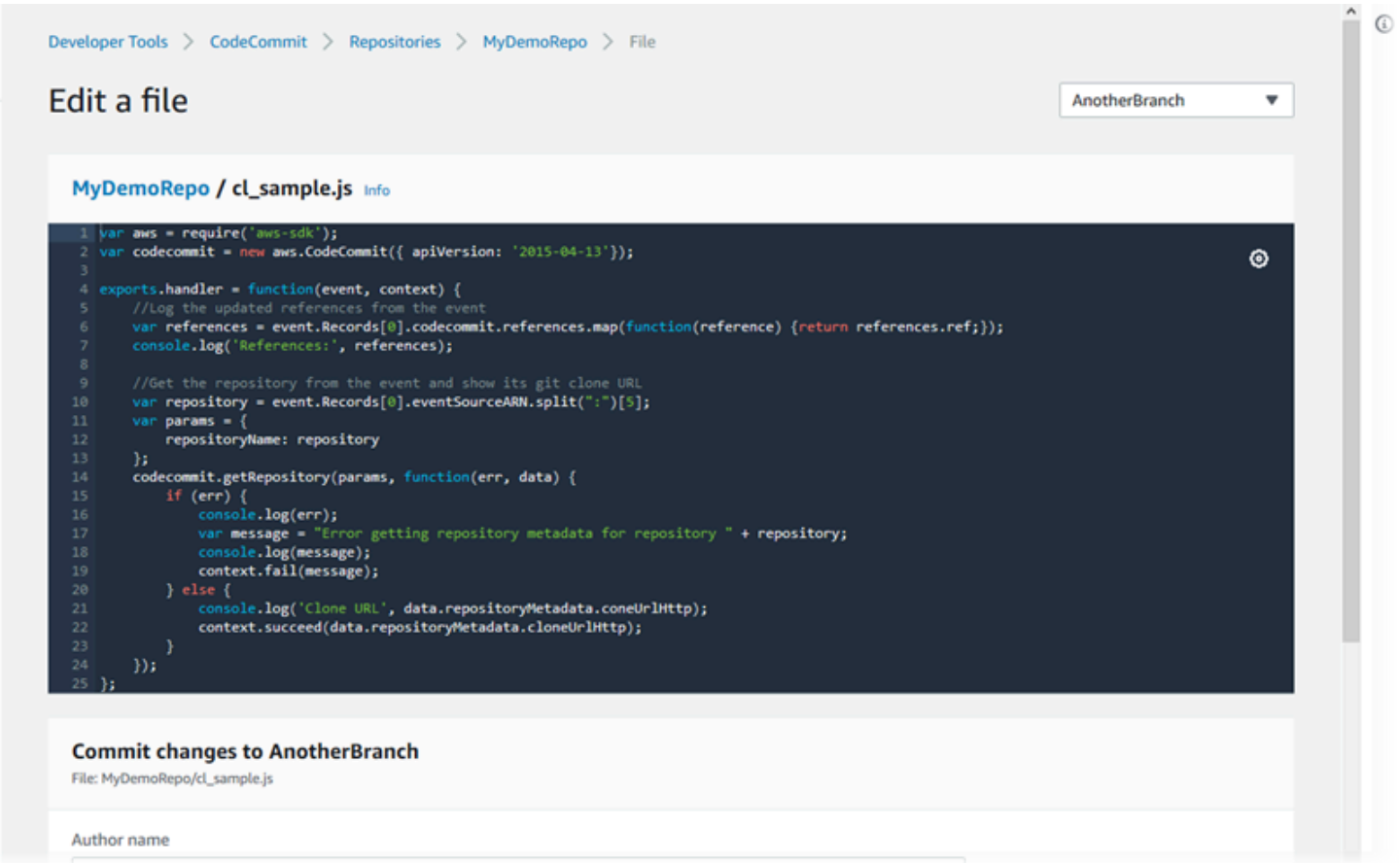
```
{
   "blobId": "2eb4af3bEXAMPLE",
   "commitId": "317f8570EXAMPLE",
   "treeId": "347a3408EXAMPLE"
}
```

Hinzufügen einer Datei (Git)

Sie können Dateien in einem lokalen Repository hinzufügen und Ihre Änderungen per Push an ein CodeCommit-Repository übertragen. Weitere Informationen finden Sie unter Erste Schritte mit Git und AWS CodeCommit.

Bearbeiten Sie den Inhalt einer Datei in einemAWS CodeCommit Repository

Du kannst die CodeCommit Konsole oder einen Git-Client verwendenAWS CLI, um den Inhalt einer Datei in einem CodeCommit Repository zu bearbeiten.



Themen

- Bearbeiten von eingebundenen Dateien (Konsole)
- Eine Datei bearbeiten oder löschen (AWS CLI)
- Eine Datei bearbeiten (Git)

Bearbeiten von eingebundenen Dateien (Konsole)

Sie können die CodeCommit Konsole verwenden, um eine Datei zu bearbeiten, die zu einem Zweig in einem CodeCommit Repository hinzugefügt wurde. Beim Bearbeiten der Datei können Sie den Benutzernamen und eine E-Mail-Adresse angeben. Sie können auch eine Commit-Nachricht hinzufügen, sodass andere Benutzer wissen, wer die Datei geändert hat und aus welchem Grund.

So bearbeiten Sie eine Datei in einem Repository

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.

2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie eine Datei bearbeiten möchten.
3. Wählen Sie in der Codeansicht den Branch aus, in dem Sie die Datei bearbeiten möchten. Standardmäßig werden beim Öffnen der Codeansicht die Inhalte der Standardverzweigung angezeigt.

   Um einen anderen Branch anzuzeigen, wählen Sie die Schaltfläche der Ansichtsauswahl. Wählen Sie entweder in der Dropdown-Liste einen Branch-Namen aus oder geben Sie diesen im Filterfeld ein und wählen Sie ihn dann aus der Liste aus.
4. Navigieren Sie durch den Inhalt des Branch und wählen Sie die Datei aus, die Sie bearbeiten möchten. Wählen Sie in der Dateiansicht die Option Edit (Bearbeiten).

   > **Note**
   >
   > Wenn Sie eine Binärdatei auswählen, wird eine Warnmeldung eingeblendet, in der Sie zur Bestätigung aufgefordert werden, ob Sie den Inhalt anzeigen möchten. Sie sollten die CodeCommit Konsole nicht zum Bearbeiten von Binärdateien verwenden.
5. Bearbeiten Sie die Datei. Stellen Sie den anderen Benutzern Informationen darüber bereit, wer diese Änderung getätigt hat und warum.

   - Geben Sie im Feld Author name (Name des Autors) Ihren Namen ein. Dieser Name wird in den Commit-Informationen sowohl als Autorenname als auch als Committername verwendet. CodeCommit verwendet standardmäßig Ihren IAM-Benutzernamen oder eine Ableitung Ihres Konsolen-Logins als Autorennamen.
   - Geben Sie im Feld E-Mail-Adresse eine E-Mail-Adresse ein, damit andere Repository-Benutzer Sie bezüglich dieser Änderung kontaktieren können.
   - Geben Sie in das Feld Commit message (Commit-Nachricht) eine kurze Beschreibung Ihrer Änderungen ein.
6. Wählen Sie Commit changes (Änderungen übernehmen), um Ihre Änderungen an der Datei zu speichern und diese in das Repository zu übernehmen.

Eine Datei bearbeiten oder löschen (AWS CLI)

Sie können den BefehlAWS CLI und denput-file Befehl verwenden, um Änderungen an einer Datei in einem CodeCommit Repository vorzunehmen. Sie können auch den Befehl put-file verwenden,

um ein Verzeichnis oder eine Pfadstruktur für die geänderte Datei hinzuzufügen, wenn Sie diese an einem anderen Speicherort als dem ursprünglichen Ort speichern möchten. Wenn Sie eine Datei vollständig löschen möchten, können Sie den Befehl delete-file verwenden.

> **Note**
>
> UmAWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie dieAWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

So bearbeiten Sie eine Datei in einem Repository

1. Nehmen Sie unter Verwendung einer lokalen Kopie der Datei die Änderungen vor, die Sie dem CodeCommit-Repository hinzufügen möchten.
2. Führen Sie am Terminal oder in der Befehlszeile den Befehl put-file unter Angabe der folgenden Informationen aus:

   - Das Repository, dem Sie die bearbeitete Datei hinzufügen möchten
   - Der Branch, dem Sie die bearbeitete Datei hinzufügen möchten
   - Vollständige Commit-ID des letzten Commits an diesem Branch (auch bezeichnet als oberster oder HEAD-Commit)
   - Der lokale Speicherort der Datei
   - Der Name der aktualisierten Datei, die Sie hinzufügen möchten, einschließlich des Pfads, in dem sich die aktualisierte Datei im Repository befindet (falls vorhanden)
   - Der Benutzername und die E-Mail-Adresse, den/die Sie mit dieser Dateiänderung verknüpfen möchten
   - Eine Commit-Nachricht, die die Änderung erläutert

   Der Benutzername, die E-Mail-Adresse und die Commit-Nachricht sind optional. Teilen Sie aber anderen Benutzern mit, wer die Änderung warum vorgenommen hat. Wenn Sie keinen Benutzernamen angeben, CodeCommit wird standardmäßig Ihr IAM-Benutzername oder eine Ableitung Ihres Konsolen-Logins verwendet.

   Um beispielsweise Änderungen an einer Datei *`ExampleSolutionnamens.py`* zu einem Repository hinzuzufügen, das einem Zweig namens *`feature-randomizationfeature`* zugeordnet ist, dessen letzter Commit die ID *`4c925148Example`* hat: *`MyDemoRepo`*

```
aws codecommit put-file --repository-name MyDemoRepo --branch-name feature-
randomizationfeature --file-content file://MyDirectory/ExampleSolution.py --file-
path /solutions/ExampleSolution.py --parent-commit-id 4c925148EXAMPLE --name "María
 García" --email "maría_garcía@example.com" --commit-message "I fixed the bug Mary
 found."
```

> **Note**
>
> Wenn Sie eine geänderte Binärdatei hinzufügen möchten, sollten Sie `--file-content` mit der Notation **`fileb://`***`MyDirectory`*`/`*`MyFile`*`.raw` verwenden.

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
   "blobId": "2eb4af3bEXAMPLE",
   "commitId": "317f8570EXAMPLE",
   "treeId": "347a3408EXAMPLE"
}
```

Verwenden Sie zum Löschen einer Datei den Befehl delete-file. Um beispielsweise eine Datei mit dem Namen *README.md* in einem Zweig namens *main* mit der neuesten Commit-ID *c5709475Example* in einem Repository mit dem Namen zu löschen *MyDemoRepo*:

```
aws codecommit delete-file --repository-name MyDemoRepo --branch-name main --file-
path README.md --parent-commit-id c5709475EXAMPLE
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
  "blobId":"559b44fEXAMPLE",
  "commitId":"353cf655EXAMPLE",
  "filePath":"README.md",
  "treeId":"6bc824cEXAMPLE"
}
```

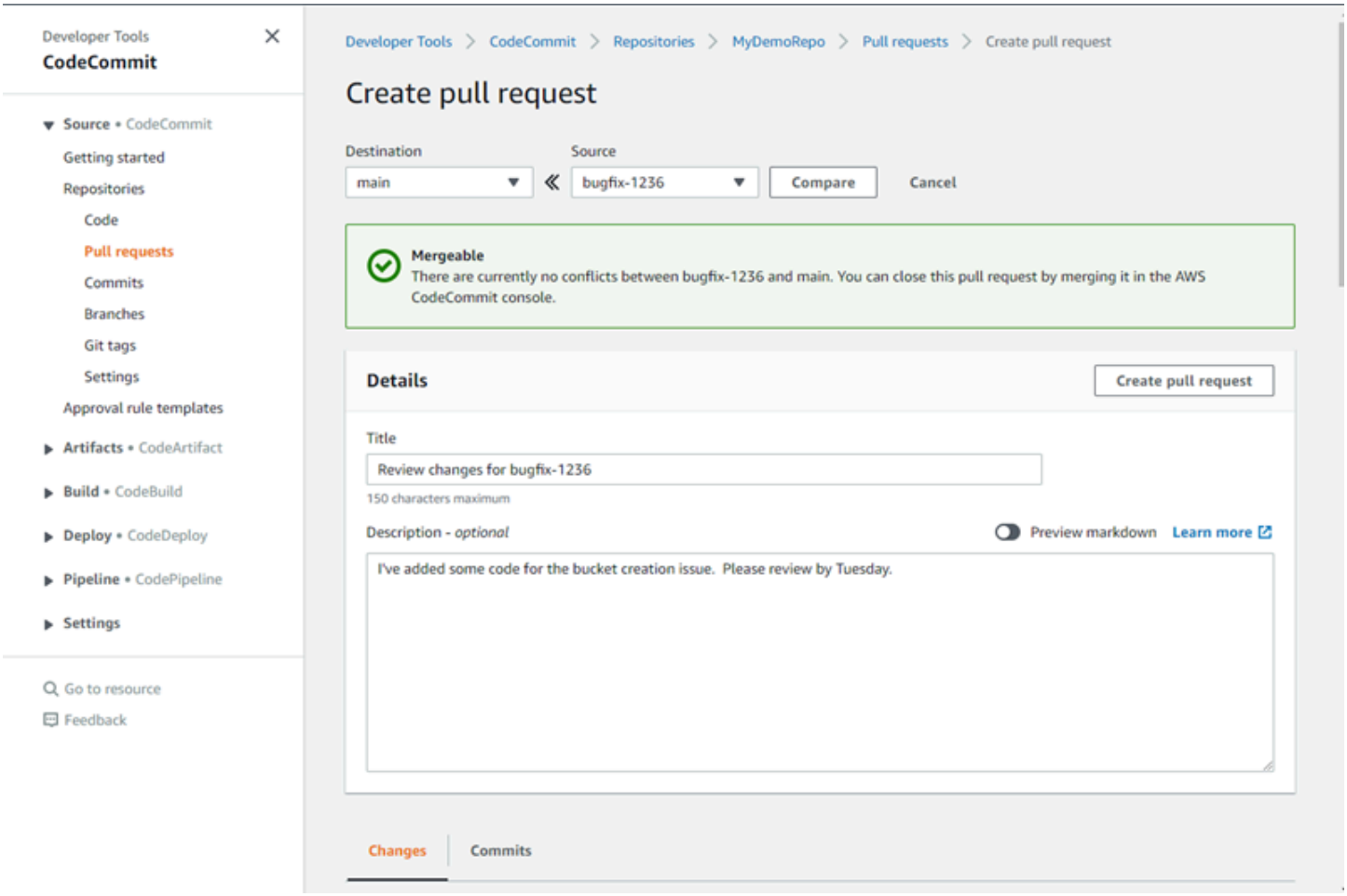
Eine Datei bearbeiten (Git)

Sie können Dateien in einem lokalen Repo bearbeiten und Ihre Änderungen in ein CodeCommit Repository übertragen. Weitere Informationen finden Sie unter Erste Schritte mit Git und AWS CodeCommit.

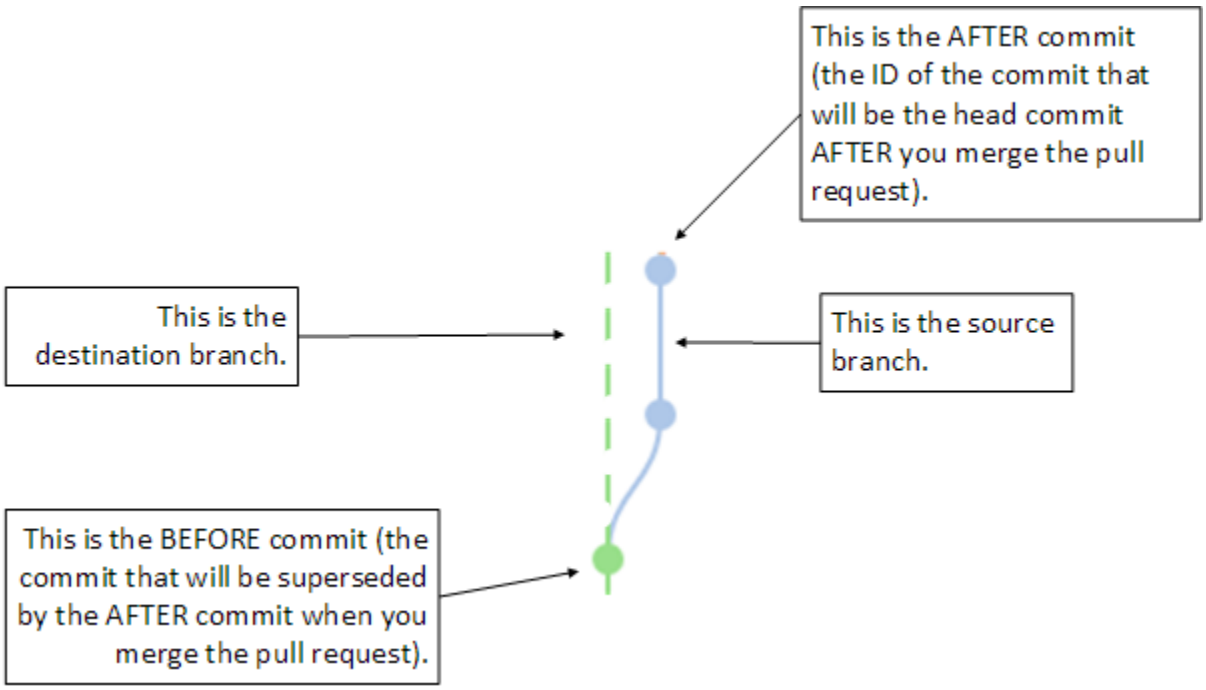
Verwenden von Pull-Anforderungen inAWS CodeCommitRepositories

Eine Pull-Anforderung ist die primäre Vorgehensweise, wie Sie und andere Repository-Benutzer Codeänderungen prüfen, kommentieren und über Branches hinweg zusammenführen können. Mit Pull-Anforderungen können Sie Codeänderungen auf kleine Änderungen oder Fehlerkorrekturen, größere Funktionserweiterungen oder neue Versionen Ihrer veröffentlichten Software gemeinsam prüfen. Hier finden Sie einen möglichen Workflow für eine Pull-Anforderung:

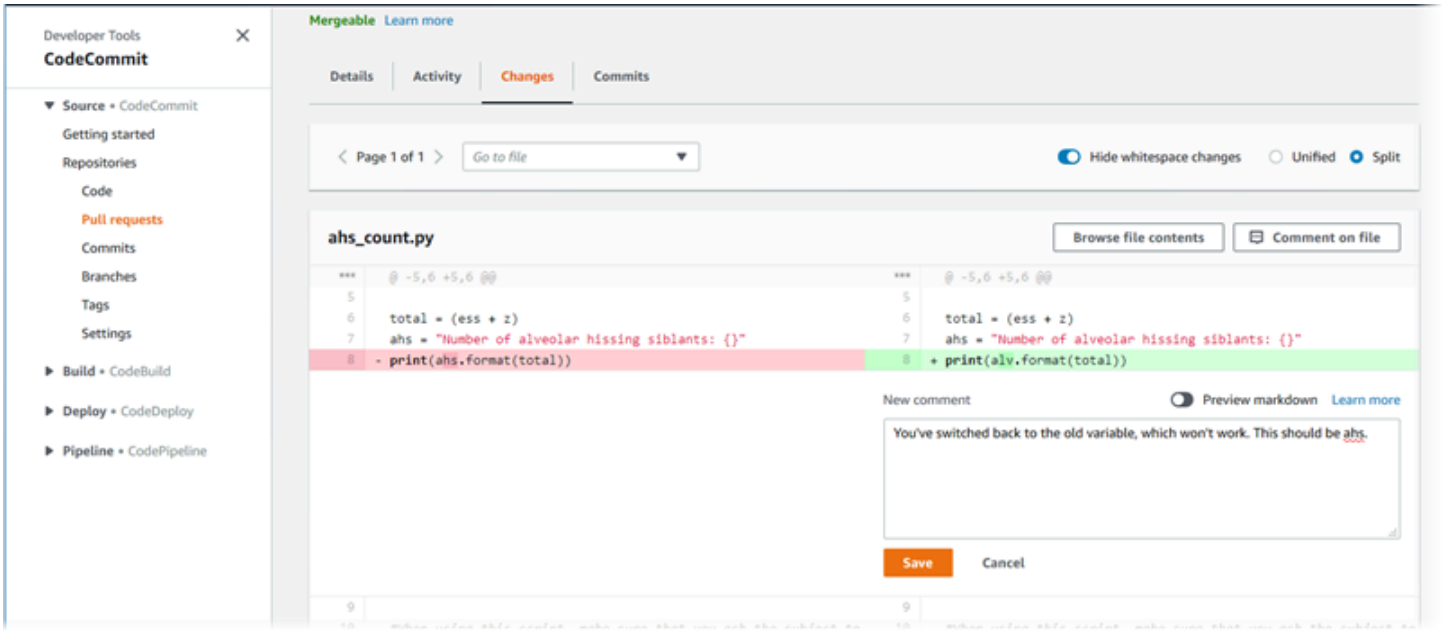
Li Juan, eine Entwicklerin, die ein Repository mit dem Namen MyDemoRepo verwendet, möchte an einer neuen Funktion für eine künftige Version eines Produkts arbeiten. Um ihre Arbeit vom produktionsbereiten Code zu trennen, erstellt sie für diese Arbeit einen Branch, der vom Standard-Branch abzweigt, und nennt diesen *`feature-randomizationfeature`*. Sie schreibt Code, führt Commits durch und überträgt den neuen Funktionscode auf diesen Branch. Sie möchte die Codequalität durch andere Repository-Benutzer überprüfen lassen, bevor sie ihre Änderungen in den Standard-Branch einfügt. Zu diesem Zweck erstellt sie eine Pull-Anforderung. Die Pull-Anforderung enthält den Vergleich zwischen dem Branch mit ihrer Arbeit und dem Branch mit dem Code, mit dem sie ihre Änderungen zusammenführen möchte (in diesem Fall der Standard-Branch). Sie kann außerdem eine Genehmigungsregel erstellen, die voraussetzt, dass die Pull-Anforderung von einer bestimmten Anzahl von Benutzern genehmigt wird. Und sie kann einen Genehmigungs-Pool von Benutzern angeben. Andere Benutzer prüfen ihren Code sowie die Änderungen und fügen Kommentare und Vorschläge hinzu. Als Reaktion auf Kommentare aktualisiert sie den Branch mit ihrer Arbeit möglicherweise mehrmals mit Codeänderungen. Ihre Änderungen werden jedes Mal in die Pull-Anforderung einbezogen, wenn sie diese auf den Branch in CodeCommit per Push überträgt. Sie kann zudem Änderungen einbeziehen, die im vorgesehenen Ziel-Branch vorgenommen wurden, während die Pull-Anforderung offen ist, sodass Benutzer sicher sein können, dass sie alle Änderungsvorschläge im Kontext sehen. Wenn sie und die Prüfer zufrieden sind und die Bedingungen für Genehmigungsregeln (falls vorhanden) erfüllt sind, führt sie oder einer der Prüfer den Code zusammen und schließt die Pull-Anforderung.



Pull-Anforderungen erfordern zwei Branches: einen Quell-Branch mit dem Code, den Sie prüfen lassen möchten, und einen Ziel-Branch, mit dem der geprüfte Code zusammengeführt werden soll. Der Quell-Branch enthält den AFTER-Commit, bei dem es sich um den Commit mit den Änderungen handelt, die Sie mit dem Ziel-Branch zusammenführen möchten. Der Ziel-Branch enthält den BEFORE-Commit, der den Status des Codes repräsentiert, bevor der Pull-Anforderungs-Branch mit dem Ziel-Branch zusammengeführt wird. Die Wahl der Mergestrategie wirkt sich auf die Details aus, wie Commits zwischen den Quell- und Ziel-Branches in der CodeCommit-Konsole zusammengeführt werden. Weitere Informationen zu Mergestrategien in CodeCommit finden Sie unterZusammenführen einer Pull-Anforderung (-Konsole)aus.



Die Pull-Anforderung zeigt die Unterschiede zwischen der Spitze des Source-Branch und dem neuesten Commit auf dem Ziel-Branch, auf dem die Pull-Anforderung erstellt wird, sodass die Benutzer Änderungen ansehen und kommentieren können. Sie können die Pull-Anforderung aufgrund der Kommentare mit weiteren Änderungen aktualisieren, indem Sie Commits für Änderungen durchführen und diese per Push auf den Quell-Branch übertragen.



Wenn Ihr Code geprüft wurde und die Genehmigungsregeln (falls vorhanden) erfüllt sind, können Sie die Pull-Anforderung auf verschiedene Arten schließen:

- Führen Sie die Branches lokal zusammen und übertragen Sie Ihre Änderungen per Push. Damit wird die Anforderung automatisch geschlossenWenn die Mergestrategie mit Vorlauf verwendet wird und keine Mergekonflikte auftretenaus.
- Verwenden Sie die AWS CodeCommit-Konsole, um die Pull-Anforderung zu schließen, ohne sie zusammenzuführen, Konflikte in einer Zusammenführung zu lösen oder die Branches mit einer der verfügbaren Merge-Strategien zu schließen und zusammenzuführen, wenn keine Konflikte vorliegen.
- Verwenden Sie die AWS CLI.

Bevor Sie eine Pull-Anforderung erstellen, müssen Sie folgende Schritte ausführen:

- Stellen Sie sicher, dass Sie die zu prüfenden Codeänderungen bestätigt und per Push an einen Branch (den Quell-Branch) übergeben haben.
- Richten Sie Benachrichtigungen für Ihr Repository ein, sodass die Benutzer über die Pull-Anforderung und entsprechende Änderungen informiert werden. (Dieser Schritt ist zwar optional, wird aber empfohlen.)
- Erstellen Sie Genehmigungsregelvorlagen mit dem Repository und ordnen Sie sie zu, damit automatisch Genehmigungsregeln für Pull-Anforderungen erstellt werden und die Code-Qualität sichergestellt bleibt. Weitere Informationen finden Sie unter Arbeiten mit Genehmigungsregelvorlagen .

Pull-Anforderungen sind effektiver, wenn Sie IAM-Benutzer für Ihre Repository-Benutzer in Ihrem Amazon Web Services Services-Konto eingerichtet haben. Das erleichtert das Identifizieren des Benutzers, der einen Kommentar abgegeben hat. Der andere Vorteil besteht darin, dass IAM-Benutzer Git-Anmeldeinformationen für den Repository-Zugriff verwenden können. Weitere Informationen finden Sie unter Schritt 1: Erstkonfiguration für CodeCommit . Sie können Pull-Anforderungen mit anderen Arten von Benutzern verwenden, einschließlich Benutzern mit verbundenem Zugriff.

Weitere Informationen zur Verwendung anderer Repository-Funktionen in CodeCommit finden Sie unterArbeiten mit Repositorien,Arbeiten mit Genehmigungsregelvorlagen,Arbeiten mit Dateien,Mit Commits arbeiten,Arbeiten mit Zweigen, undArbeiten mit Benutzereinstellungenaus.

Themen

- Erstellen einer Pull-Anforderung
- Erstellen einer Genehmigungsregel für eine Pull-Anforderung

- Anzeigen von Pull-Anforderungen in einemAWS CodeCommitEndlager
- Überprüfen einer Pull-Anforderung
- Aktualisieren einer Pull-Anforderung
- Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung
- Überschreiben von Genehmigungsregeln für eine Pull-Anforderung
- Zusammenführen einer Pull-Anforderung in einemAWS CodeCommitEndlager
- Lösen von Konflikten in einer Pull-Anforderung in einemAWS CodeCommitEndlager
- Schließen einer Pull-Anforderung in einemAWS CodeCommitEndlager

Erstellen einer Pull-Anforderung

Durch das Erstellen von Pull-Anforderungen können andere Benutzer Ihre Codeänderungen sehen und prüfen, bevor Sie diese mit einem anderen Branch zusammenführen. Zunächst erstellen Sie einen Branch für Ihre Codeänderungen. Dieser wird als Quell-Branch einer Pull-Anforderung bezeichnet. Nachdem Sie einen Commit für diese Änderungen durchgeführt und sie per Push auf das Repository übertragen haben, können Sie eine Pull-Anforderung erstellen, die den Inhalt dieses Branch (Quell-Branch) mit dem Branch vergleicht, mit dem Sie Ihre Änderungen nach dem Schließen der Pull-Anforderung zusammenführen möchten (Ziel-Branch).

Sie können über die AWS CodeCommit-Konsole oder die AWS CLI Pull-Anforderungen für Ihr Repository erstellen.

Themen

- Erstellen einer Pull-Anforderung (-Konsole)
- Erstellen Sie eine Pull-Anforderung (AWS CLI)

Erstellen einer Pull-Anforderung (-Konsole)

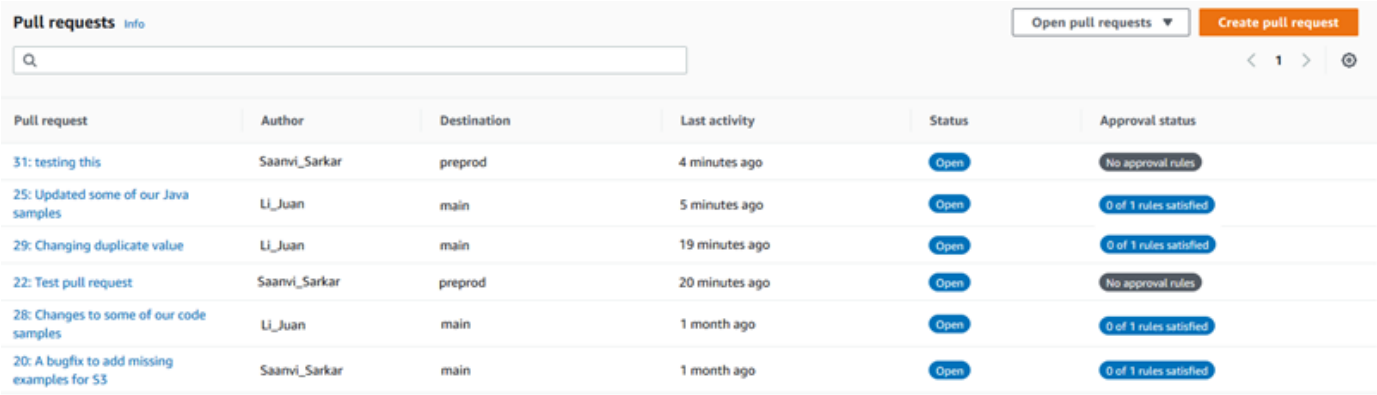
Sie können die CodeCommit-Konsole verwenden, um eine Pull-Anforderung in einem CodeCommit-Repository zu erstellen. Wenn Ihr Repository mit Benachrichtigungen konfiguriert ist, erhalten Benutzer mit Abonnement eine E-Mail, wenn Sie eine Pull-Anforderung erstellen.

1. Öffnen Sie die CodeCommit-Konsole unter.https://console.aws.amazon.com/codesuite/codecommit/homeaus.

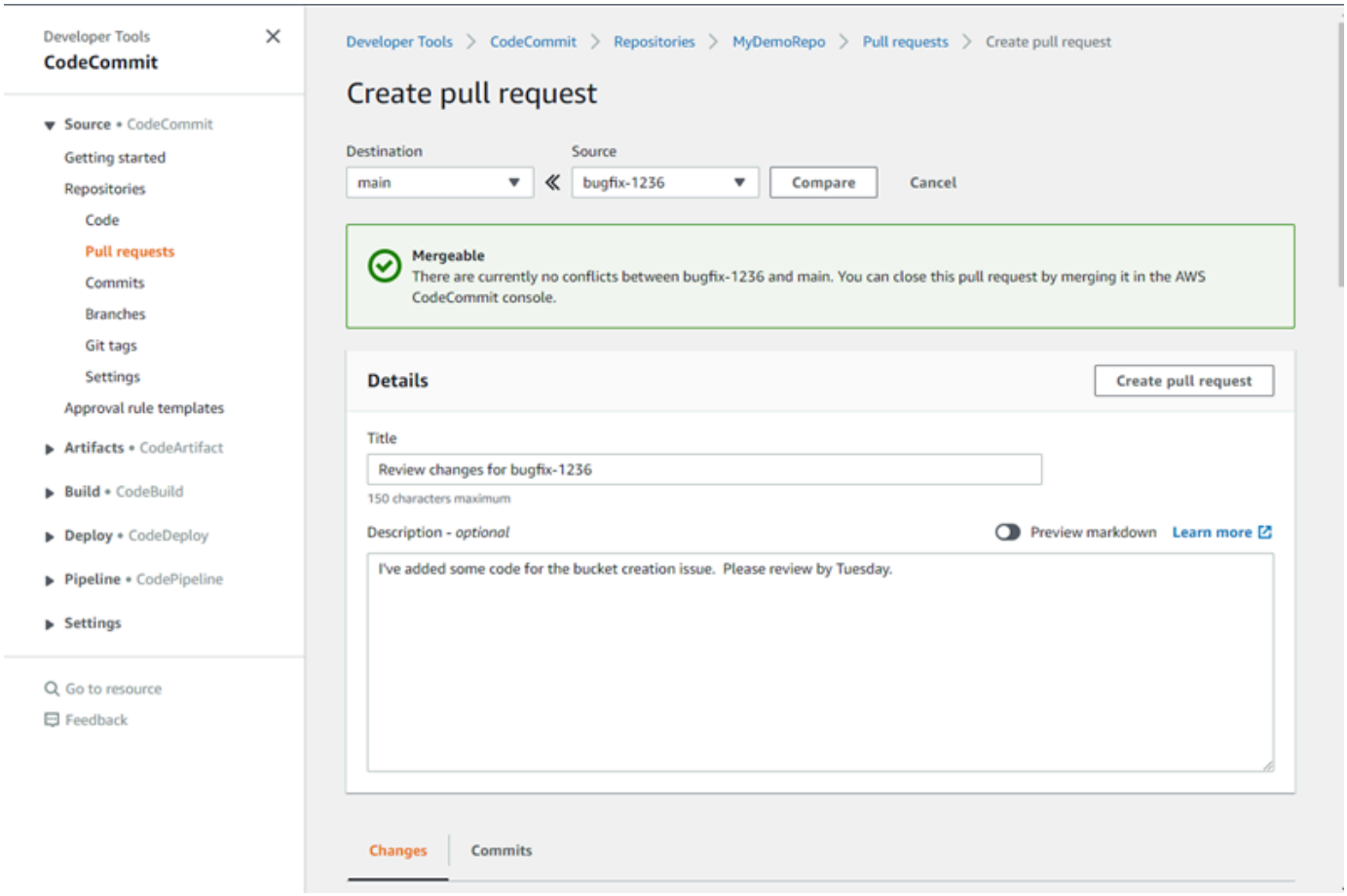
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie eine Pull-Anforderung erstellen möchten.
3. Wählen Sie im Navigationsbereich Pull Requests aus.

> **Tip**
>
> Darüber hinaus können Sie Pull-Anforderungen über Branches und Code erstellen.

4. Wählen Sie Create pull request aus.



5. Wählen Sie in Create pull request unter Source den Branch aus, der die zu prüfenden Änderungen enthält.
6. Wählen Sie unter Destination (Ziel) den Branch aus, mit dem Sie Ihre Codeänderungen nach dem Schließen der Pull-Anforderung zusammenführen möchten.
7. Wählen Sie Compare aus. Die beiden Branches werden verglichen und die Unterschiede zwischen ihnen angezeigt. Zudem wird eine Analyse durchgeführt, um zu ermitteln, ob die beiden Branches nach dem Schließen der Pull-Anforderung automatisch zusammengeführt werden können.
8. Prüfen Sie die Details des Vergleichs und die Änderungen, um sicherzustellen, dass die Pull-Anforderung die Änderungen und Commits enthält, die geprüft werden sollen. Andernfalls passen Sie Ihre Auswahl für Quell- und Ziel-Branch an und wählen erneut Compare aus.
9. Wenn Sie mit den Vergleichsergebnissen für die Pull-Anforderung zufrieden sind, geben Sie unter Title (Titel) einen kurzen, aussagekräftigen Namen für diese Prüfung an. Dieser Titel wird in der Liste der Pull-Anforderungen für das Repository angezeigt.
10. (Optional) Unter Description (Beschreibung) können Sie angeben, wofür diese Prüfung vorgesehen ist, sowie weitere nützliche Informationen für Prüfer hinterlassen.
11. Wählen Sie Create (Erstellen) aus.



Ihre Pull-Anforderung wird in der Liste der Pull-Anforderungen für das Repository angezeigt. Wenn SieKonfigurierte Benachrichtigungenerhalten Abonnenten des Amazon SNS -Themas eine E-Mail bezüglich der neu erstellten Pull-Anforderung.

Erstellen Sie eine Pull-Anforderung (AWS CLI)

Um zu verwendenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLIaus. Weitere Informationen finden Sie unter Befehlszeilenreferenz .

So verwenden Sie denAWS CLIErstellen Sie eine Pull-Anforderung in einem CodeCommit-Repository

1. Führen Sie den Befehl create-pull-request aus und geben Sie Folgendes an:

   - Name der Pull-Anforderung (mit der Option --title).
   - Beschreibung der Pull-Anforderung (mit der Option --description).
   - Liste der Ziele für den Befehl create-pull-request, einschließlich:

  - Name des CodeCommit-Repositorys, in dem die Pull-Anforderung erstellt wird (mit demrepositoryNameAttribut).
  - Name des Branches, der die zu prüfenden Codeänderungen enthält, auch Quell-Branch genannt (mit dem Attribut sourceReference).
  - (Optional) Name des Branches, mit dem Sie Ihre Codeänderungen zusammenführen möchten (auch Ziel-Branch genannt), falls Sie sie nicht mit dem Standard-Branch zusammenführen möchten (mit dem Attribut destinationReference).
- Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option --client-request-token).

In diesem Beispiel wird eine Pull-Anforderung mit dem Namen *Pronunciation difficulty analyzer* und der Beschreibung *Please review these changes by Tuesday* erstellt, die sich auf den Quell-Branch *jane-branch* bezieht. Die Pull-Anforderung soll in den Standardzweig zusammengeführt werden*Haupt*in einem CodeCommit-Repository mit dem NamenMyDemoRepo:

```
aws codecommit create-pull-request --title "Pronunciation difficulty analyzer"
 --description "Please review these changes by Tuesday" --client-request-token
 123Example --targets repositoryName=MyDemoRepo,sourceReference=jane-branch
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "originApprovalRuleTemplate": {
                    "approvalRuleTemplateId": "dd3d22fe-EXAMPLE",
                    "approvalRuleTemplateName": "2-approver-rule-for-main"
                },
                "ruleContentSha256": "4711b576EXAMPLE"
```

```
            }
        ],
        "authorArn": "arn:aws:iam::111111111111:user/Jane_Doe",
        "description": "Please review these changes by Tuesday",
        "title": "Pronunciation difficulty analyzer",
        "pullRequestTargets": [
            {
                "destinationCommit": "5d036259EXAMPLE",
                "destinationReference": "refs/heads/main",
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "317f8570EXAMPLE",
                "sourceReference": "refs/heads/jane-branch",
                "mergeMetadata": {
                    "isMerged": false
                }
            }
        ],
        "lastActivityDate": 1508962823.285,
        "pullRequestId": "42",
        "clientRequestToken": "123Example",
        "pullRequestStatus": "OPEN",
        "creationDate": 1508962823.285
    }
}
```

Erstellen einer Genehmigungsregel für eine Pull-Anforderung

Durch das Erstellen von Genehmigungsregeln für Pull-Anforderungen wird die Qualität des Codes sichergestellt, weil Benutzer die Pull-Anforderung genehmigen müssen, bevor der Code im Ziel-Branch zusammengeführt werden kann. Sie können die Anzahl der Benutzer angeben, die eine Pull-Anforderung genehmigen müssen. Sie können auch einen Genehmigungs-Pool von Benutzern für die Regel angeben. In diesem Fall werden nur Genehmigungen dieser Benutzer auf die Anzahl der erforderlichen Genehmigungen für die Regel angerechnet.

> **Note**
>
> Sie können auch Genehmigungsregelvorlagen erstellen, mit denen Sie die Erstellung von Genehmigungsregeln über Repositorys hinweg automatisieren können. Weitere Informationen finden Sie unter Arbeiten mit Genehmigungsregelvorlagen.

Sie können mit AWS CodeCommit-Konsole oder AWS CLI Regeln für das Repository erstellen.

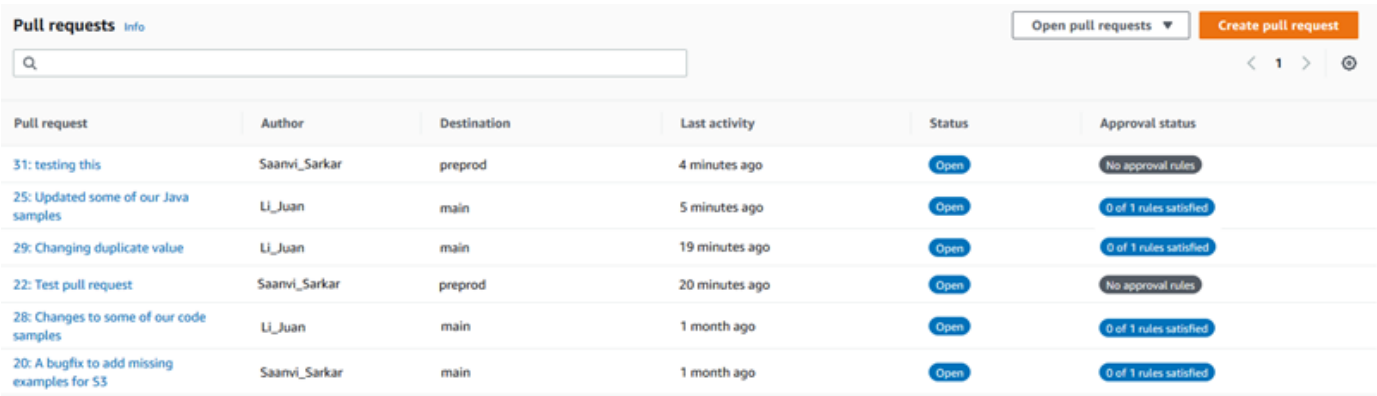
Themen

- Erstellen einer Pull-Anforderung
- Erstellen einer Genehmigungsregel für eine Pull-Anforderung (AWS CLI)

Erstellen einer Pull-Anforderung

Sie können das CodeCommit -Konsole, um eine Pull-Anforderung in einem CodeCommit -Repository.

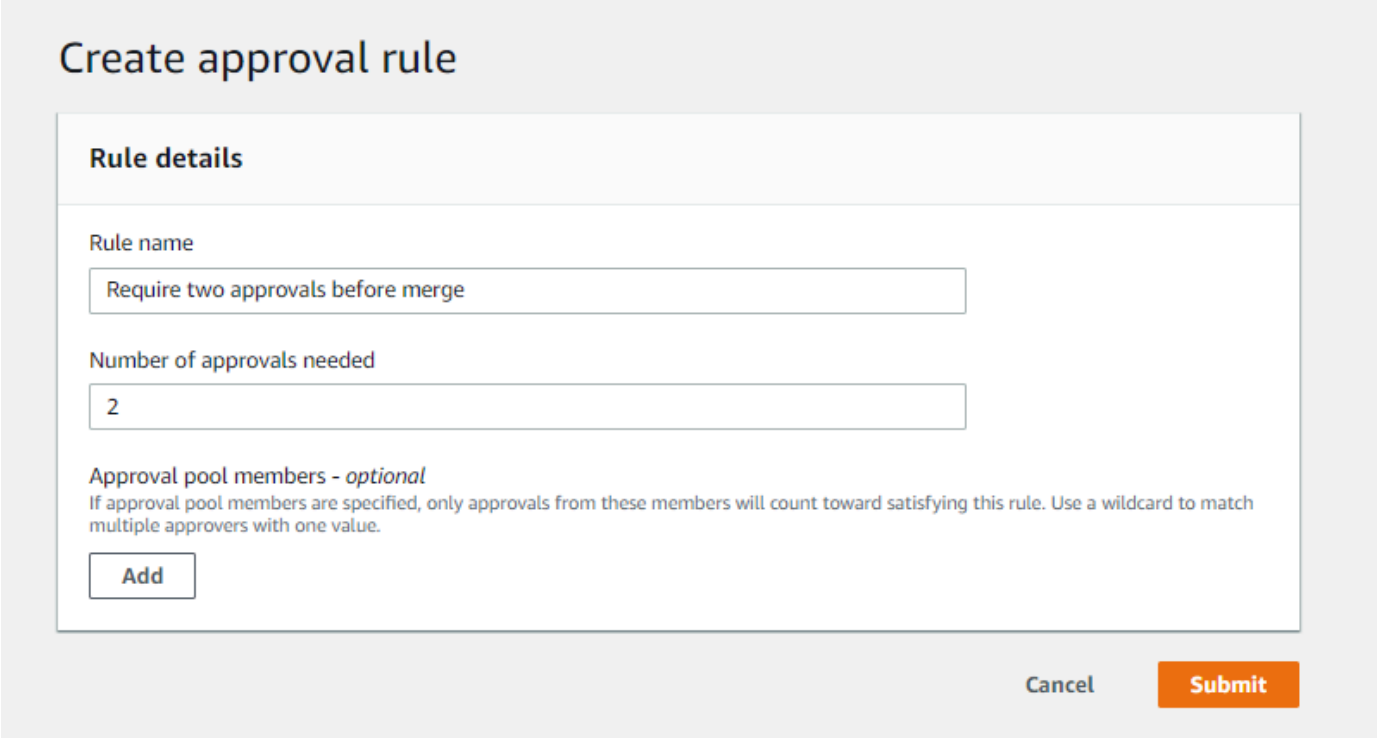
1. Öffnen Sie CodeCommit -Konsole beihttps://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem eine Genehmigungsregel für eine Pull-Anforderung erstellt werden soll.
3. Wählen Sie im Navigationsbereich Pull Requests aus.
4. Wählen Sie in der Liste die Pull-Anforderung aus, für die Sie eine Genehmigungsregel erstellen möchten. Sie können Genehmigungsregeln nur für offene Pull-Anforderungen erstellen.



5. Wählen Sie in der Pull-Anforderung Approvals (Genehmigungen) und dann Create approval rule (Genehmigungsregel erstellen) aus.
6. Geben Sie in das Feld Rule name (Regelname) einen aussagekräftigen Namen ein. Wenn Sie z. B. möchten, dass eine Pull-Anforderung von zwei Personen genehmigt werden muss, bevor sie zusammengeführt werden kann, können Sie die Regel beispielsweise **`Require two approvals before merge`** nennen.

> **Note**
>
> Sie können den Namen einer Genehmigungsregel nicht mehr ändern, nachdem sie erstellt wurde.

Geben Sie in das Feld Number of approvals needed (Anzahl erforderlicher Genehmigungen) die gewünschte Anzahl ein. Der Standardwert ist 1.



7. (Optional) Wenn Sie voraussetzen möchten, dass die Genehmigungen für eine Pull-Anforderung von einer bestimmten Gruppe von Benutzern stammen, wählen Sie unter Approval rule members (Genehmigungsregelmitglieder) die Option Add (Hinzufügen) aus. Wählen Sie unter Approver type (Genehmigertyp) eine der folgenden Optionen aus:

   - IAM-Benutzername oder übernommene Rolle: Diese Option füllt dasAWS-Konto-ID mit dem Konto ein, mit dem Sie sich angemeldet haben. Vorausgesetzt wird nur ein Name. Sie kann sowohl für IAM-Benutzer als auch für Verbundzugriffsbenutzer verwendet werden, deren Name dem angegebenen Namen entspricht. Dies ist eine sehr leistungsfähige Option, die hohe Flexibilität bietet. Wenn Sie beispielsweise mit dem Amazon-Web-Services-Konto (123456789012) angemeldet sind, diese Option auswählen,**`Mary_Major`**, werden alle folgenden Genehmigungen als Genehmigungen dieses Benutzers gezählt:

- Ein IAM-Benutzer im Konto (`arn:aws:iam::123456789012:user/Mary_Major`)
- Ein Verbundbenutzer in IAM mit dem Namen Mary_Major (`arn:aws:sts::123456789012:federated-user/Mary_Major`)

Diese Option erkennt keine aktive Sitzung eines Benutzers, der die Rolle **`CodeCommitReview`** angenommen hat und den Rollensitzungsnamen Mary_Major (`arn:aws:sts::123456789012:assumed-role/CodeCommitReview/Mary_Major`) verwendet, sofern Sie kein Platzhalterzeichen angegeben (`*Mary_Major`). Sie können den Rollennamen auch explizit angeben (`CodeCommitReview/Mary_Major`).

- Vollständig qualifiziertes ARN: Mit dieser Option können Sie den vollständig qualifizierten Amazon-Ressourcennamen (ARN) des IAM-Benutzers oder der IAM-Rolle angeben. Diese Option unterstützt auch angenommene Rollen, die von anderen AWS-Services wie AWS Lambda und AWS CodeBuild verwendet werden. Bei angenommenen Rollen sollte das ARN-Format bei Rollen „`arn:aws:sts::`*`AccountID`*`:assumed-role/`*`RoleName`*“ und bei Funktionen „`arn:aws:sts::`*`AccountID`*`:assumed-role/`*`FunctionName`*“ sein.

Wenn Sie gewählt habenIAM-Benutzername oder übernommene Rollewie der Genehmigende eingibt, inWert, geben Sie den Namen des IAM-Benutzers oder der IAM-Rolle ein. Wählen Sie erneut Add (Hinzufügen) aus, um weitere Benutzer oder Rollen hinzuzufügen, bis Sie alle Benutzer oder Rollen hinzugefügt haben, deren Genehmigungen auf die Anzahl der erforderlichen Genehmigungen angerechnet werden sollen.

Für beide Genehmigertypen können Platzhalterzeichen (*) in den Werten verwendet werden. Wenn Sie beispielsweiseIAM-Benutzername oder übernommene Rolle-Option, und Sie spezifizieren**`CodeCommitReview/*`**, alle Benutzer, die die Rolle des**`CodeCommitReview`**werden im Genehmigungspool gezählt. Die individuellen Rollensitzungsnamen zählen zum Erreichen der erforderlichen Anzahl von Genehmigern. Auf diese Weise zählen sowohl Mary_Major als auch Li_Juan als Genehmigungen, wenn sie angemeldet sind und die Rolle `CodeCommitReview` annehmen. Weitere Informationen zu IAM-ARNs, Platzhaltern und Formaten finden Sie unterIAM-IDs.

> **Note**
>
> Genehmigungsregeln unterstützen keine kontoübergreifenden Genehmigungen.

8. Wählen Sie nach dem Konfigurieren der Genehmigungsregel Submit (Senden).

Erstellen einer Genehmigungsregel für eine Pull-Anforderung (AWS CLI)

Um zu verwendenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

So erstellen Sie eine Genehmigungsregel für eine Pull-Anforderung in einem CodeCommit Endlager

1. Führen Sie den Befehl create-pull-request-approval-rule aus und geben Sie Folgendes an:

   - ID der Pull-Anforderung (mit der Option --id).
   - Name der Genehmigungsregel (mit der Option --approval-rule-name).
   - Inhalt der Genehmigungsregel (mit der Option --approval-rule-content).

   Wenn Sie die Genehmigungsregel erstellen, können Sie Genehmiger in einem Genehmigungs-Pool auf zwei Arten angeben:

   - CodeCommitApprovers: Diese Option setzt nur ein Amazon-Web-Services-Konto und eine Ressource voraus. Sie kann sowohl für IAM-Benutzer als auch für Verbundzugriffsbenutzer verwendet werden, deren Name dem angegebenen Ressourcennamen entspricht. Dies ist eine sehr leistungsfähige Option, die hohe Flexibilität bietet. Wenn Sie beispielsweise das Amazon Web Services Services-Konto 123456789012 angeben und**`Mary_Major`**, werden alle folgenden Genehmigungen als Genehmigungen dieses Benutzers gezählt:
     - Ein IAM-Benutzer im Konto (`arn:aws:iam::123456789012:user/Mary_Major`)
     - Ein Verbundbenutzer in IAM mit dem Namen Mary_Major (`arn:aws:sts::123456789012:federated-user/Mary_Major`)

     Diese Option erkennt keine aktive Sitzung eines Benutzers, der die Rolle **`CodeCommitReview`** angenommen hat und den Rollensitzungsnamen Mary_Major (`arn:aws:sts::123456789012:assumed-role/CodeCommitReview/Mary_Major`) verwendet, sofern Sie kein Platzhalterzeichen angegeben (`*Mary_Major`).
   - Vollständig qualifiziertes ARN: Mit dieser Option können Sie den vollständig qualifizierten Amazon-Ressourcennamen (ARN) des IAM-Benutzers oder der IAM-Rolle angeben.

   Weitere Informationen zu IAM-ARNs, Platzhaltern und Formaten finden Sie unterIAM-IDs.

Im folgenden Beispiel wird eine Genehmigungsregel namens`Require two approved approvers`für eine Pull-Anforderung mit der ID27. Die Regel gibt an, dass zwei Genehmigungen aus einem Genehmigungs-Pool erforderlich sind. Der Pool enthält alle Benutzer, die darauf zugreifen CodeCommit und übernimm die Rolle von**`CodeCommitReview`**im123456789012Amazon Web Services Services-Konto. Er enthält außerdem entweder einen IAM-Benutzer oder einen IAM-Benutzer namens`Nikhil_Jayashankar`im selben Amazon-Web-Services-Konto:

```
aws codecommit create-pull-request-approval-rule --pull-request-id 27
 --approval-rule-name "Require two approved approvers" --approval-
rule-content "{\"Version\": \"2018-11-08\",\"Statements\": [{\"Type\":
 \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers
\": [\"CodeCommitApprovers:123456789012:Nikhil_Jayashankar\",
 \"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "approvalRule": {
        "approvalRuleName": "Require two approved approvers",
        "lastModifiedDate": 1570752871.932,
        "ruleContentSha256": "7c44e6ebEXAMPLE",
        "creationDate": 1570752871.932,
        "approvalRuleId": "aac33506-EXAMPLE",
        "approvalRuleContent": "{\"Version\": \"2018-11-08\",\"Statements\":
 [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers
\": [\"CodeCommitApprovers:123456789012:Nikhil_Jayashankar\",
 \"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major"
    }
}
```

Anzeigen von Pull-Anforderungen in einemAWS CodeCommitEndlager

Sie können Pull-Anforderungen für Ihr Repository über die AWS CodeCommit-Konsole oder die AWS CLI anzeigen. Standardmäßig sehen Sie nur offene Pull-Anforderungen. Sie können jedoch den Filter

ändern und alle Pull-Anforderungen, nur geschlossene Anforderungen, nur von Ihnen erstellte Pull-Anforderungen usw. anzeigen.

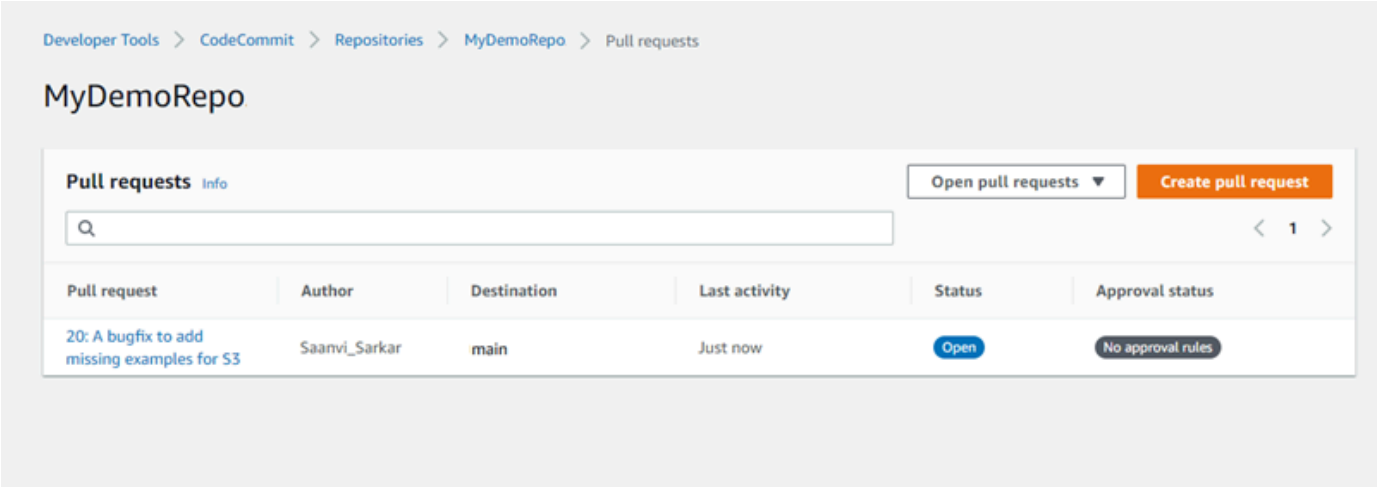
Themen

- Anzeigen von Pull-Anforderungen (Konsole)
- Anzeigen von Pull-Anforderungen (AWS CLI)

Anzeigen von Pull-Anforderungen (Konsole)

Sie können dasAWS CodeCommit-Konsole zum Anzeigen einer Liste mit Pull-Anforderungen in einem CodeCommit-Repository. Durch Änderung des Filters können Sie die Listenanzeige so ändern, dass nur eine bestimmte Gruppe von Pull-Anforderungen angezeigt wird. Beispielsweise können Sie eine Liste mit Pull-Anforderungen anzeigen, die Sie erstellt haben und deren Status Open ist, oder Sie können einen anderen Filter wählen und von Ihnen erstellte Pull-Anforderungen mit dem Status Closed ansehen.

1. Öffnen Sie die CodeCommit-Konsole unter.https://console.aws.amazon.com/codesuite/codecommit/homeaus.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Pull-Anforderungen anzeigen möchten.
3. Wählen Sie im Navigationsbereich Pull Requests aus.
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt.



5. Um den Anzeigefilter zu ändern, wählen Sie aus der Liste der verfügbaren Filter aus:

- Öffnen von Pull-Anforderungen(default): Zeigt alle Pull-Anforderungen mit dem Status vonÖffnenaus.
- Alle Pull-Anforderungen: Zeigt alle Pull-Anforderungen an.
- Geschlossene Pull-Anfragen: Zeigt alle Pull-Anforderungen mit dem Status vonClosed (Abgeschlossen)aus.
- Meine Pull-Anforderungen: Zeigt alle von Ihnen erstellten Pull-Anforderungen unabhängig vom Status an. Es werden keine Überprüfungen angezeigt, die Sie kommentiert haben, oder an denen Sie anderweitig beteiligt waren.
- Meine Open pull requests: Zeigt alle von Ihnen erstellten Pull-Anforderungen mit dem Status vonÖffnenaus.
- Meine geschlossenen Pull-Anfragen: Zeigt alle von Ihnen erstellten Pull-Anforderungen mit dem Status vonClosed (Abgeschlossen)aus.

6. Wenn Sie eine Pull-Anforderung in der angezeigten Liste finden, die Sie anzeigen möchten, wählen Sie diese aus.

Anzeigen von Pull-Anforderungen (AWS CLI)

Um zu verwendenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLIaus. Weitere Informationen finden Sie unter Befehlszeilenreferenz .

Führen Sie folgende Schritte aus, um dasAWS CLIUm Pull-Anforderungen in einem CodeCommit-Repository anzuzeigen.

1. Um eine Liste mit Pull-Anforderungen in einem Repository anzuzeigen, führen Sie den Befehl list-pull-requests aus. Geben Sie dabei Folgendes an:

   - Den Namen des CodeCommit-Repositorys, in dem Sie Pull-Anforderungen anzeigen möchten (mit dem--repository-nameOption).
   - (Optional) Den Status der Pull-Anforderung (mit der Option --pull-request-status).
   - (Optional) Den Amazon Resource Name (ARN) des IAM-Benutzers, der die Pull-Anfrage erstellt hat (mit dem--author-arnOption).
   - (Optional) Ein Aufzählungs-Token, das verwendet werden kann, um Ergebnisstapel zurückzugeben (mit der Option --next-token).
   - (Optional) Eine Einschränkung in Bezug auf die Anzahl der zurückgegebenen Ergebnisse pro Anforderung (mit der Option --max-results).

Zum Beispiel, um Pull-Anfragen aufzulisten, die von einem IAM-Benutzer mit dem ARN erstellt wurden*arn:aws:iam# 111111111:Benutzer/Li_Juan*und der Status von*GESCHLOSSEN*in einem CodeCommit-Repository mit dem Namen`MyDemoRepo`:

```
aws codecommit list-pull-requests --author-arn arn:aws:iam::111111111111:user/
Li_Juan --pull-request-status CLOSED --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
   "nextToken": "",
   "pullRequestIds": ["2","12","16","22","23","35","30","39","47"]
}
```

Pull-Anforderungs-IDs werden in der Reihenfolge der letzten Aktivität angezeigt.

2. Um Details zu einer bestimmten Pull-Anforderung anzuzeigen, führen Sie den Befehl get-pull-request mit der Option --pull-request-id aus, wobei Sie die ID der Pull-Anforderung angeben. So zeigen Sie beispielsweise Informationen zu einer Pull-Anforderung mit der ID *27* an:

```
aws codecommit get-pull-request --pull-request-id 27
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",\"Statements
\": [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers
\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
```

```
        "lastActivityDate": 1562619583.565,
        "pullRequestTargets": [
            {
                "sourceCommit": "ca45e279EXAMPLE",
                "sourceReference": "refs/heads/bugfix-1234",
                "mergeBase": "a99f5ddbEXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": false
                },
                "destinationCommit": "2abfc6beEXAMPLE",
                "repositoryName": "MyDemoRepo"
            }
        ],
        "revisionId": "e47def21EXAMPLE",
        "title": "Quick fix for bug 1234",
        "authorArn": "arn:aws:iam::123456789012:user/Nikhil_Jayashankar",
        "clientRequestToken": "d8d7612e-EXAMPLE",
        "creationDate": 1562619583.565,
        "pullRequestId": "27",
        "pullRequestStatus": "OPEN"
    }
}
```

3. Um Genehmigungen für eine Pull-Anforderung anzuzeigen, führen Sie den Befehl get-pull-request-approval-state aus. Geben Sie dabei Folgendes an:

   - ID der Pull-Anforderung (mit der Option --pull-request-id).
   - Revisions-ID der Pull-Anforderung (mit --revision-id option)). Mit dem Befehl get-pull-request können Sie die aktuelle Revisions-ID für eine Pull-Anforderung abrufen.

   Sie können beispielsweise Genehmigungen für eine Pull-Anforderung mit der ID *8* und der Revision-ID *9f29d167EXAMPLE* anzeigen:

   ```
   aws codecommit get-pull-request-approval-state --pull-request-id 8 --revision-
   id 9f29d167EXAMPLE
   ```

   Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
   ```

```
    "approvals": [
        {
            "userArn": "arn:aws:iam::123456789012:user/Mary_Major",
            "approvalState": "APPROVE"
        }
    ]
}
```

4. Um Ereignisse in einer Pull-Anforderung anzuzeigen, führen Sie den Befehl describe-pull-request-events mit der Option --pull-request-id aus, wobei Sie die ID der Pull-Anforderung angeben. Führen Sie folgenden Befehl aus, um beispielsweise Ereignisse zu einer Pull-Anforderung mit der ID *8* anzuzeigen:

```
aws codecommit describe-pull-request-events --pull-request-id 8
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "pullRequestEvents": [
        {
            "pullRequestId": "8",
            "pullRequestEventType": "PULL_REQUEST_CREATED",
            "eventDate": 1510341779.53,
            "actor": "arn:aws:iam::111111111111:user/Zhang_Wei"
        },
        {
            "pullRequestStatusChangedEventMetadata": {
                "pullRequestStatus": "CLOSED"
            },
            "pullRequestId": "8",
            "pullRequestEventType": "PULL_REQUEST_STATUS_CHANGED",
            "eventDate": 1510341930.72,
            "actor": "arn:aws:iam::111111111111:user/Jane_Doe"
        }
    ]
}
```

5. Wenn Sie sehen möchten, ob für eine Pull-Anforderung Konflikte für die Zusammenführung vorliegen, führen Sie den Befehl get-merge-conflicts aus. Geben Sie dabei Folgendes an:

   - Der Name des CodeCommit-Repositorys (mit dem--repository-nameOption).

- Branch, Tag, HEAD oder eine andere vollständig qualifizierte Referenz für die Quelle der Änderungen, die bei der Bewertung der Zusammenführung verwendet werden soll (mit der Option --source-commit-specifier).
- Branch, Tag, HEAD oder eine andere vollständig qualifizierte Referenz für das Ziel der Änderungen, das bei der Bewertung der Zusammenführung verwendet werden soll (mit der Option --destination-commit-specifier).
- Die zu verwendende Zusammenführungsoption (mit der Option --merge-option).

Wenn Sie anzeigen möchten, ob Konflikte für die Zusammenführung zwischen der Spitze des Source-Branch namens vorliegen*meine-Feature-Zweig*und ein Zielzweig mit dem Namen*Haupt*in einem Repository namensMyDemoRepo:

```
aws codecommit get-merge-conflicts --repository-name MyDemoRepo --source-commit-
specifier my-feature-branch --destination-commit-specifier main --merge-option
 FAST_FORWARD_MERGE
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "destinationCommitId": "fac04518EXAMPLE",
    "mergeable": false,
    "sourceCommitId": "16d097f03EXAMPLE"
}
```

Überprüfen einer Pull-Anforderung

Du kannst die AWS CodeCommit Konsole verwenden, um die in einer Pull-Anfrage enthaltenen Änderungen zu überprüfen. Sie können der Anforderung, den Dateien und einzelnen Codezeilen Kommentare hinzufügen. Sie können auch auf Kommentare von anderen Benutzern antworten. Wenn Ihr Repository mit Benachrichtigungen konfiguriert ist, erhalten Sie E-Mails, wenn Benutzer auf Ihre Kommentare antworten oder eine Pull-Anforderung kommentieren.

Sie können die verwenden AWS CLI , um eine Pull-Anfrage zu kommentieren und auf Kommentare zu antworten. Um die Änderungen zu überprüfen, müssen Sie die CodeCommit Konsole, den git diff Befehl oder ein Diff-Tool verwenden.

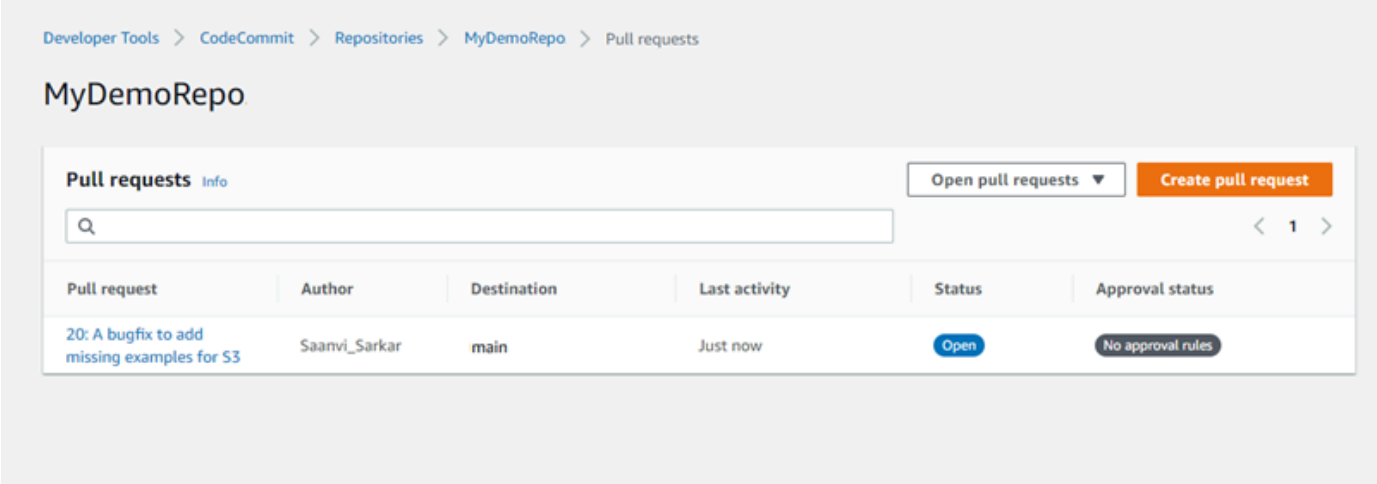
Themen

- Überprüfen Sie eine Pull-Anfrage (Konsole)
- Überprüfen Sie Pull-Requests (AWS CLI)

Überprüfen Sie eine Pull-Anfrage (Konsole)

Sie können die CodeCommit Konsole verwenden, um eine Pull-Anfrage in einem CodeCommit Repository zu überprüfen.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie im Bereich Repositories (Repositorys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie überprüfen möchten.

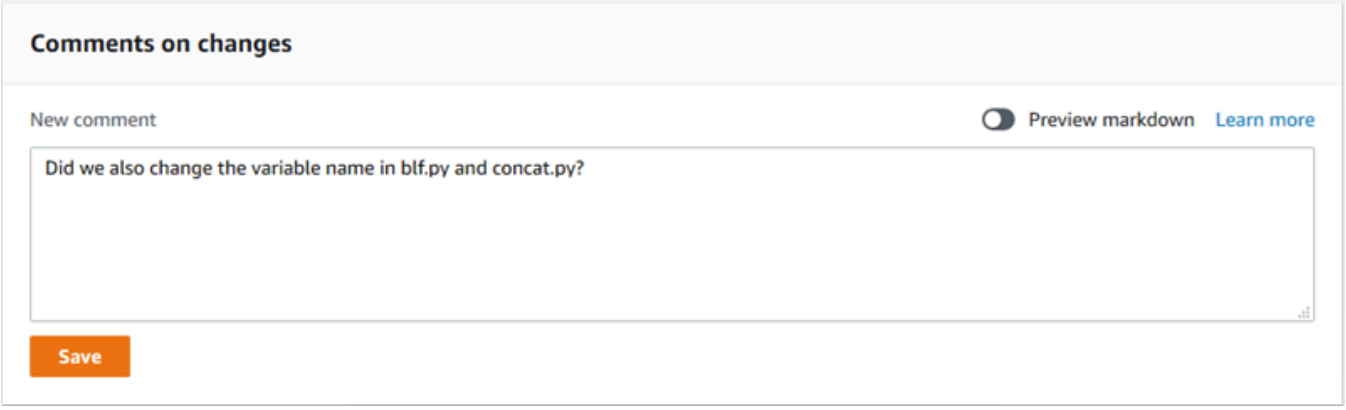


> **Note**
>
> Sie können eine geschlossene oder zusammengeführte Pull-Anforderung kommentieren, aber Sie können sie nicht zusammenführen oder erneut öffnen.

5. Wählen Sie in der Pull-Anforderung Changes.
6. Führen Sie eine der folgenden Aktionen aus:

   - Um einen Kommentar zur Pull-Anforderung insgesamt hinzuzufügen, geben Sie unter Comments on changes (Kommentare zu Änderungen), New comment (Neuer Kommentar)

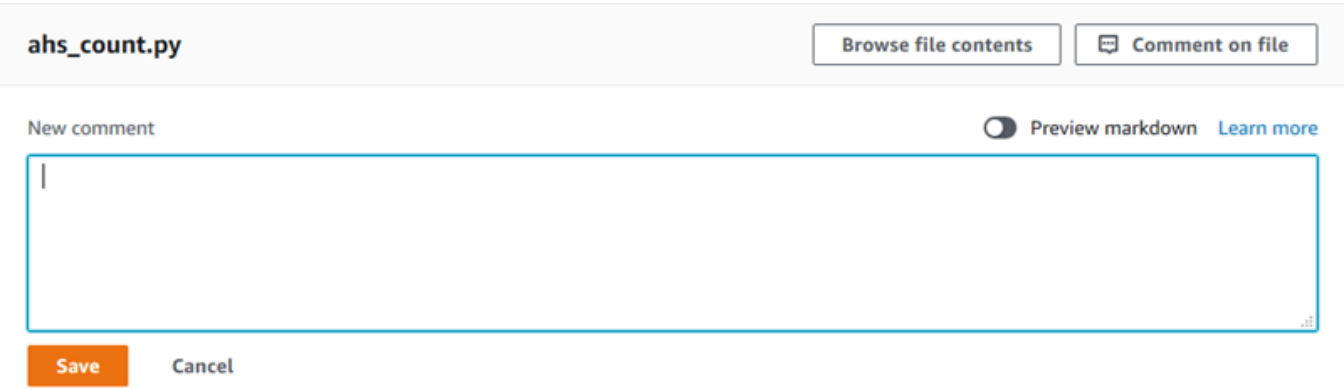
einen Kommentar ein und wählen Sie dann Save (Speichern) aus. Sie können Markdown verwenden oder Ihren Kommentar als Klartext eingeben.



- Um einer Datei im Commit einen Kommentar hinzuzufügen, suchen Sie in Changes den Namen der Datei. Wählen Sie das Kommentarsymbol

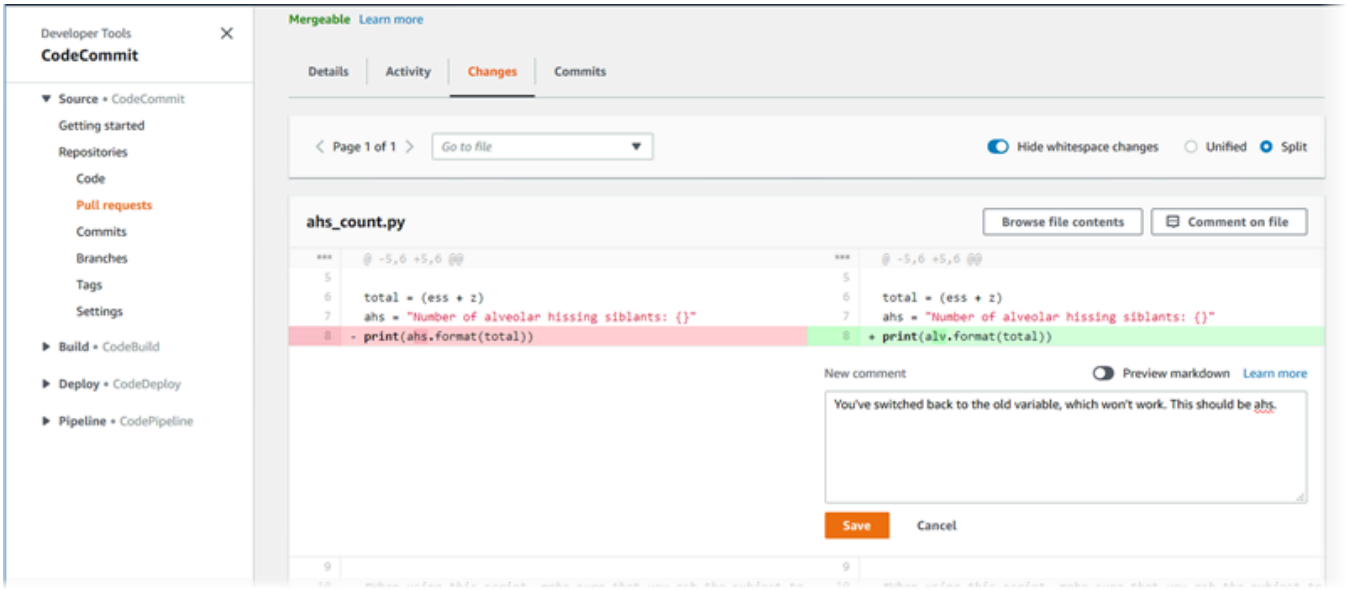
  neben dem Dateinamen aus, geben Sie einen Kommentar ein und wählen Sie dann Save (Speichern) aus.



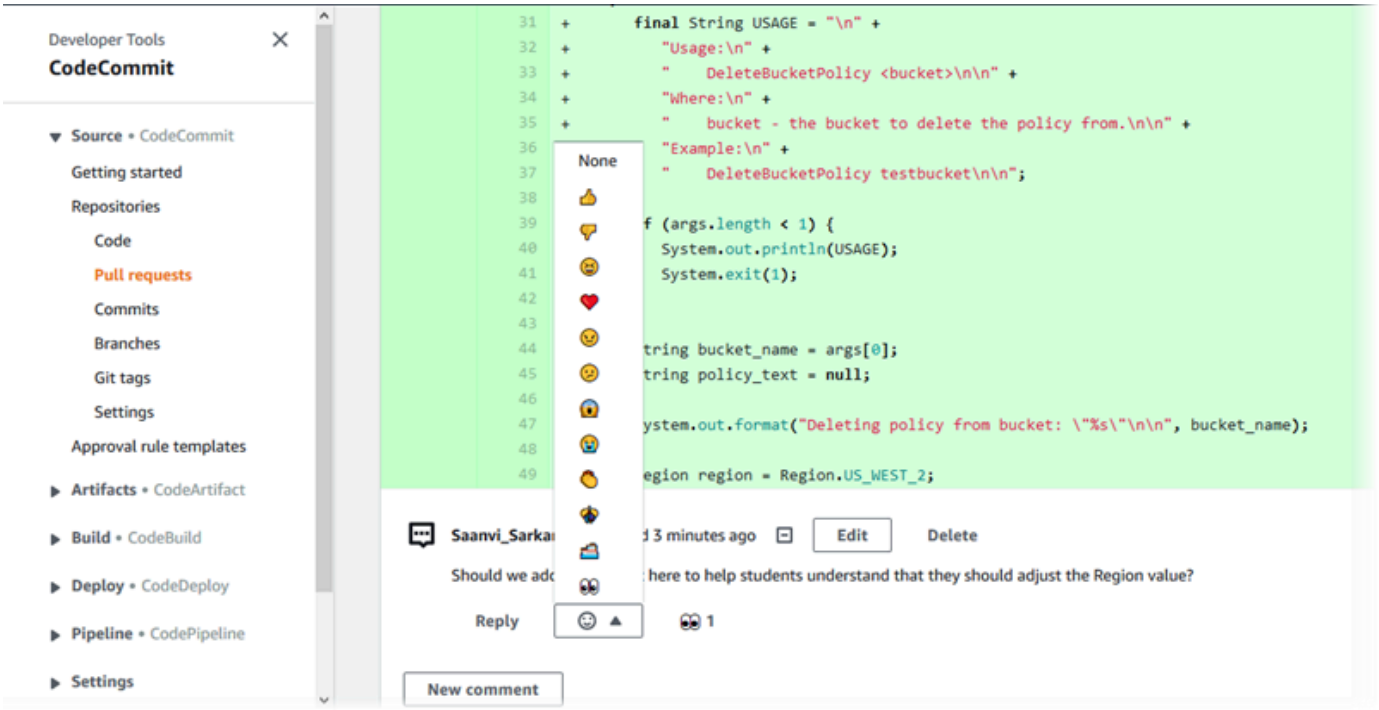
- Um einer geänderte Zeile in der Pull-Anfrage einen Kommentar hinzuzufügen, suchen Sie in Changes (Änderungen) die Zeile, die kommentiert werden soll. Wählen Sie das für die Zeile angezeigte Kommentarsymbol

  aus, geben Sie einen Kommentar ein und wählen Sie dann Save (Speichern) aus.

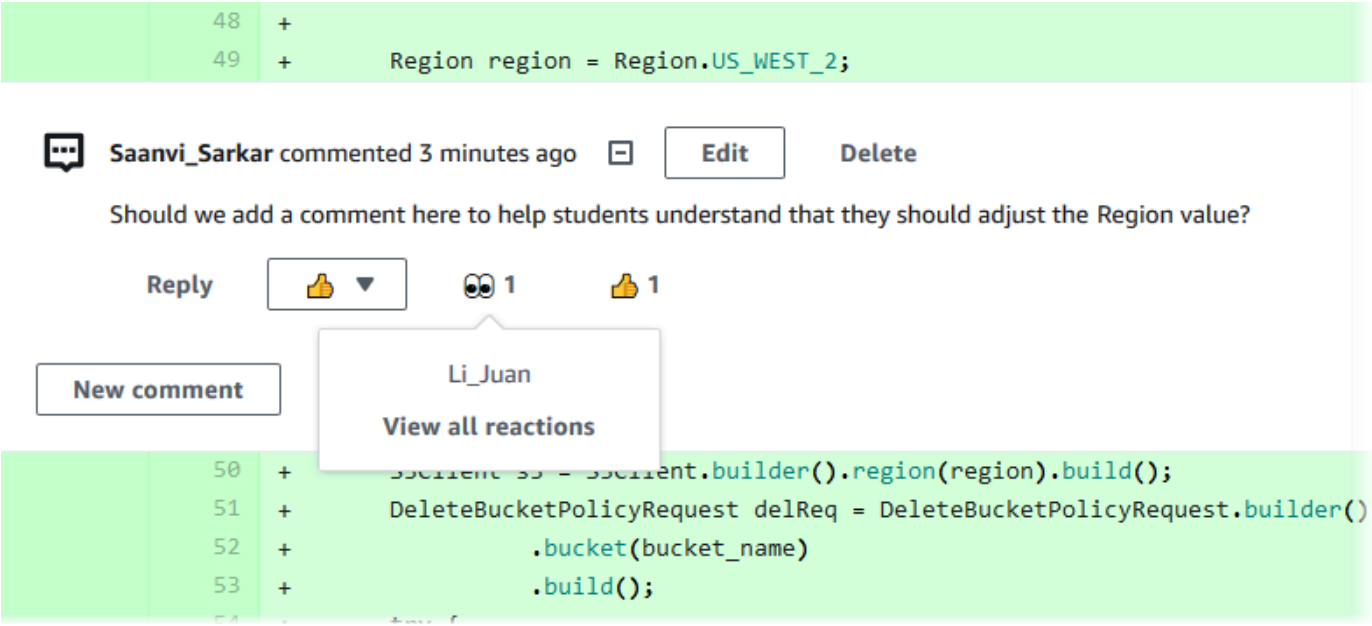


7. Zum Beantworten von Kommentaren im Commit wählen Sie unter Changes oder Activity die Option Reply aus. Sie können mit Text und Emojis antworten.



Du kannst dir die Namen der Personen ansehen, die mit einer bestimmten Emoji-Reaktion geantwortet haben, indem du sie auswählst. Um alle Emoji-Reaktionen und Informationen darüber zu sehen, wer mit welchen Emojis geantwortet hat, wähle Alle Reaktionen anzeigen. Wenn du mit einem Emoji auf einen Kommentar geantwortet hast, wird deine Antwort im Symbol für die Emoji-Reaktionsschaltfläche angezeigt.

> **Note**
>
> Die Anzahl der Reaktionen, die in der Konsole angezeigt werden, entspricht dem Zeitpunkt, zu dem die Seite geladen wurde. Aktuellste Informationen zur Anzahl der Emoji-Reaktionen erhalten Sie, wenn Sie entweder die Seite aktualisieren oder „Alle Reaktionen anzeigen“ wählen.

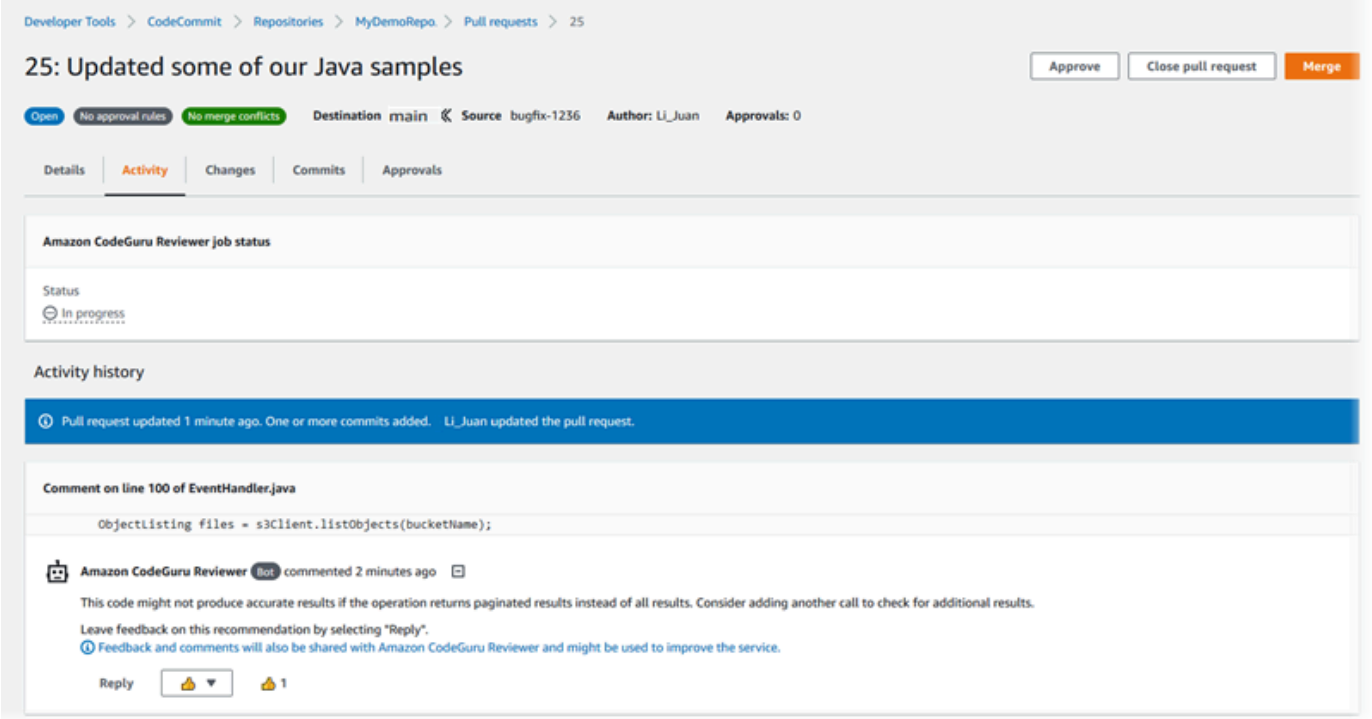


8. (Optional) Um auf eine von Amazon CodeGuru Reviewer erstellte Empfehlung zu antworten und auch Feedback zur Qualität der Empfehlung zu geben, wählen Sie Antworten. Verwenden Sie die Reaktionsschaltflächen, um allgemein anzugeben, ob Sie die Empfehlung genehmigen oder ablehnen. Verwenden Sie das Kommentarfeld, um weitere Details zur Reaktion anzugeben.

> **Note**
>
> Amazon CodeGuru Reviewer ist ein automatisierter Code-Review-Service, der Programmanalyse und maschinelles Lernen verwendet, um häufig auftretende Probleme zu erkennen und Korrekturen in Ihrem Java- oder Python-Code zu empfehlen.
>
> - Sie sehen Amazon CodeGuru Reviewer-Kommentare nur, wenn Sie das Repository mit Amazon CodeGuru Reviewer verknüpft haben, wenn die Analyse abgeschlossen ist und wenn der Code in der Pull-Anfrage Java- oder Python-Code ist. Weitere

Informationen finden Sie unter Ein AWS CodeCommit Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben.

- Kommentare von Amazon CodeGuru Reviewer werden nur dann auf der Registerkarte „Änderungen“ angezeigt, wenn sie zur letzten Version des Pull-Requests abgegeben wurden. Sie werden immer auf der Registerkarte Aktivität angezeigt.
- Sie können zwar mit allen verfügbaren Emoji-Reaktionen auf Empfehlungen von Amazon CodeGuru Rezensenten antworten, aber nur Emoji-Reaktionen mit Daumen hoch und Daumen runter werden verwendet, um die Nützlichkeit der Empfehlung zu bewerten.



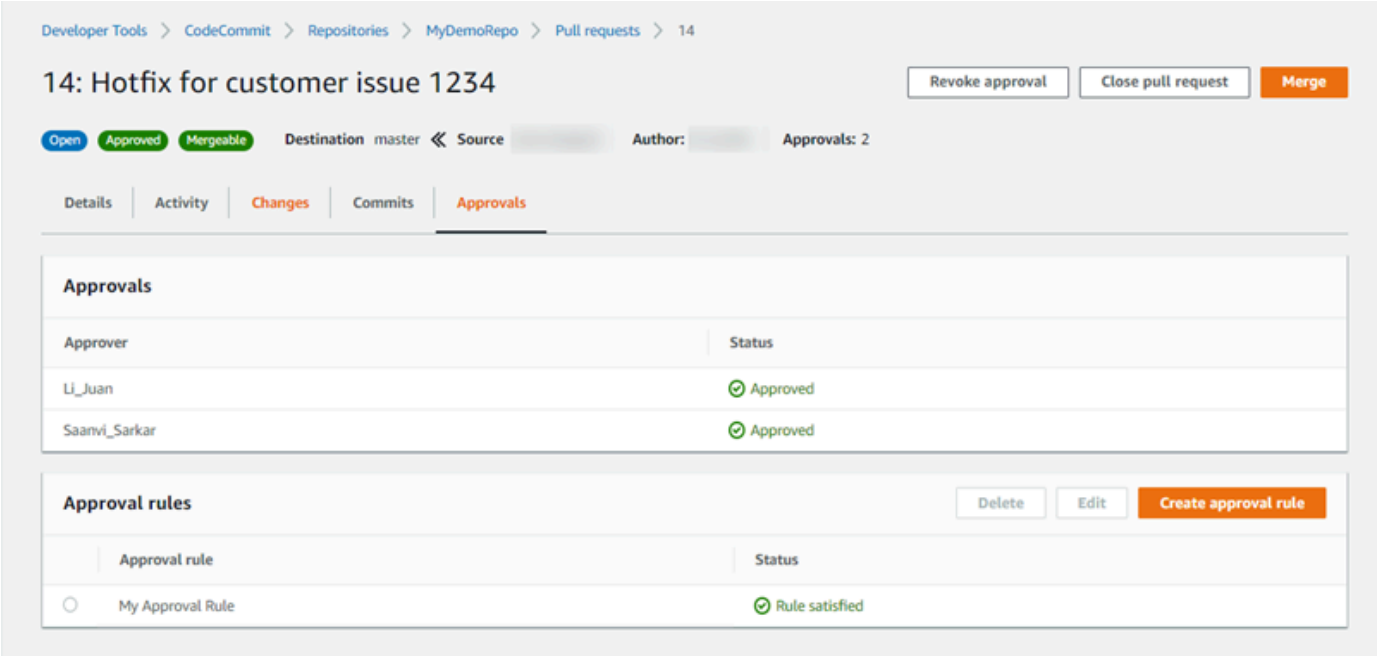
9. Um die in einer Pull-Anforderung vorgenommenen Änderungen zu genehmigen, wählen Sie Approve (Genehmigen).

> **Note**
>
> Sie können eine von Ihnen erstellte Pull-Anforderung nicht genehmigen.

Genehmigungen, Genehmigungsregeln für eine Pull-Anforderung und von Genehmigungsregelvorlagen erstellte Genehmigungsregeln sehen Sie unter Approvals

(Genehmigungen). Wenn Sie die Pull-Anforderung anschließend nicht genehmigen möchten, können Sie die Option Revoke approval (Genehmigung widerrufen) auswählen.

> **Note**
>
> Sie können Genehmigungen nur für offene Pull-Anforderungen bestätigen oder widerrufen. Sie können keine Genehmigung für eine Pull-Anforderung bestätigen oder widerrufen, die den Status „Merged (Zusammengeführt)“ oder „Closed (Geschlossen)“ hat.



Überprüfen Sie Pull-Requests (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

Sie können Pull-Requests mit den folgenden AWS CLI Befehlen überprüfen:

- post-comment-for-pull-request, um einen Kommentar zu einer Pull-Anfrage hinzuzufügen
- get-comments-for-pull-request, um Kommentare zu sehen, die zu einer Pull-Anfrage hinterlassen wurden

- update-pull-request-approval-state, um die Genehmigung für einen Pull-Request zu genehmigen oder zu widerrufen
- post-comment-reply, um auf einen Kommentar in einer Pull-Anfrage zu antworten

Du kannst Emojis auch mit Kommentaren in einer Pull-Anfrage mit den folgenden Befehlen verwenden:

- Um auf einen Kommentar mit einem Emoji zu antworten, führe den Befehl aus. put-comment-reaction
- Um Emoji-Reaktionen auf einen Kommentar anzuzeigen, führe den Befehl aus. get-comment-reactions

Um den zu verwenden AWS CLI , um Pull-Requests in einem CodeCommit Repository zu überprüfen

1. Um einer Pull-Anforderung in einem Repository einen Kommentar hinzuzufügen, führen Sie den Befehl post-comment-for-pull-request aus. Geben Sie dabei Folgendes an:
   - ID der Pull-Anforderung (mit der Option --pull-request-id).
   - Den Namen des Repositorys, das die Pull-Anforderung enthält (mit der Option --repository-name).
   - Die vollständige Commit-ID des Commits im Ziel-Branch, in dem die Pull-Anforderung zusammengeführt wird (mit der Option --before-commit-id).
   - Die vollständige Commit-ID des Commits im Quell-Branch, die die aktuelle Spitze des Branch für die Pull-Anforderung darstellt, wenn Sie den Kommentar posten (mit der Option --after-commit-id).
   - Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option --client-request-token).
   - Der Inhalt Ihres Kommentars (mit der Option --content).
   - Eine Liste der Speicherortangaben, wo der Kommentar abgelegt werden soll, unter anderem:
     - Der Name der zu vergleichenden Datei, einschließlich der Erweiterung und des Unterverzeichnisses (mit dem Attribut filePath).
     - Die Zeilennummer der Änderung innerhalb einer verglichenen Datei (mit dem Attribut filePosition).
     - Die Angabe, ob der Kommentar zu der Änderung „vor“ oder „nach“ dem Vergleich zwischen den Quell- und Ziel-Branches steht (mit dem Attribut relativeFileVersion).

Verwenden Sie diesen Befehl beispielsweise, um den Kommentar hinzuzufügen*: „Diese scheinen nirgends verwendet zu werden. Können wir sie entfernen?“* über die Änderung an der Datei *ahs_count.py* in einem Pull-Request mit der ID *47* in einem Repository mit dem Namen *MyDemoRepo*.

```
aws codecommit post-comment-for-pull-request --pull-request-id "47" --
repository-name MyDemoRepo --before-commit-id 317f8570EXAMPLE --after-
commit-id 5d036259EXAMPLE --client-request-token 123Example --content
 "These don't appear to be used anywhere. Can we remove them?" --location
 filePath=ahs_count.py,filePosition=367,relativeFileVersion=AFTER
```

Ist der Befehl erfolgreich, wird eine Ausgabe wie die folgende erzeugt.

```
{
        "afterBlobId": "1f330709EXAMPLE",
        "afterCommitId": "5d036259EXAMPLE",
        "beforeBlobId": "80906a4cEXAMPLE",
        "beforeCommitId": "317f8570EXAMPLE",
        "comment": {
              "authorArn": "arn:aws:iam::111111111111:user/Saanvi_Sarkar",
              "clientRequestToken": "123Example",
              "commentId": "abcd1234EXAMPLEb5678efgh",
              "content": "These don't appear to be used anywhere. Can we remove
 them?",
              "creationDate": 1508369622.123,
              "deleted": false,
              "lastModifiedDate": 1508369622.123,
              "callerReactions": [],
              "reactionCounts": []
           }
            "location": {
              "filePath": "ahs_count.py",
              "filePosition": 367,
              "relativeFileVersion": "AFTER"
            },
        "repositoryName": "MyDemoRepo",
        "pullRequestId": "47"
 }
```

2. Zum Anzeigen von Kommentaren für eine Pull-Anforderung führen Sie den Befehl get-comments-for-pull-request aus. Geben Sie dabei Folgendes an:

   - Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
   - Die vom System generierte ID der Pull-Anforderung (mit der Option `--pull-request-id`).
   - (Optional) Ein Aufzählungs-Token zum Zurückgeben des nächsten Ergebnisstapels (mit der Option `--next-token`).
   - (Optional) Eine nicht negative ganze Zahl, um die Anzahl der zurückgegebenen Ergebnisse zu begrenzen (mit der Option `--max-results`).

   Verwenden Sie diesen Befehl beispielsweise, um Kommentare für eine Pull-Anforderung mit der ID 42 anzuzeigen.

```
aws codecommit get-comments-for-pull-request --pull-request-id 42
```

   Ist der Befehl erfolgreich, wird eine Ausgabe wie die folgende erzeugt.

```
{
   "commentsForPullRequestData": [
      {
         "afterBlobId": "1f330709EXAMPLE",
         "afterCommitId": "5d036259EXAMPLE",
         "beforeBlobId": "80906a4cEXAMPLE",
         "beforeCommitId": "317f8570EXAMPLE",
         "comments": [
            {
               "authorArn": "arn:aws:iam::111111111111:user/Saanvi_Sarkar",
               "clientRequestToken": "",
               "commentId": "abcd1234EXAMPLEb5678efgh",
               "content": "These don't appear to be used anywhere. Can we remove
 them?",
               "creationDate": 1508369622.123,
               "deleted": false,
               "lastModifiedDate": 1508369622.123,
               "callerReactions": [],
               "reactionCounts":
                {
                  "THUMBSUP" : 6,
                  "CONFUSED" : 1
                }
```

```
            },
            {
               "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
               "clientRequestToken": "",
               "commentId": "442b498bEXAMPLE5756813",
               "content": "Good catch. I'll remove them.",
               "creationDate": 1508369829.104,
               "deleted": false,
               "lastModifiedDate": 150836912.273,
               "callerReactions": ["THUMBSUP"]
               "reactionCounts":
                {
                  "THUMBSUP" : 14
                }
             }
         ],
         "location": {
            "filePath": "ahs_count.py",
            "filePosition": 367,
            "relativeFileVersion": "AFTER"
         },
         "repositoryName": "MyDemoRepo",
         "pullRequestId": "42"
      }
   ],
   "nextToken": "exampleToken"
}
```

3. Um die Genehmigung für eine Pull-Anforderung zu bestätigen oder zu widerrufen, führen Sie den Befehl update-pull-request-approval-state aus und geben Sie Folgendes an:

   - ID der Pull-Anforderung (mit der Option --pull-request-id).
   - Revisions-ID der Pull-Anforderung (mit --revision-id option)). Sie können die aktuelle Revisions-ID für einen Pull-Request mit dem get-pull-requestBefehl abrufen.
   - Der Genehmigungsstatus, den Sie zuweisen möchten (mit der Option --approval-state). Gültige Genehmigungsstatus sind `APPROVE` und `REVOKE`.

   Verwenden Sie diesen Befehl beispielsweise, um eine Pull-Anforderung mit der ID *27* und der Revision-ID *9f29d167EXAMPLE* zu genehmigen.

```
aws codecommit update-pull-request-approval-state --pull-request-id 27 --revision-
id 9f29d167EXAMPLE --approval-state "APPROVE"
```

Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.

4. Um eine Antwort auf einen Kommentar in einer Pull-Anforderung hinzuzufügen, führen Sie den Befehl post-comment-reply aus. Geben Sie dabei Folgendes an:

   - Die vom System generierte ID des Kommentars, auf den Sie antworten wollen (mit der Option --in-reply-to).
   - Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option --client-request-token).
   - Der Inhalt Ihrer Antwort (mit der Option --content).

   Verwenden Sie diesen Befehl beispielsweise, um die Antwort *„Guter catch“ hinzuzufügen. Ich werde sie entfernen.“ zum Kommentar mit der vom System generierten ID ABCD1234ExampleB5678EFGH.*

```
aws codecommit post-comment-reply --in-reply-to abcd1234EXAMPLEb5678efgh --
content "Good catch. I'll remove them." --client-request-token 123Example
```

Ist der Befehl erfolgreich, wird eine Ausgabe wie die folgende erzeugt.

```
{
    "comment": {
        "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
        "clientRequestToken": "123Example",
        "commentId": "442b498bEXAMPLE5756813",
        "content": "Good catch. I'll remove them.",
        "creationDate": 1508369829.136,
        "deleted": false,
        "lastModifiedDate": 150836912.221,
        "callerReactions": [],
        "reactionCounts": []
    }
 }
```

Aktualisieren einer Pull-Anforderung

Sie können eine Pull-Anforderung mit weiteren Codeänderungen aktualisieren, indem Sie Commits in den Quell-Branch einer offenen Pull-Anforderung senden. Weitere Informationen finden Sie unter Erstellen Sie einen Commit in AWS CodeCommit .

Sie können den Titel oder die Beschreibung einer Pull-Anforderung über die AWS CodeCommit-Konsole oder mit der AWS CLI aktualisieren. Sie können den Titel oder die Beschreibung der Pull-Anforderung aus folgenden Gründen aktualisieren:

- Andere Benutzer verstehen die Beschreibung nicht oder der ursprüngliche Titel ist irreführend.
- Sie möchten, dass der Titel oder die Beschreibung die Änderungen berücksichtigt, die am Quell-Branch einer offenen Pull-Anforderung vorgenommen wurden.

Aktualisieren einer Pull-Anforderung (-Konsole)

Sie können die CodeCommit-Konsole verwenden, um den Titel und die Beschreibung einer Pull-Anforderung in einem CodeCommit-Repository zu aktualisieren. Um den Code in der Pull-Anforderung zu aktualisieren, drücken Sie Commits in den Quell-Branch einer offenen Pull-Anforderung.

1. Öffnen Sie die CodeCommit-Konsole unter.https://console.aws.amazon.com/codesuite/codecommit/homeaus.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie eine Pull-Anforderungen aktualisieren möchten.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie aktualisieren möchten.
5. Wählen Sie in der Pull-Anforderung Details und dann Edit Details (Details bearbeiten) zum Bearbeiten des Titels oder der Beschreibung aus.

> **Note**
>
> Den Titel oder die Beschreibung einer geschlossenen oder zusammengeführten Pull-Anforderung können Sie nicht aktualisieren.

Aktualisieren von Pull-Anforderungen (AWS CLI)

Um zu verwendenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLIaus. Weitere Informationen finden Sie unter Befehlszeilenreferenz .

Auch folgende Befehle können interessant sein:

- update-pull-request-approval-state, um die Genehmigung für eine Pull-Anforderung zu gewähren oder zu widerrufen.
- create-pull-request-approval-rule, um eine Genehmigungsregel für eine Pull-Anforderung zu erstellen.
- delete-pull-request-approval-rule, um eine Genehmigungsregel für eine Pull-Anforderung zu löschen.
- Erstelle einen Commit dem AWS CLIoderEinen Commit mit einem Git-Client erstellen, um zusätzliche Codeänderungen zu erstellen und in den Quell-Branch einer offenen Pull-Anforderung zu senden.

So verwenden Sie denAWS CLISo aktualisieren Pull-Anforderungen in einem CodeCommit-Repository

1. Um den Titel einer Pull-Anforderung in einem Repository zu aktualisieren, führen Sie den Befehl update-pull-request-title aus, wobei Sie Folgendes angeben:

   - ID der Pull-Anforderung (mit der Option --pull-request-id).
   - Den Titel der Pull-Anforderung (mit der Option --title).

   Um beispielsweise den Titel einer Pull-Anforderung mit der ID *47* zu aktualisieren, schreiben Sie:

   ```
   aws codecommit update-pull-request-title --pull-request-id 47 --title
    "Consolidation of global variables - updated review"
   ```

   Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "pullRequest": {
           "approvalRules": [
               {
   ```

```
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "originApprovalRuleTemplate": {
                    "approvalRuleTemplateId": "dd8b26gr-EXAMPLE",
                    "approvalRuleTemplateName": "2-approver-rule-for-main"
                },
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
        "clientRequestToken": "",
        "creationDate": 1508530823.12,
        "description": "Review the latest changes and updates to the global
 variables. I have updated this request with some changes, including removing some
 unused variables.",
        "lastActivityDate": 1508372657.188,
        "pullRequestId": "47",
        "pullRequestStatus": "OPEN",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": false,
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables - updated review"
    }
}
```

2. Um die Beschreibung einer Pull-Anforderung zu aktualisieren, führen Sie den Befehl update-pull-request-description aus, wobei Sie Folgendes angeben:

- ID der Pull-Anforderung (mit der Option --pull-request-id).
- Die Beschreibung (mit der --description-Option).

Um beispielsweise die Beschreibung einer Pull-Anforderung mit der ID *47* zu aktualisieren, schreiben Sie:

```
aws codecommit update-pull-request-description --pull-request-id 47 --description
 "Updated the pull request to remove unused global variable."
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
   "pullRequest": {
      "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
      "clientRequestToken": "",
      "creationDate": 1508530823.155,
      "description": "Updated the pull request to remove unused global variable.",
      "lastActivityDate": 1508372423.204,
      "pullRequestId": "47",
      "pullRequestStatus": "OPEN",
      "pullRequestTargets": [
         {
            "destinationCommit": "9f31c968EXAMPLE",
            "destinationReference": "refs/heads/main",
            "mergeMetadata": {
               "isMerged": false,
            },
            "repositoryName": "MyDemoRepo",
            "sourceCommit": "99132ab0EXAMPLE",
            "sourceReference": "refs/heads/variables-branch"
         }
      ],
      "title": "Consolidation of global variables"
   }
}
```

Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung

Falls eine Genehmigungsregel für eine Pull-Anforderung vorliegt, können Sie diese Pull-Anforderung erst zusammenführen, wenn die Bedingungen erfüllt sind. Sie können die Genehmigungsregeln für Pull-Anforderungen ändern, um die Erfüllung der Bedingungen zu erleichtern oder für striktere Prüfungen zu sorgen. Sie können die Anzahl der Benutzer ändern, die eine Pull-Anforderung genehmigen müssen. Sie können für die Regel Mitgliedschaften in einem Genehmigungs-Pool von Benutzern hinzufügen, entfernen oder ändern. Wenn Sie eine Genehmigungsregel für eine Pull-Anforderung nicht mehr verwenden möchten, können Sie diese löschen.

> **Note**
>
> Sie können Genehmigungsregeln für eine Pull-Anforderung auch außer Kraft setzen. Weitere Informationen finden Sie unter Überschreiben von Genehmigungsregeln für eine Pull-Anforderung .

Sie können die AWS CodeCommit-Konsole oder AWS CLI verwenden, um Genehmigungsregeln für das Repository zu bearbeiten und zu löschen.

Themen

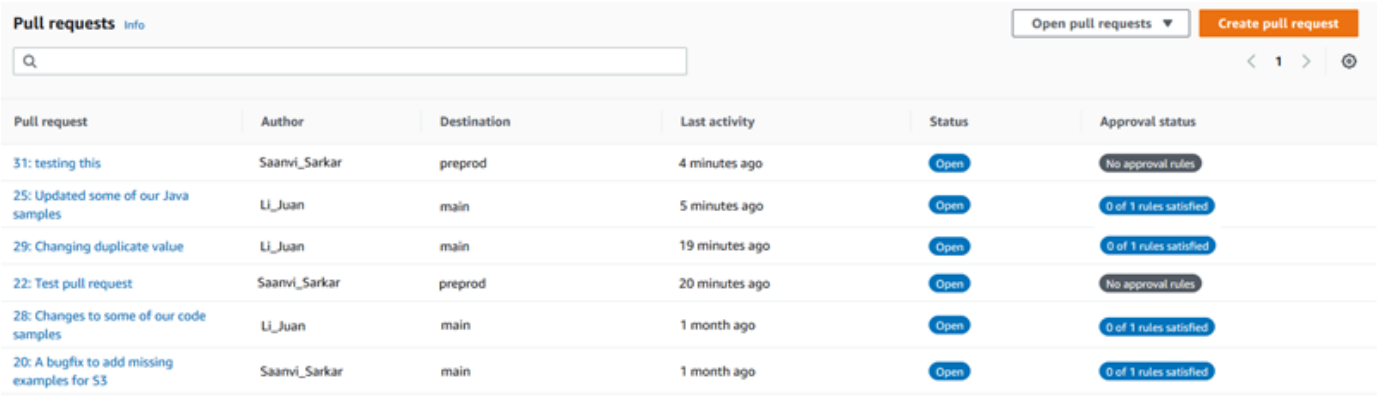
- Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung
- Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung (AWS CLI)

Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung

Sie können die CodeCommit-Konsole verwenden, um eine Genehmigungsregel für eine Pull-Anforderung in einem CodeCommit-Repository bearbeiten oder löschen.

1. Öffnen Sie die CodeCommit-Konsole unter.https://console.aws.amazon.com/codesuite/codecommit/homeaus.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie eine Genehmigungsregel für eine Pull-Anforderung bearbeiten oder löschen möchten.
3. Wählen Sie im Navigationsbereich Pull Requests aus.

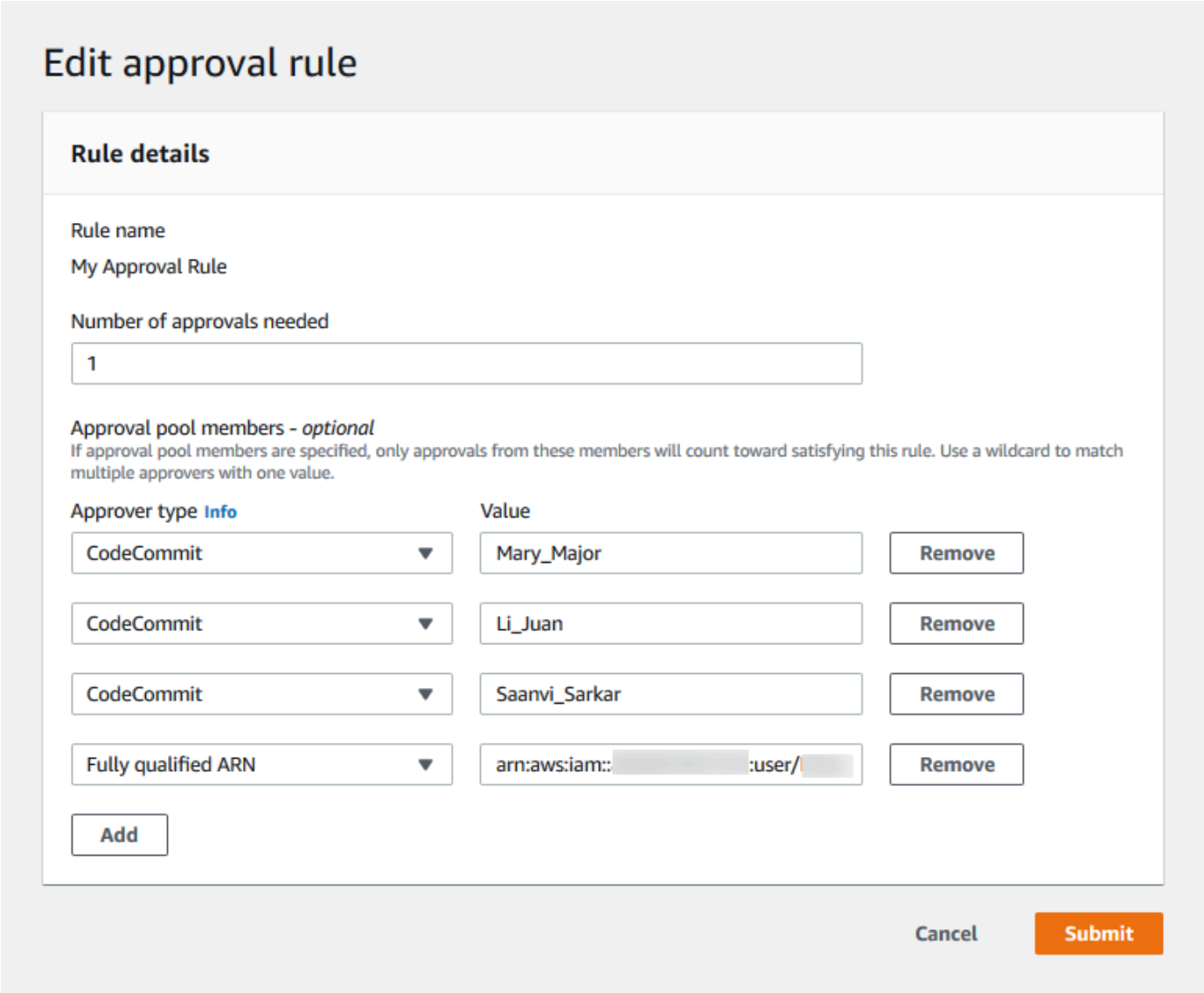
4. Wählen Sie die Pull-Anforderung aus, in der Sie eine Genehmigungsregel bearbeiten oder löschen möchten. Sie können Genehmigungsregeln nur für offene Pull-Anforderungen bearbeiten und löschen.



5. Wählen Sie in der Pull-Anforderung Approvals (Genehmigungen) und dann die Regel aus, die Sie bearbeiten oder aus der Liste löschen möchten. Führen Sie eine der folgenden Aufgaben aus:

   - Wenn Sie die Regel bearbeiten möchten, wählen Sie Edit (Bearbeiten) aus.
   - Wenn Sie die Regel löschen möchten, wählen Sie Delete (Löschen) aus und befolgen Sie dann die Anweisungen, mit denen das Löschen der Regel verifiziert werden kann.

6. Nehmen Sie unter Edit approval rule (Genehmigungsregel bearbeiten) die gewünschten Änderungen an der Regel vor und wählen Sie dann Submit (Senden).



7. Wählen Sie nach dem Konfigurieren der Genehmigungsregel Submit (Senden).

Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung (AWS CLI)

Um zu verwendenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLIaus. Weitere Informationen finden Sie unter Befehlszeilenreferenz .

Sie können mit der AWS CLI den Inhalt einer Genehmigungsregel bearbeiten und eine Genehmigungsregel löschen.

> **Note**
>
> Auch folgende Befehle können interessant sein:
>
> - update-pull-request-approval-state, um die Genehmigung für eine Pull-Anforderung zu gewähren oder zu widerrufen.
> - get-pull-request-approval-states, um die Genehmigungen für die Pull-Anforderung anzuzeigen.
> - evaluate-pull-request-approval-rules, um festzustellen, ob die Bedingungen der Genehmigungsregeln für eine Pull-Anforderung erfüllt sind.

So verwenden Sie denAWS CLISo verwenden Sie zum Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung in einem CodeCommit-Repository

1. Um eine Genehmigungsregel zu bearbeiten, führen Sie den Befehl update-pull-request-approval-rule-content aus, indem Sie Folgendes angeben:

   - ID der Pull-Anforderung (mit der Option --id).
   - Name der Genehmigungsregel (mit der Option --approval-rule-name).
   - Inhalt der Genehmigungsregel (mit der Option --approval-rule-content).

   In diesem Beispiel wird eine Genehmigungsregel namens *Require two approved approvers* für eine Pull-Anforderung mit der ID *27* aktualisiert. Die Regel erfordert die Genehmigung eines Benutzers aus einem Genehmigungspool, der jeden IAM-Benutzer in der*123456789012*Amazon Web Services Services-Konto:

   ```
   aws codecommit update-pull-request-approval-rule-content --pull-request-id 27
    --approval-rule-name "Require two approved approvers" --approval-rule-content
    "{Version: 2018-11-08, Statements: [{Type: \"Approvers\", NumberOfApprovalsNeeded:
    1, ApprovalPoolMembers:[\"CodeCommitApprovers:123456789012:user/*\"]}]}"
   ```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "approvalRule": {
   ```

```
        "approvalRuleContent": "{Version: 2018-11-08, Statements:
 [{Type: \"Approvers\", NumberOfApprovalsNeeded: 1, ApprovalPoolMembers:
[\"CodeCommitApprovers:123456789012:user/*\"]}]}",
        "approvalRuleId": "aac33506-EXAMPLE",
        "originApprovalRuleTemplate": {},
        "creationDate": 1570752871.932,
        "lastModifiedDate": 1570754058.333,
        "approvalRuleName": Require two approved approvers",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "ruleContentSha256": "cd93921cEXAMPLE",
    }
}
```

3. Um eine Genehmigungsregel zu löschen, führen Sie den Befehl delete-pull-request-approval-rule aus, indem Sie Folgendes angeben:

   - ID der Pull-Anforderung (mit der Option --id).
   - Name der Genehmigungsregel (mit der Option --approval-rule-name).

   So löschen Sie beispielsweise eine Genehmigungsregel mit dem Namen *My Approval Rule* für eine Pull-Anforderung mit der ID *15*:

   ```
   aws codecommit delete-pull-request-approval-rule --pull-request-id 15 --approval-
   rule-name "My Approval Rule"
   ```

   Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

   ```
   {
       "approvalRuleId": "077d8e8a8-EXAMPLE"
   }
   ```

Überschreiben von Genehmigungsregeln für eine Pull-Anforderung

Im normalen Verlauf der Entwicklung möchten Sie, dass Benutzer die Bedingungen der Genehmigungsregeln erfüllen, bevor Sie Pull-Anforderungen zusammenführen. Es kann jedoch vorkommen, dass Sie das Zusammenführen einer Pull-Anforderung beschleunigen müssen. Beispiel: Sie möchten eine Fehlerbehebung in die Produktion übernehmen, aber niemand im Genehmigungs-Pool ist verfügbar, um die Pull-Anforderung zu genehmigen. In solchen Fällen

können Sie die Genehmigungsregeln für eine Pull-Anforderung außer Kraft setzen. Sie können alle Genehmigungsregeln für eine Pull-Anforderung außer Kraft setzen, einschließlich derjenigen, die speziell für die Pull-Anforderung erstellt und aus einer Genehmigungsregelvorlage generiert wurden. Sie können nicht selektiv eine bestimmte Genehmigungsregel außer Kraft setzen, sondern nur alle Regeln. Nachdem Sie die Genehmigungsregelanforderungen durch Außerkraftsetzen der Regeln aufgehoben haben, können Sie die Pull-Anforderung mit dem Ziel-Branch zusammenführen.

Wenn Sie Genehmigungsregeln für eine Pull-Anforderung außer Kraft setzen, werden Informationen über den Benutzer, der die Regeln außer Kraft setzt, in der Aktivität für die Pull-Anforderung aufgezeichnet. Auf diese Weise können Sie den Verlauf einer Pull-Anforderung rückverfolgen und ermitteln, wer die Regeln außer Kraft gesetzt hat. Sie können die Außerkraftsetzung auch widerrufen, wenn die Pull-Anforderung noch geöffnet ist. Nachdem die Pull-Anforderung zusammengeführt wurde, können Sie die Außerkraftsetzung nicht mehr widerrufen.

Themen

- Genehmigungsregeln außer Kraft setzen (Konsole)
- Überschreiben von Genehmigungsregeln (AWS CLI)

Genehmigungsregeln außer Kraft setzen (Konsole)

Sie können die Anforderungen von Genehmigungsregeln für eine Pull-Anforderung in der Konsole im Rahmen der Prüfung einer Pull-Anforderung außer Kraft setzen. Wenn Sie Ihre Meinung ändern, können Sie die Außerkraftsetzung widerrufen. Die Genehmigungsregelanforderungen werden dann wieder angewendet. Sie können Genehmigungsregeln nur außer Kraft setzen oder eine Außerkraftsetzung widerrufen, wenn die Pull-Anforderung noch geöffnet ist. Nachdem sie zusammengeführt oder geschlossen wurde, können Sie den Außerkraftsetzungsstatus nicht mehr ändern.

1. Öffnen Sie die CodeCommit-Konsole unter.https://console.aws.amazon.com/codesuite/codecommit/homeaus.
2. Wählen Sie im Bereich Repositories (Repositorys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus. Wählen Sie die Pull-Anforderung, für die die Genehmigungsregel außer Kraft gesetzt werden soll, oder widerrufen Sie eine Außerkraftsetzung.
4. Wählen Sie auf der Registerkarte Approvals (Genehmigungen) die Option Override approval rules (Genehmigungsregel außer Kraft setzen) aus. Die Anforderungen werden verlagert und

der Schaltflächentext wird in Revoke override (Außerkraftsetzung widerrufen) geändert. Um die Genehmigungsregelanforderungen wieder anzuwenden, wählen Sie Revoke override (Außerkraftsetzung widerrufen).

Überschreiben von Genehmigungsregeln (AWS CLI)

Sie können die AWS CLI verwenden, um die Genehmigungsregelanforderungen außer Kraft zu setzen. Sie können auch den Außerkraftsetzungsstatus für eine Pull-Anforderung anzeigen.

So setzen Sie Genehmigungsregelanforderungen für eine Pull-Anforderung außer Kraft

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl override-pull-request-approval-rules unter Angabe der folgenden Informationen aus:

   - Die vom System generierte ID der Pull-Anforderung.
   - Die neueste Revisions-ID der Pull-Anforderung. Verwenden Sie get-pull-request, um diese Informationen anzuzeigen.
   - Der zu überschreibende Status: `OVERRIDE` oder `REVOKE`. Der Status `REVOKE` entfernt den Status `OVERRIDE`, wird aber nicht gespeichert.

   So setzen Sie beispielsweise Genehmigungsregeln für eine Pull-Anforderung mit der ID **34** und der Revisions-ID **927df8d8EXAMPLE** außer Kraft:

   ```
   aws codecommit override-pull-request-approval-rules --pull-request-id 34 --
   revision-id 927df8d8dEXAMPLE --override-status OVERRIDE
   ```

2. Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.
3. So widerrufen Sie die Außerkraftsetzung für eine Pull-Anforderung mit der ID **34** und der Revisions-ID **927df8d8EXAMPLE**:

   ```
   aws codecommit override-pull-request-approval-rules --pull-request-id 34 --
   revision-id 927df8d8dEXAMPLE --override-status REVOKE
   ```

So erhalten Sie Informationen zum Außerkraftsetzungsstatus einer Pull-Anforderung

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl get-pull-request-override-state unter Angabe der folgenden Informationen aus:

- Die vom System generierte ID der Pull-Anforderung.
- Die neueste Revisions-ID der Pull-Anforderung. Verwenden Sie get-pull-request, um diese Informationen anzuzeigen.

So zeigen Sie beispielsweise den Außerkraftsetzungsstatus für eine Pull-Anforderung mit der ID **34** und der Revisions-ID **927df8d8EXAMPLE** an:

```
aws codecommit get-pull-request-override-state --pull-request-id 34 --revision-
id 927df8d8dEXAMPLE
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "overridden": true,
    "overrider": "arn:aws:iam::123456789012:user/Mary_Major"
}
```

Zusammenführen einer Pull-Anforderung in einemAWS CodeCommitEndlager

Nachdem der Code geprüft wurde und alle Genehmigungsregeln (falls vorhanden) für die Pull-Anforderung erfüllt wurden, können Sie eine Pull-Anforderung auf verschiedene Arten zusammenführen:

- In der Konsole können Sie den Quell-Branch mit einer der verfügbaren Merge-Strategien mit dem Ziel-Branch zusammenführen, wodurch auch die Pull-Anforderung geschlossen wird. Sie können Zusammenführungskonflikte auch in der Konsole lösen. Die Konsole zeigt eine Meldung an, die angibt, ob die Pull-Anforderung zusammengeführt werden kann oder Konflikte gelöst werden müssen. Sobald alle Konflikte gelöst wurden und Sie Merge (Zusammenführen) auswählen, wird die Zusammenführung unter Anwendung der von Ihnen ausgewählten Merge-Strategie durchgeführt. FAST_FORWARD ist die Merge-Standardstrategie und die Standardoption für Git. Je nach dem Status des Codes in den Quell- und Ziel-Branches ist diese Strategie möglicherweise nicht verfügbar, dafür aber andere Optionen, wie z. B. Squashmerge oder 3-Wege-Merge.
- Mit der AWS CLI können Sie die Pull-Anforderung unter Verwendung der Merge-Strategie mit Vorlauf, Squash oder der Dreiwege-Merge-Strategie zusammenzuführen.

- Verwenden Sie auf Ihrem lokalen Computer den Befehl git merge, um den Quell-Branch mit dem Ziel-Branch zusammenzuführen, und übertragen Sie dann den zusammengeführten Code an den Ziel-Branch. Dieser Herangehensweise hat Nachteile, die Sie berücksichtigen müssen. Die Pull-Anforderung wird unabhängig davon zusammengeführt, ob die Anforderungen der Genehmigungsregeln für die Pull-Anforderung erfüllt wurden – diese Kontrollen werden umgangen. Durch das Zusammenführen und Pushen des Ziel-Branches wird die Pull-Anforderung auch automatisch geschlossen, wenn die Pull-Anforderung mithilfe der Merge-Strategie mit Vorlauf zusammengeführt wird. Ein Vorteil dieses Ansatzes ist, dass dergit mergeMit dem Befehl können Sie Mergeoptionen und -strategien wählen, die in der CodeCommit-Konsole nicht verfügbar sind. Weitere Informationen zu git merge und den Mergeoptionen finden Sie unter git-merge oder in Ihrer Git-Dokumentation.

CodeCommit schließt eine Pull-Anforderung automatisch, wenn der Quell- oder Ziel-Branch der Pull-Anforderung gelöscht wird.

Themen

- Zusammenführen einer Pull-Anforderung (-Konsole)
- Zusammenführen einer Pull-Anforderung (AWS CLI)

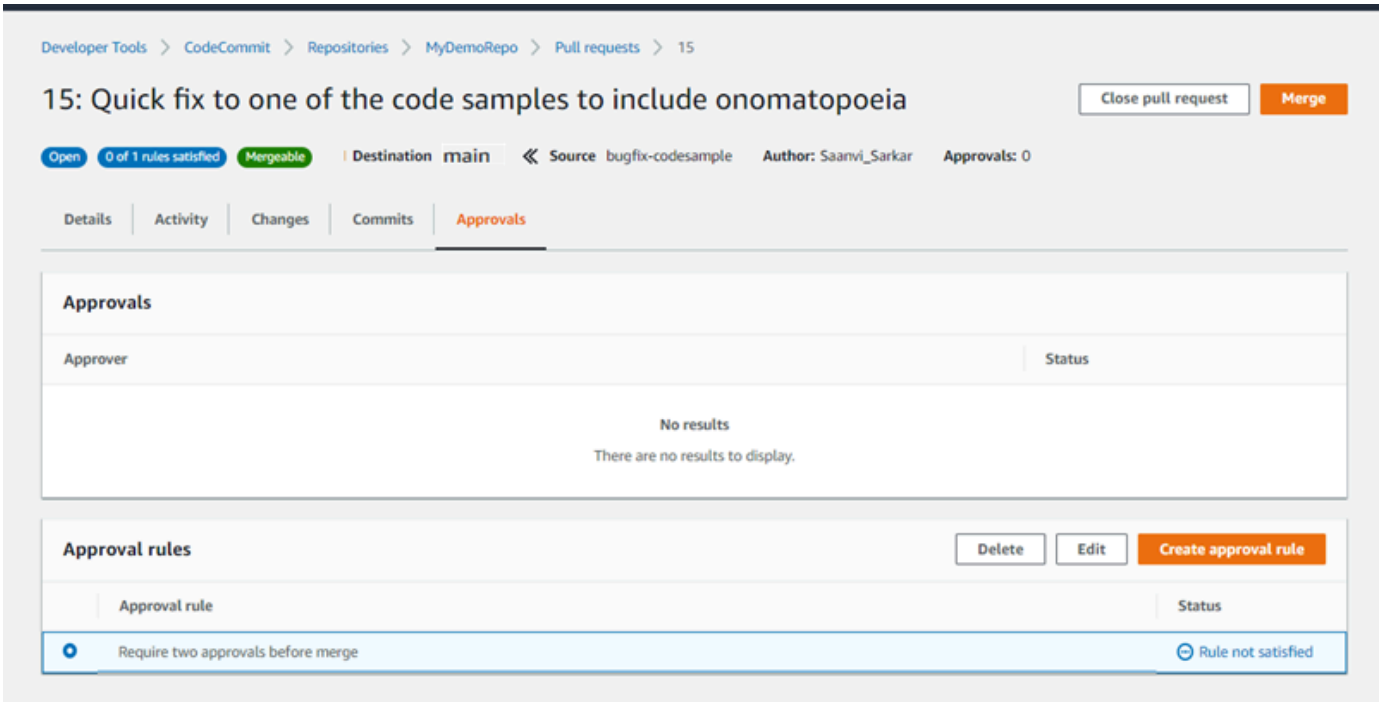
Zusammenführen einer Pull-Anforderung (-Konsole)

Sie können die CodeCommit-Konsole verwenden, um eine Pull-Anforderung in einem CodeCommit-Repository zusammenzuführen. Nachdem der Status einer Pull-Anforderung in Merged (Zusammengeführt) geändert wurde, wird sie nicht mehr in der Liste offener Pull-Anforderungen angezeigt. Eine zusammengeführte Pull-Anforderung wird als geschlossen kategorisiert. Ihr Status kann nicht wieder in Open (Geöffnet) geändert werden, aber Benutzer können die Änderungen weiterhin kommentieren und auf Kommentare antworten. Nachdem eine Pull-Anforderung zusammengeführt oder geschlossen wurde, können Sie sie nicht mehr genehmigen, eine erteilte Genehmigung für sie widerrufen oder die Genehmigungsregeln außer Kraft setzen, die der Pull-Anforderung zugewiesen wurden.

1. Öffnen Sie die CodeCommit-Konsole unter.https://console.aws.amazon.com/codesuite/codecommit/homeaus.
2. Wählen Sie im Bereich Repositories (Repositorys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.

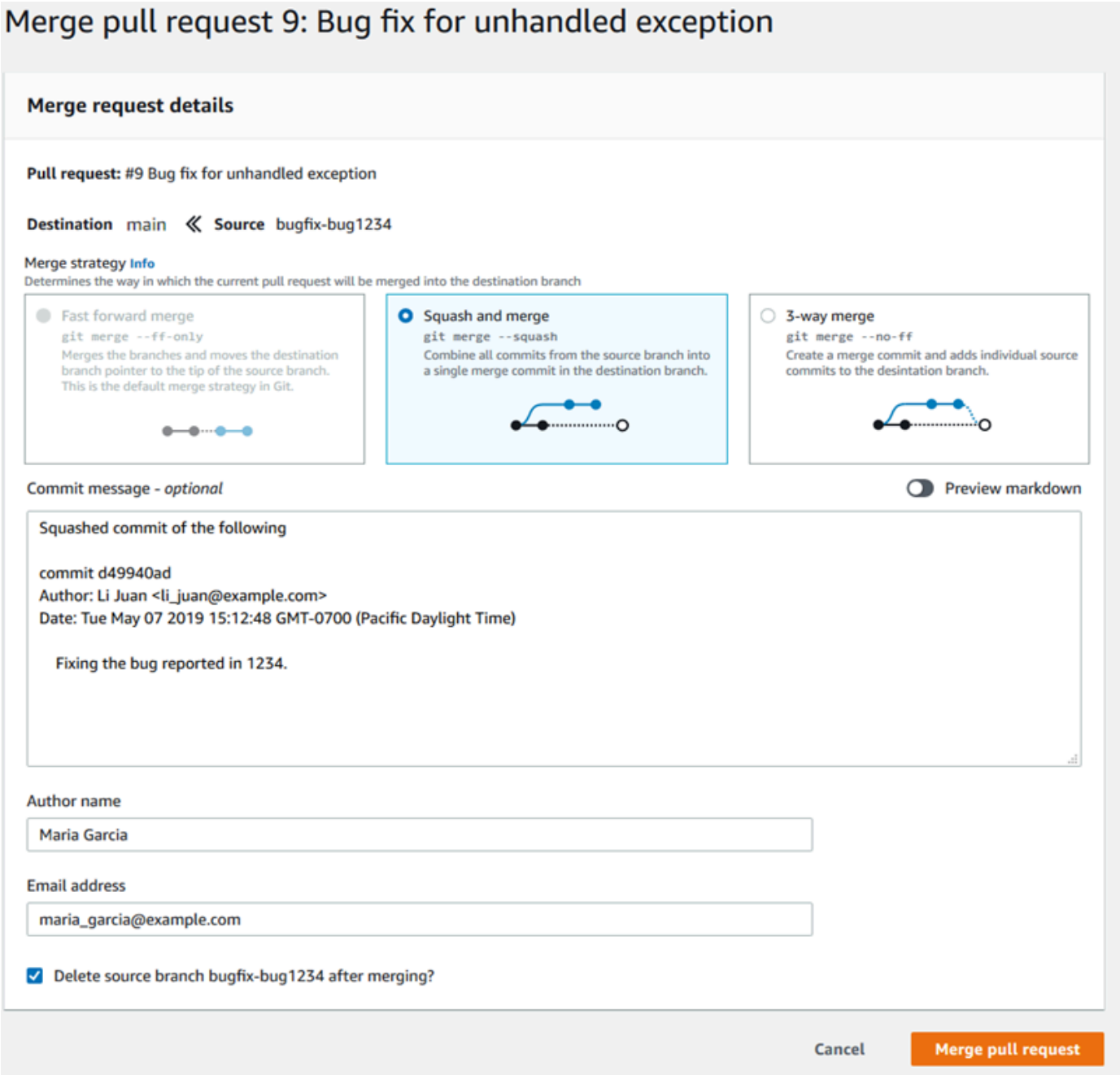
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie zusammenführen möchten.
5. Wählen Sie in der Pull-Anforderung die Option Approvals (Genehmigungen). Verifizieren Sie anhand der Liste der Genehmiger, dass die Bedingungen aller Genehmigungsregeln (falls vorhanden) erfüllt wurden. Sie können eine Pull-Anforderung nicht zusammenführen, wenn einzelne oder mehrere Genehmigungsregeln den Status Rule not satisfied (Regel nicht erfüllt) haben. Wenn kein Benutzer die Pull-Anforderung genehmigt hat, müssen Sie abwägen, ob sie zusammengeführt werden soll oder ob Sie auf Genehmigungen warten möchten.

> **Note**
>
> Wenn eine Genehmigungsregel für eine Pull-Anforderung erstellt wurde, können Sie sie bearbeiten oder löschen, um die Blockierung der Zusammenführung aufzuheben. Wenn die Genehmigungsregel mit einer Genehmigungsregelvorlage erstellt wurde, können Sie sie nicht bearbeiten oder löschen. Sie können sich nur entscheiden, die Anforderungen außer Kraft zu setzen. Weitere Informationen finden Sie unter Überschreiben von Genehmigungsregeln für eine Pull-Anforderung .



6. Wählen Sie Merge (Zusammenführen).

7. Wählen Sie in der Pull-Anforderung zwischen den verfügbaren Mergestrategien. Merge-Strategien, die nicht angewendet werden können, werden abgedunkelt dargestellt. Wenn keine Mergestrategien verfügbar sind, können Sie Konflikte auf Wunsch manuell in der CodeCommit-Konsole beheben oder Sie können sie mit Ihrem Git-Client lokal lösen. Weitere Informationen finden Sie unter Lösen von Konflikten in einer Pull-Anforderung in einemAWS CodeCommitEndlager .



- Bei einem Merge mit Vorlauf bewegt sich die Referenz für den Ziel-Branch vorwärts zum neuesten Commit des Quell-Branches. Dies ist das Standardverhalten von Git, soweit möglich.

Es wird kein Merge-Commit erstellt, der gesamte Commit-Verlauf vom Quell-Branch bleibt aber erhalten, als ob er im Ziel-Branch aufgetreten wäre. Zusammenführungen mit Vorlauf werden nicht als Branch-Zusammenführungen in der Commit-Ansicht des Verlaufs für den Ziel-Branch dargestellt, weil kein Merge-Commit erstellt wird. Die Spitze des Quell-Branches wird per Vorlauf an die Spitze des Ziel-Branches weitergeleitet.

- Bei einem Squash-Merge wird ein Commit mit den Änderungen im Quell-Branch erstellt und weist diesen einzelnen Squash-Commit dem Ziel-Branch zu. Standardmäßig enthält die Commit-Nachricht für diesen Squash-Commit alle Commit-Nachrichten der Änderungen im Quell-Branch. Individuelle Commit-Verläufe der Branch-Änderungen bleiben nicht erhalten. Dadurch lässt sich der Repository-Verlauf vereinfachen, während gleichzeitig eine grafische Darstellung der Zusammenführung in der Commit Visualizer-Ansicht der Ziel-Branch-Verlaufs beibehalten wird.
- Ein Dreiwege-Merge erstellt einen Merge-Commit für die Zusammenführung im Ziel-Branch, behält als Teil des Verlaufs des Ziel-Branches aber auch die einzelnen Commits bei, die im Quell-Branch vorgenommen wurden. Dies kann bei der Wahrung eines vollständigen Verlaufs der Änderungen an Ihrem Repository hilfreich sein.

8. Wenn Sie als Strategie den Squash-Merge oder den Dreiwege-Merge wählen, überprüfen Sie die automatisch generierte Commit-Nachricht und ändern Sie sie, wenn Sie die Informationen ändern möchten. Fügen Sie den Namen und die E-Mail-Adresse für den Commit-Verlauf hinzu.
9. (Optional) Deaktivieren Sie die Option, um den Quell-Branch im Rahmen der Zusammenführung zu löschen. Standardmäßig wird der Quell-Branch gelöscht, wenn eine Pull-Anforderung zusammengeführt wird.
10. Wählen Sie Merge pull request (Pull-Anforderung zusammenführen) um die Zusammenführung abzuschließen.

Zusammenführen einer Pull-Anforderung (AWS CLI)

Um zu verwendenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLIaus. Weitere Informationen finden Sie unter Befehlszeilenreferenz .

So verwenden Sie denAWS CLISo führen Sie Pull-Anforderungen in einem CodeCommit-Repository zusammen

1.

Um auszuwerten, ob eine Pull-Anforderung alle Genehmigungsregeln erfüllt und zusammengeführt werden kann, führen Sie den Befehl evaluate-pull-request-approval-rules aus, indem Sie Folgendes angeben:

- ID der Pull-Anforderung (mit der Option --pull-request-id).
- Revisions-ID der Pull-Anforderung (mit --revision-id option)). Sie können die aktuelle Revisions-ID für eine Pull-Anforderung mit dem Befehl get-pull-request abrufen.

Um beispielsweise den Status der Genehmigungsregeln für eine Pull-Anforderung mit der ID *27* und der Revisions-ID *9f29d167EXAMPLE* auszuwerten:

```
aws codecommit evaluate-pull-request-approval-rules --pull-request-id 27 --
revision-id 9f29d167EXAMPLE
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "evaluation": {
        "approved": false,
        "approvalRulesNotSatisfied": [
            "Require two approved approvers"
        ],
        "overridden": false,
        "approvalRulesSatisfied": []
    }
}
```

> **Note**
>
> Diese Ausgabe gibt an, dass eine Pull-Anforderung nicht zusammengeführt werden kann, weil die Anforderungen einer Genehmigungsregel nicht erfüllt wurden. Damit diese Pull-Anforderung zusammengeführt werden kann, können Sie voraussetzen, dass Genehmiger sie genehmigen und so die Bedingungen der Regel erfüllen. Abhängig von Ihren Berechtigungen und der Art der Regelerstellung können Sie die Regel möglicherweise auch bearbeiten, überschreiben oder löschen. Weitere Informationen finden Sie unter Überprüfen einer Pull-Anforderung, Überschreiben von

Genehmigungsregeln für eine Pull-Anforderung und Bearbeiten oder Löschen einer Genehmigungsregel für eine Pull-Anforderung.

2. Um eine Pull-Anforderung mittels der Mergestrategie mit Vorlauf zusammenzuführen und zu schließen, führen Sie den Befehl merge-pull-request-by-fast-forwardaus, wobei Sie Folgendes angeben:

   - ID der Pull-Anforderung (mit der Option --pull-request-id).
   - Die vollständige Commit-ID des obersten Commits des Quell-Branches (mit der Option --source-commit-id).
   - Der Name des Repositorys (mit der Option --repository-name).

   Wenn Sie beispielsweise eine Pull-Anforderung mit der ID zusammenführen und schließen möchten*47*und eine Quell-Commit-ID von*99132ab0Beispiel*in einem Repository mit dem Namen*MyDemoRepo*:

   ```
   aws codecommit merge-pull-request-by-fast-forward --pull-request-id 47 --source-
   commit-id 99132ab0EXAMPLE --repository-name MyDemoRepo
   ```

   Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "pullRequest": {
           "approvalRules": [
               {
                   "approvalRuleContent": "{\"Version\": \"2018-11-08\",\"Statements
   \": [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers
   \": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                   "approvalRuleId": "dd8b17fe-EXAMPLE",
                   "approvalRuleName": "I want one approver for this pull request",
                   "creationDate": 1571356106.936,
                   "lastModifiedDate": 571356106.936,
                   "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                   "ruleContentSha256": "4711b576EXAMPLE"
               }
           ],
           "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
           "clientRequestToken": "",
           "creationDate": 1508530823.142,
   ```

```
        "description": "Review the latest changes and updates to the global
 variables",
        "lastActivityDate": 1508887223.155,
        "pullRequestId": "47",
        "pullRequestStatus": "CLOSED",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": true,
                    "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables"
    }
}
```

3. Um eine Pull-Anforderung mittels der Squashmerge-Strategie zusammenzuführen und zu schließen, führen Sie den Befehl merge-pull-request-by-squashaus, wobei Sie Folgendes angeben:

   - ID der Pull-Anforderung (mit der Option --pull-request-id).
   - Die vollständige Commit-ID des obersten Commits des Quell-Branches (mit der Option --source-commit-id).
   - Der Name des Repositorys (mit der Option --repository-name).
   - Die Konflikt-Detailgenauigkeit, die Sie verwenden möchten (mit der Option --conflict-detail-level). Wenn dieser Parameter nicht angegeben wird, wird der Standardwert **`FILE_LEVEL`** verwendet.
   - Die Konfliktlösungsstrategie, die Sie verwenden möchten (mit der Option --conflict-resolution-strategy). Wenn dieser Parameter nicht angegeben wird, gilt der Standardwert `NONE`, und Konflikte müssen manuell behoben werden.
   - Die einzuschließende Commit-Nachricht (mit der Option --commit-message).
   - Der für den Commit zu verwendende Name (mit der Option --author-name).
   - Die für den Commit zu verwendende E-Mail-Adresse (mit der Option --email).

- Angabe, ob leere Ordner beibehalten werden sollen (mit der Option --keep-empty-folders).

Das folgende Beispiel führt beispielsweise eine Pull-Anforderung mit der ID zusammen und schließt sie dann*47*und eine Quell-Commit-ID von*99132ab0Beispiel*in einem Repository mit dem Namen*MyDemoRepo*aus. Es verwendet das Konfliktdetail `LINE_LEVEL` und die Konfliktlösungsstrategie `ACCEPT_SOURCE`:

```
aws codecommit merge-pull-request-by-squash --pull-request-id 47 --source-commit-
id 99132ab0EXAMPLE --repository-name MyDemoRepo --conflict-detail-level LINE_LEVEL
 --conflict-resolution-strategy ACCEPT_SOURCE --author-name "Jorge Souza" --email
 "jorge_souza@example.com" --commit-message "Merging pull request 47 by squash and
 accepting source in merge conflicts"
```

Ist der Befehl erfolgreich, erstellt er die gleiche Art von Ausgabe wie ein Merge mit Vorlauf, vergleichbar mit der folgenden Ausgabe:

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "originApprovalRuleTemplate": {
                    "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
                    "approvalRuleTemplateName": "2-approver-rule-for-main"
                },
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
        "clientRequestToken": "",
        "creationDate": 1508530823.142,
        "description": "Review the latest changes and updates to the global
 variables",
```

```
        "lastActivityDate": 1508887223.155,
        "pullRequestId": "47",
        "pullRequestStatus": "CLOSED",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": true,
                    "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables"
    }
}
```

4. Um eine Pull-Anforderung mittels der Dreiwege-Mergestrategie zusammenzuführen und zu schließen, führen Sie den Befehl merge-pull-request-by-three-way aus und geben Sie Folgendes an:

   - ID der Pull-Anforderung (mit der Option --pull-request-id).
   - Die vollständige Commit-ID des obersten Commits des Quell-Branches (mit der Option --source-commit-id).
   - Der Name des Repositorys (mit der Option --repository-name).
   - Die Konflikt-Detailgenauigkeit, die Sie verwenden möchten (mit der Option --conflict-detail-level). Wenn dieser Parameter nicht angegeben wird, wird der Standardwert **`FILE_LEVEL`** verwendet.
   - Die Konfliktlösungsstrategie, die Sie verwenden möchten (mit der Option --conflict-resolution-strategy). Wenn dieser Parameter nicht angegeben wird, gilt der Standardwert `NONE`, und Konflikte müssen manuell behoben werden.
   - Die einzuschließende Commit-Nachricht (mit der Option --commit-message).
   - Der für den Commit zu verwendende Name (mit der Option --author-name).
   - Die für den Commit zu verwendende E-Mail-Adresse (mit der Option --email).
   - Angabe, ob leere Ordner beibehalten werden sollen (mit der Option --keep-empty-folders).

Das folgende Beispiel führt beispielsweise eine Pull-Anforderung mit der ID zusammen und schließt sie dann*47*und eine Quell-Commit-ID von*99132ab0Beispiel*in einem Repository mit dem Namen*MyDemoRepo*aus. Es verwendet die Standardoptionen für Konfliktdetail und Konfliktlösungsstrategie:

```
aws codecommit merge-pull-request-by-three-way --pull-request-id 47 --source-
commit-id 99132ab0EXAMPLE --repository-name MyDemoRepo --author-name "Maria Garcia"
 --email "maria_garcia@example.com" --commit-message "Merging pull request 47 by
 three-way with default options"
```

Ist der Befehl erfolgreich, erstellt er die gleiche Art von Ausgabe wie ein Merge mit Vorlauf, vergleichbar mit der folgenden Ausgabe:

```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approver-rule-for-main",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "originApprovalRuleTemplate": {
                    "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
                    "approvalRuleTemplateName": "2-approver-rule-for-main"
                },
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
        "clientRequestToken": "",
        "creationDate": 1508530823.142,
        "description": "Review the latest changes and updates to the global
 variables",
        "lastActivityDate": 1508887223.155,
        "pullRequestId": "47",
```

```
        "pullRequestStatus": "CLOSED",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": true,
                    "mergedBy": "arn:aws:iam::123456789012:user/Mary_Major"
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables"
    }
}
```

Lösen von Konflikten in einer Pull-Anforderung in einemAWS CodeCommitEndlager

Wenn Ihre Pull-Anforderung Konflikte hat und nicht zusammengeführt werden kann, können Sie versuchen, die Konflikte auf eine von mehreren Arten zu lösen:

- Auf Ihrem lokalen Computer können Sie mit dem Befehl git diff nach Konflikten zwischen den beiden Branches suchen und Änderungen vornehmen, um sie zu lösen. Sie können auch ein anderes Tool oder ein anderes Programm verwenden, um sie beim Suchen und Beheben von Unterschieden zu unterstützen. Sobald Sie diese zu Ihrer Zufriedenheit gelöst haben, können Sie die Änderungen mit den gelösten Konflikten via Push auf Ihren Quell-Branch übertragen, wodurch die Pull-Anforderung aktualisiert wird. Weitere Informationen über git diff und git difftool finden Sie in der Git-Dokumentation.
- In der Konsole können Sie Resolve conflicts (Konflikte lösen) wählen. Daraufhin wird ein Texteditor geöffnet, der die Konflikte auf ähnliche Weise wie der Befehl git diff anzeigt. Sie können die Konflikte manuell in jeder Datei, in der sie enthalten sind, überprüfen, Änderungen vornehmen und die Pull-Anforderung dann mit Ihren Änderungen aktualisieren.
- In der AWS CLI können Sie mithilfe der AWS CLI Informationen über Zusammenführungskonflikte abrufen und einen unreferenzierten Mergecommit zum Testen einer Zusammenführung erstellen.

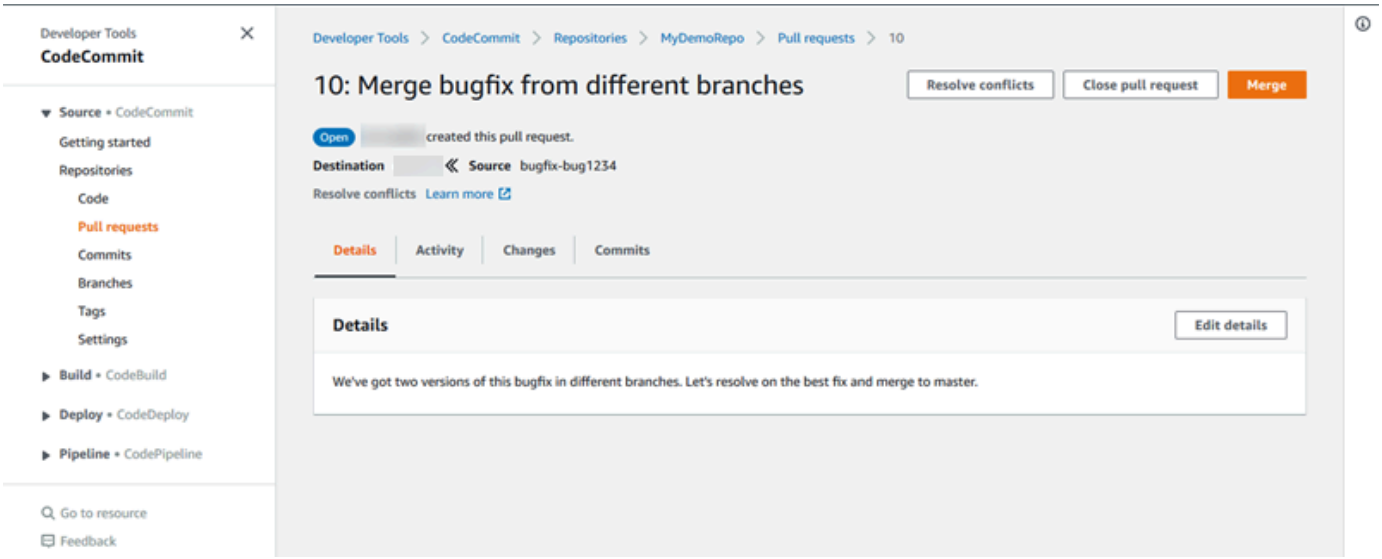
Themen

- Konflikte in einer Pull-Request (Konsole) lösen
- Lösen von Konflikten in einer Pull-Anforderung (AWS CLI)

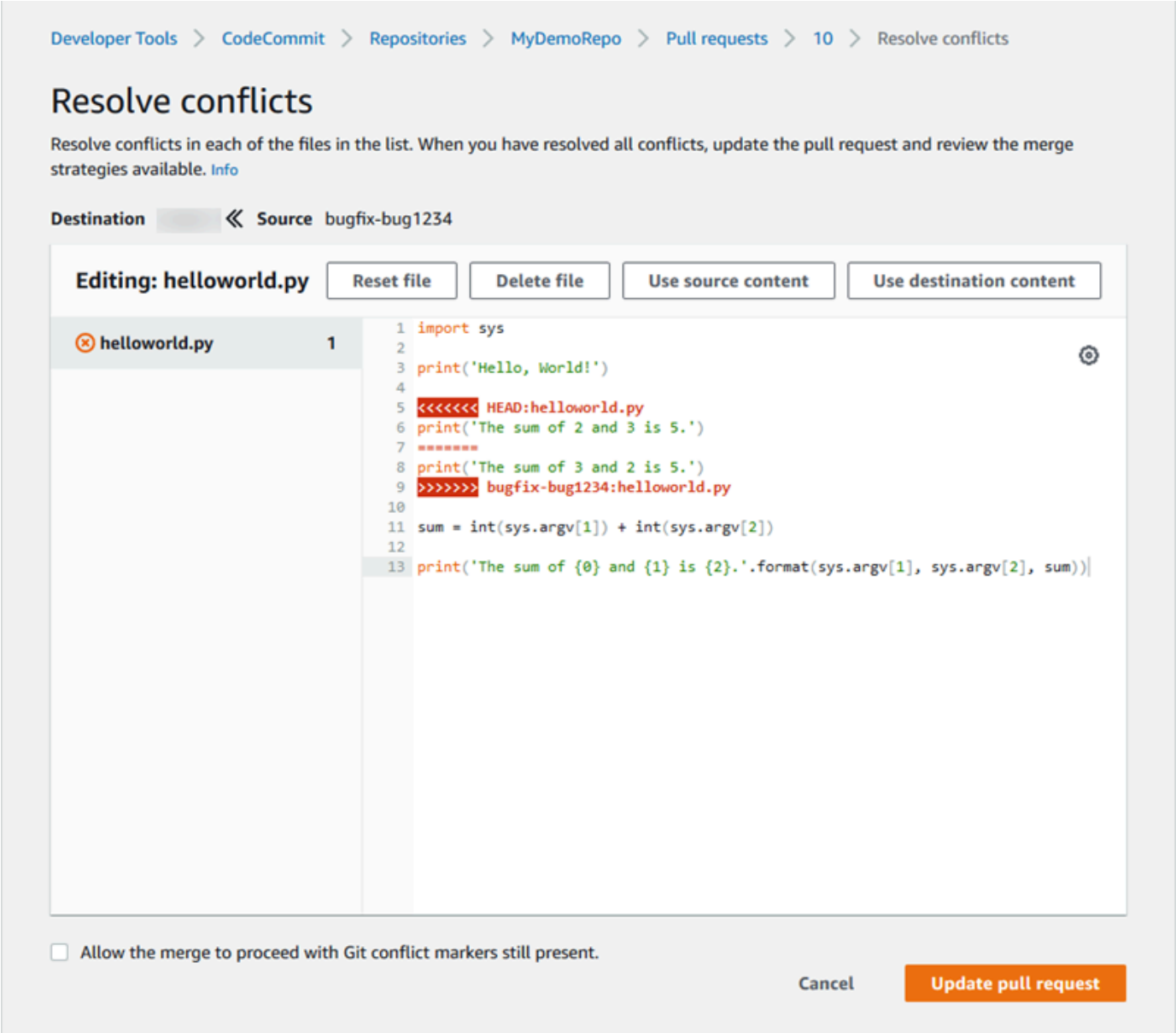
Konflikte in einer Pull-Request (Konsole) lösen

Sie können die CodeCommit-Konsole zum Lösen von Konflikten in einer Pull-Anforderung in einem CodeCommit-Repository verwenden.

1. Öffnen Sie die CodeCommit-Konsole unter.https://console.aws.amazon.com/codesuite/codecommit/homeaus.
2. Wählen Sie im Bereich Repositories (Repositorys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie zusammenführen möchten, die aber Konflikte enthält.
5. Wählen Sie in der Pull-Anforderung Resolve conflicts (Konflikte lösen). Diese Option wird nur angezeigt, wenn Konflikte vorhanden sind, die behoben werden müssen, bevor die Pull-Anforderung zusammengeführt werden kann.



6. In dem nun geöffneten Konfliktlösungsfenster werden die einzelnen Dateien mit Konflikten aufgeführt, die behoben werden müssen. Wählen Sie die einzelnen Dateien in der Liste aus, um die Konflikte zu überprüfen. Nehmen Sie alle erforderlichen Änderungen vor, bis alle Konflikte behoben wurden.



- Sie können wählen, ob Sie den Inhalt der Quelldatei oder der Zieldatei verwenden oder ob Sie bei einer anderen Datei als einer binären Datei ihren Inhalt manuell so bearbeiten möchten, dass sie nur die von Ihnen gewünschten Änderungen enthält. Standard-Git-Diff-Marker werden zum Aufzeigen der Konflikte zwischen den Ziel(HEAD)- und Quell-Branches in der Datei verwendet.
- Wenn es sich bei der Datei um eine binäre Datei oder ein Git-Untermodul handelt oder wenn ein Konflikt bei den Datei-/Ordnernamen vorliegt, müssen Sie entscheiden, ob die Quelldatei oder die Zieldatei zum Lösen der Konflikte verwendet werden soll. Es können keine Binärdateien in der CodeCommit-Konsole angezeigt oder bearbeitet werden.

- Wenn Dateimoduskonflikte vorliegen, können Sie die Option zum Lösen dieses Konfits anzeigen, indem Sie zwischen dem Dateimodus der Quelldatei und dem Dateimodus der Zieldatei wählen.
- Wenn Sie Ihre Änderungen an einer Datei verwerfen und deren konfligierten Zustand wiederherstellen möchten, wählen Sie Reset file (Datei zurücksetzen). Dies ermöglicht Ihnen, die Konflikte auf andere Weise zu lösen.

7. Wenn Sie mit Ihren Änderungen zufrieden sind, wählen Sie Update pull request (Pull-Anforderung aktualisieren).

> **Note**
>
> Sie müssen alle Konflikte in allen Dateien lösen, bevor Sie die Pull-Anforderung erfolgreich mit Ihren Änderungen aktualisieren können.

8. Die Pull-Anforderung wird mit Ihren Änderungen aktualisiert und kann zusammengeführt werden. Sie sehen den Zusammenführung-Seite. Sie können bestimmen, dass die Pull-Anforderung zu diesem Zeitpunkt zusammengeführt werden soll, oder zur Liste der Pull-Anforderungen zurückkehren.

Lösen von Konflikten in einer Pull-Anforderung (AWS CLI)

Um zu verwendenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLIaus. Weitere Informationen finden Sie unter Befehlszeilenreferenz .

Kein einzelnerAWS CLI-Befehl befähigt Sie, Konflikte in einer Pull-Anforderung zu lösen und diese Anforderung zusammenzuführen. Sie können mit einzelnen Befehlen jedoch Konflikte aufdecken und versuchen, sie zu lösen, und testen, ob eine Pull-Anforderung zusammengeführt werden kann. Sie können Folgendes verwenden:

- get-merge-options, um herauszufinden, welche Mergeoptionen für eine Zusammenführung zwischen zwei Commit-Spezifizierern verfügbar sind.
- get-merge-conflicts, um eine Liste von Dateien mit Konflikten beim Zusammenführen zwischen zwei Commit-Spezifizierern abzurufen.
- batch-describe-merge-conflicts, um Informationen über alle Zusammenführungskonflikte in Dateien in einer Zusammenführung zwischen zwei Commits mit einer bestimmten Strategie zu erhalten.

- describe-merge-conflicts, um detaillierte Informationen über Zusammenführungskonflikte für eine bestimmte Datei zwischen zwei Commits mit einer bestimmten Strategie zu erhalten.
- create-unreferenced-merge-commit, um das Ergebnis des Zusammenführens von zwei Commit-Spezifizierern mit einer bestimmten Strategie zu testen.

1. Um herauszufinden, welche Mergeoptionen für eine Zusammenführung zwischen zwei Commit-Spezifizierern verfügbar sind, führen Sie den Befehl get-merge-options aus und geben Sie Folgendes an:

   - Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option --source-commit-specifier).
   - Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option --destination-commit-specifier).
   - Der Name des Repositorys (mit der Option --repository-name).
   - (Optional) Eine Konfliktlösungstrategie, die verwendet werden soll (mit der Option --conflict-resolution-strategy).
   - (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option --conflict-detail-level).

   So bestimmen Sie beispielsweise die für die Zusammenführung eines Quell-Branch mit dem Namen verfügbaren Mergestrategien*bugfix-1234*mit einem Zielzweig namens*Haupt*in einem Repository namens*MyDemoRepo*:

   ```
   aws codecommit get-merge-options --source-commit-specifier bugfix-1234 --
   destination-commit-specifier main --repository-name MyDemoRepo
   ```

   Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "mergeOptions": [
           "FAST_FORWARD_MERGE",
           "SQUASH_MERGE",
           "THREE_WAY_MERGE"
       ],
       "sourceCommitId": "d49940adEXAMPLE",
       "destinationCommitId": "86958e0aEXAMPLE",
   ```

```
    "baseCommitId": "86958e0aEXAMPLE"
}
```

2. Um eine Liste der Dateien mit Zusammenführungskonflikten in einer Zusammenführung zwischen zwei Commit-Spezifizierern zu erhalten, führen Sie den Befehl get-merge-conflicts aus und geben Sie Folgendes an:

   - Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option --source-commit-specifier).
   - Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option --destination-commit-specifier).
   - Der Name des Repositorys (mit der Option --repository-name).
   - Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option --merge-option).
   - (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option --conflict-detail-level).
   - (Optional) Eine Konfliktlösungstrategie, die verwendet werden soll (mit der Option --conflict-resolution-strategy).
   - (Optional) Die maximale Anzahl von Dateien mit Konflikten, die zurückgegeben werden soll (mit der Option --max-conflict-files).

   So erhalten Sie eine Liste von Dateien mit Konflikten in einer Zusammenführung zwischen einem Quell-Branch mit dem Namen feature-randomizationfeature und einem Ziel-Branch mit dem Namen main mittels der 3-Wege-Mergestrategie in einem Repository mit dem Namen MyDemoRepo:

   ```
   aws codecommit get-merge-conflicts --source-commit-specifier feature-
   randomizationfeature --destination-commit-specifier main --merge-option
    THREE_WAY_MERGE --repository-name MyDemoRepo
   ```

   Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "mergeable": false,
       "destinationCommitId": "86958e0aEXAMPLE",
       "sourceCommitId": "6ccd57fdEXAMPLE",
       "baseCommitId": "767b6958EXAMPLE",
       "conflictMetadataList": [
   ```

```
        {
            "filePath": "readme.md",
            "fileSizes": {
                "source": 139,
                "destination": 230,
                "base": 85
            },
            "fileModes": {
                "source": "NORMAL",
                "destination": "NORMAL",
                "base": "NORMAL"
            },
            "objectTypes": {
                "source": "FILE",
                "destination": "FILE",
                "base": "FILE"
            },
            "numberOfConflicts": 1,
            "isBinaryFile": {
                "source": false,
                "destination": false,
                "base": false
            },
            "contentConflict": true,
            "fileModeConflict": false,
            "objectTypeConflict": false,
            "mergeOperations": {
                "source": "M",
                "destination": "M"
            }
        }
    ]
}
```

3. Um Informationen über Zusammenführungskonflikte in allen Dateien und einem Teilsatz von Dateien in einer Zusammenführung zwischen zwei Commit-Spezifizierern zu erhalten, führen Sie den Befehl batch-describe-merge-conflicts aus und geben Sie Folgendes an:

   - Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option --source-commit-specifier).
   - Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option --destination-commit-specifier).

- Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option --merge-option).
- Der Name des Repositorys (mit der Option --repository-name).
- (Optional) Eine Konfliktlösungstrategie, die verwendet werden soll (mit der Option --conflict-resolution-strategy).
- (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option --conflict-detail-level).
- (Optional) Die maximale Anzahl von Zusammenführungsstücken, die zurückgegeben werden soll (mit der Option --max-merge-hunks).
- (Optional) Die maximale Anzahl von Dateien mit Konflikten, die zurückgegeben werden soll (mit der Option --max-conflict-files).
- (Optional) Der Pfad der Ziel-Dateien, der zur Beschreibung der Konflikte verwendet werden soll (mit der Option --file-paths).

So bestimmen Sie beispielsweise die Zusammenführungskonflikte für die Zusammenführung eines Quell-Branch mit dem Namen*Funktion randomizationfeature*mit einem Zielzweig namens*Haupt*Verwendung von*THREE_WAY_MERGE*Strategie in einem Repository namens*MyDemoRepo*:

```
aws codecommit batch-describe-merge-conflicts --source-commit-specifier feature-
randomizationfeature --destination-commit-specifier main --merge-
option THREE_WAY_MERGE --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "conflicts": [
        {
            "conflictMetadata": {
                "filePath": "readme.md",
                "fileSizes": {
                    "source": 139,
                    "destination": 230,
                    "base": 85
                },
                "fileModes": {
                    "source": "NORMAL",
                    "destination": "NORMAL",
```

```
                    "base": "NORMAL"
                },
                "objectTypes": {
                    "source": "FILE",
                    "destination": "FILE",
                    "base": "FILE"
                },
                "numberOfConflicts": 1,
                "isBinaryFile": {
                    "source": false,
                    "destination": false,
                    "base": false
                },
                "contentConflict": true,
                "fileModeConflict": false,
                "objectTypeConflict": false,
                "mergeOperations": {
                    "source": "M",
                    "destination": "M"
                }
            },
            "mergeHunks": [
                {
                    "isConflict": true,
                    "source": {
                        "startLine": 0,
                        "endLine": 3,
                        "hunkContent": "VGhpcyBpEXAMPLE=="
                    },
                    "destination": {
                        "startLine": 0,
                        "endLine": 1,
                        "hunkContent": "VXNlIHRoEXAMPLE="
                    }
                }
            ]
        }
    ],
    "errors": [],
    "destinationCommitId": "86958e0aEXAMPLE",
    "sourceCommitId": "6ccd57fdEXAMPLE",
    "baseCommitId": "767b6958EXAMPLE"
}
```

4. Um detaillierte Informationen über Zusammenführungskonflikte für eine bestimmte Datei in einer Zusammenführung zwischen zwei Commit-Spezifizierern zu erhalten, führen Sie den Befehl describe-merge-conflicts aus und geben Sie Folgendes an:

   - Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option --source-commit-specifier).
   - Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option --destination-commit-specifier).
   - Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option --merge-option).
   - Der Pfad der Zieldatei, die zur Beschreibung der Konflikte verwendet werden soll (mit der Option --file-path).
   - Der Name des Repositorys (mit der Option --repository-name).
   - (Optional) Eine Konfliktlösungstrategie, die verwendet werden soll (mit der Option --conflict-resolution-strategy).
   - (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option --conflict-detail-level).
   - (Optional) Die maximale Anzahl von Zusammenführungsstücken, die zurückgegeben werden soll (mit der Option --max-merge-hunks).
   - (Optional) Die maximale Anzahl von Dateien mit Konflikten, die zurückgegeben werden soll (mit der Option --max-conflict-files).

   Um beispielsweise die Zusammenführungskonflikte für eine Datei mit dem Namen zu ermitteln*readme.md*in einem Quellzweig namens*Funktion randomizationfeature*mit einem Zielzweig namens*Haupt*Verwendung von*THREE_WAY_MERGE*Strategie in einem Repository namens*MyDemoRepo*:

   ```
   aws codecommit describe-merge-conflicts --source-commit-specifier feature-randomizationfeature --destination-commit-specifier main --merge-option THREE_WAY_MERGE --file-path readme.md --repository-name MyDemoRepo
   ```

   Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "conflictMetadata": {
           "filePath": "readme.md",
   ```

```
        "fileSizes": {
            "source": 139,
            "destination": 230,
            "base": 85
        },
        "fileModes": {
            "source": "NORMAL",
            "destination": "NORMAL",
            "base": "NORMAL"
        },
        "objectTypes": {
            "source": "FILE",
            "destination": "FILE",
            "base": "FILE"
        },
        "numberOfConflicts": 1,
        "isBinaryFile": {
            "source": false,
            "destination": false,
            "base": false
        },
        "contentConflict": true,
        "fileModeConflict": false,
        "objectTypeConflict": false,
        "mergeOperations": {
            "source": "M",
            "destination": "M"
        }
    },
    "mergeHunks": [
        {
            "isConflict": true,
            "source": {
                "startLine": 0,
                "endLine": 3,
                "hunkContent": "VGhpcyBpEXAMPLE=="
            },
            "destination": {
                "startLine": 0,
                "endLine": 1,
                "hunkContent": "VXNlIHRoEXAMPLE="
            }
        }
    ],
```

```
    "destinationCommitId": "86958e0aEXAMPLE",
    "sourceCommitId": "6ccd57fdEXAMPLE",
    "baseCommitId": "767b69580EXAMPLE"
}
```

5. Um einen uneferenzierten Commit zu erstellen, der das Ergebnis des Zusammenführens von zwei Commit-Spezifizierern repräsentiert, führen Sie den Befehl create-unreferenced-merge-commit aus und geben Sie Folgendes an:

   - Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option --source-commit-specifier).
   - Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option --destination-commit-specifier).
   - Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option --merge-option).
   - Der Name des Repositorys (mit der Option --repository-name).
   - (Optional) Eine Konfliktlösungstrategie, die verwendet werden soll (mit der Option --conflict-resolution-strategy).
   - (Optional) Die gewünschte Detailgenauigkeit aller Konflikte (mit der Option --conflict-detail-level).
   - (Optional) Die einzuschließende Commit-Nachricht (mit der Option --commit-message).
   - (Optional) Der für den Commit zu verwendende Name (mit der Option --name).
   - (Optional) Die für den Commit zu verwendende E-Mail-Adresse (mit der Option --email).
   - (Optional) Angabe, ob leere Ordner beibehalten werden sollen (mit der Option --keep-empty-folders).

   So bestimmen Sie beispielsweise die Zusammenführungskonflikte für die Zusammenführung eines Quell-Branch mit dem Namen*bugfix-1234*mit einem Zielzweig namens*Haupt*Verwenden der ACCEPT_SOURCE-Strategie in einem Repository namens*MyDemoRepo*:

   ```
   aws codecommit create-unreferenced-merge-commit --source-commit-
   specifier bugfix-1234 --destination-commit-specifier main --merge-option
    THREE_WAY_MERGE --repository-name MyDemoRepo --name "Maria Garcia" --email
    "maria_garcia@example.com" --commit-message "Testing the results of this merge."
   ```

   Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "commitId": "4f178133EXAMPLE",
    "treeId": "389765daEXAMPLE"
}
```

Schließen einer Pull-Anforderung in einemAWS CodeCommitEndlager

Wenn Sie eine Pull-Anforderung schließen möchten, ohne den Code zusammenzuführen, stehen Ihnen dazu mehrere Optionen zur Verfügung:

- In der Konsole können Sie eine Pull-Anforderung schließen, ohne den Code zusammenzuführen. Dies ist sinnvoll, wenn Sie Branches manuell mithilfe des Befehls git merge zusammenführen möchten oder wenn der Code im Quell-Branch der Pull-Anforderung nicht der Code ist, der mit dem Ziel-Branch zusammengeführt werden soll.
- Sie können den in der Pull-Anforderung angegebenen Quellzweig löschen. CodeCommit schließt eine Pull-Anforderung automatisch, wenn der Quell- oder Ziel-Branch der Pull-Anforderung gelöscht wird.
- In der AWS CLI können Sie den Status einer Pull-Anforderung von `OPEN` auf `CLOSED` aktualisieren. Damit wird die Pull-Anforderung geschlossen, ohne den Code zusammenzuführen.

Themen

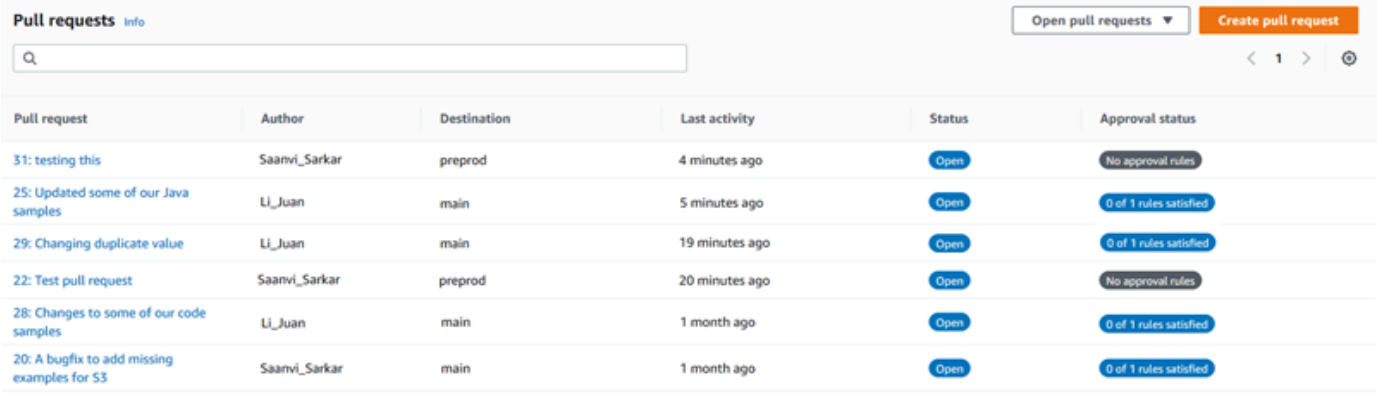
- Schließen einer Pull-Anforderung (-Konsole)
- Schließen einer Pull-Anforderung (AWS CLI)

Schließen einer Pull-Anforderung (-Konsole)

Sie können die CodeCommit-Konsole verwenden, um eine Pull-Anforderung in einem CodeCommit--Repository zu schließen. Nachdem der Status einer Pull-Anforderung in Closed geändert wurde, kann er nicht wieder zu Open wechseln. Die Benutzer können jedoch weiterhin Änderungen kommentieren und auf Kommentare antworten.

1. Öffnen Sie die CodeCommit-Konsole unter.https://console.aws.amazon.com/codesuite/codecommit/homeaus.

2. Wählen Sie im Bereich Repositories (Repositorys) den Namen des Repositorys aus.
3. Wählen Sie im Navigationsbereich Pull requests (Pull-Anforderungen) aus.
4. Standardmäßig wird eine Liste aller offenen Pull-Anforderungen angezeigt. Wählen Sie die offene Pull-Anforderung aus, die Sie schließen möchten.



5. Wählen Sie in der Pull-Anforderung Close pull request (Pull-Anforderung schließen). Diese Option schließt die Pull-Anforderung ohne zu versuchen, den Quell-Branch mit dem Ziel-Branch zusammenzuführen. Diese Option bietet keine Möglichkeit, den Quell-Branch beim Schließen der Pull-Anforderung zu löschen. Sie können diesen Vorgang aber selbst ausführen, sobald die Anforderung geschlossen ist.

Schließen einer Pull-Anforderung (AWS CLI)

Um zu verwendenAWS CLIBefehle mit CodeCommit, installieren Sie dasAWS CLIaus. Weitere Informationen finden Sie unter Befehlszeilenreferenz .

So verwenden Sie denAWS CLISo schließen Sie Pull-Anforderungen in einem CodeCommit--Repository

- Um den Status einer Pull-Anforderung in einem Repository von `OPEN` zu `CLOSED` zu ändern, führen Sie den Befehl update-pull-request-status aus, wobei Sie Folgendes angeben:
  - ID der Pull-Anforderung (mit der Option --pull-request-id).
  - Der Status der Pull-Anforderung (mit der Option --pull-request-status).

  Wenn Sie beispielsweise den Status einer Pull-Anforderung mit der ID aktualisieren möchten*42*zu einem Status von*GESCHLOSSEN*in einem CodeCommit-Repository mit dem Namen`MyDemoRepo`:

```
aws codecommit update-pull-request-status --pull-request-id 42 --pull-request-
status CLOSED
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

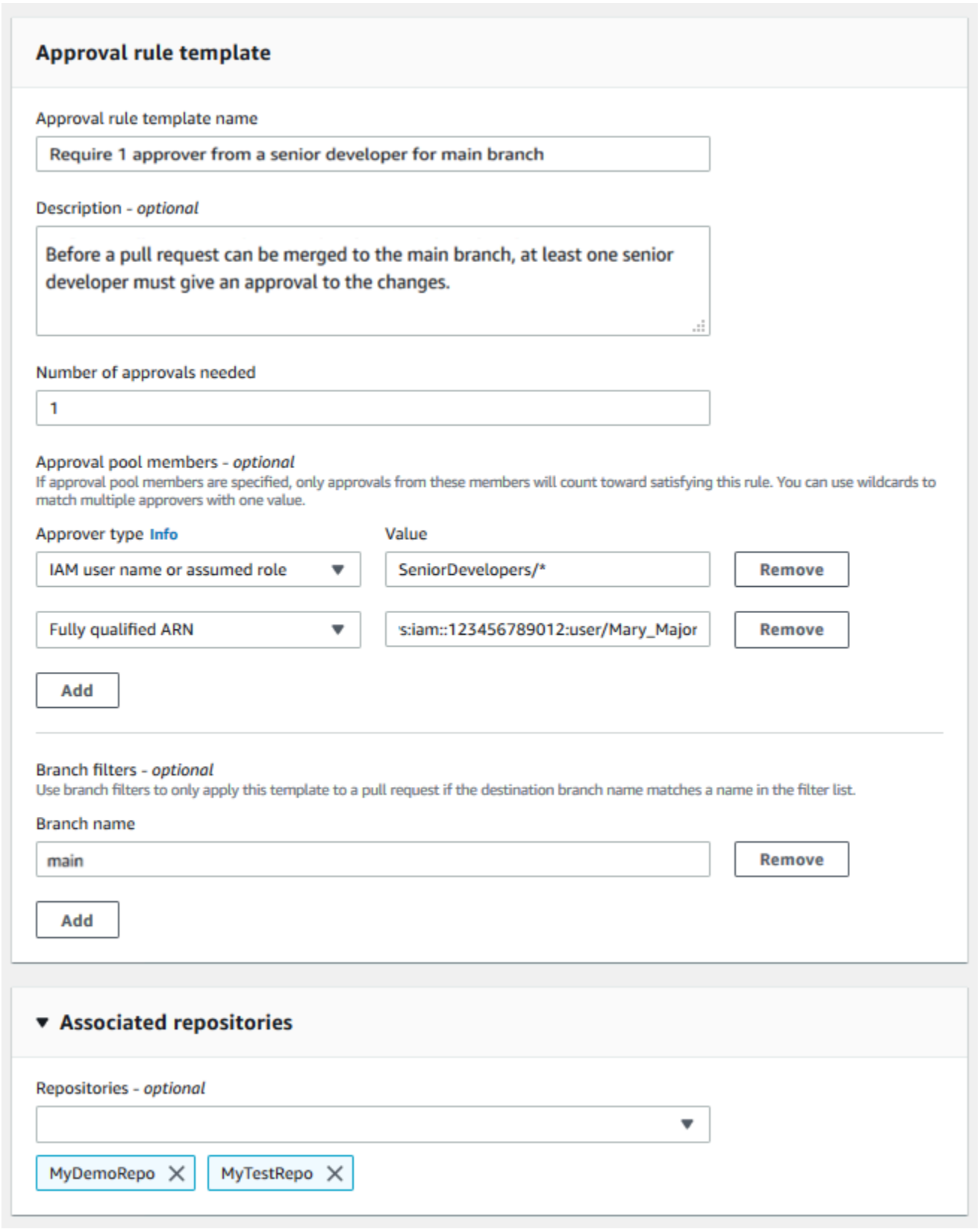
```
{
    "pullRequest": {
        "approvalRules": [
            {
                "approvalRuleContent": "{\"Version\": \"2018-11-08\",\"Statements
\": [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers
\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
                "approvalRuleId": "dd8b17fe-EXAMPLE",
                "approvalRuleName": "2-approvers-needed-for-this-change",
                "creationDate": 1571356106.936,
                "lastModifiedDate": 571356106.936,
                "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
                "ruleContentSha256": "4711b576EXAMPLE"
            }
        ],
        "authorArn": "arn:aws:iam::123456789012:user/Li_Juan",
        "clientRequestToken": "",
        "creationDate": 1508530823.165,
        "description": "Updated the pull request to remove unused global
 variable.",
        "lastActivityDate": 1508372423.12,
        "pullRequestId": "47",
        "pullRequestStatus": "CLOSED",
        "pullRequestTargets": [
            {
                "destinationCommit": "9f31c968EXAMPLE",
                "destinationReference": "refs/heads/main",
                "mergeMetadata": {
                    "isMerged": false,
                },
                "repositoryName": "MyDemoRepo",
                "sourceCommit": "99132ab0EXAMPLE",
                "sourceReference": "refs/heads/variables-branch"
            }
        ],
        "title": "Consolidation of global variables"
    }
```

```
}
```

Arbeiten mit Genehmigungsregelvorlagen

Sie können Genehmigungsregeln für Pull-Anforderungen erstellen. Verwenden Sie Vorlagen für Genehmigungsregeln, um Genehmigungsregeln automatisch auf einige oder alle in Repositorys erstellten Pull-Requests anzuwenden. Genehmigungsregelvorlagen erleichtern das Anpassen von Entwicklungs-Workflows über Repositorys hinweg, sodass für verschiedene Branches ausreichende Genehmigungen und Kontrollen vorgesehen sind. Sie können verschiedene Regeln für Produktions- und Entwicklungs-Branches definieren. Diese Regeln werden angewendet, wenn eine Pull-Anforderung erstellt wird, die den Regelbedingungen entspricht. Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unterBerechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln undVon AWS verwaltete Richtlinien für CodeCommit.

Sie können eine Vorlage für Genehmigungsregeln mit einem oder mehreren Repositorys verknüpfen, inAWS-Region denen sie erstellt wurden. Wenn eine Vorlage einem Repository zugeordnet ist, erstellt sie beim Erstellen von Pull-Anforderungen automatisch Genehmigungsregeln für Pull-Anforderungen in diesem Repository. Genau wie eine Genehmigungsregel definiert eine Genehmigungsregelvorlage eine Genehmigungsregelstruktur, einschließlich der Anzahl der erforderlichen Genehmigungen und eines optionalen Pools der Benutzer, von denen die Genehmigungen stammen müssen. Im Unterschied zu einer Genehmigungsregel können Sie außerdem Zielverweise (Branches) definieren, die auch als Branch-Filter bezeichnet werden. Wenn Sie Zielverweise definieren, werden nur für Pull-Anforderungen, deren Branch-Zielnamen den angegebenen Branch-Namen (Zielverweise) in der Vorlage entsprechen, Regeln erstellt. Wenn Sie z. B. **`refs/heads/main`** als Zielverweis angeben, wird die in der Vorlage definierte Genehmigungsregel nur auf Pull-Anforderungen angewendet, wenn der Ziel-Branch `main` ist.


Approval rule template
Approval rule template name
Require 1 approver from a senior developer for main branch
Description - optional
Before a pull request can be merged to the main branch, at least one senior developer must give an approval to the changes.
Number of approvals needed
1
Approval pool members - optional
If approval pool members are specified, only approvals from these members will count toward satisfying this rule. You can use wildcards to match multiple approvers with one value.
Approver type Info
Value
IAM user name or assumed role
SeniorDevelopers/*
Remove
Fully qualified ARN
s:iam::123456789012:user/Mary_Major
Remove
Add
Branch filters - optional
Use branch filters to only apply this template to a pull request if the destination branch name matches a name in the filter list.
Branch name
main
Remove
Add
Associated repositories
Repositories - optional
MyDemoRepo
MyTestRepo

Themen

- Erstellen einer Genehmigungsregelvorlage
- Eine Genehmigungsregelvorlage einem Repository zuordnen.
- Verwalten von Genehmigungsregelvorlagen
- So trennen Sie eine Genehmigungsregelvorlage.
- Eine Genehmigungsregelvorlage löschen

Erstellen einer Genehmigungsregelvorlage

Sie können einzelne oder mehrere Genehmigungsregelvorlagen erstellen, um Entwicklungs-Workflows über Repositorys hinweg anzupassen. Durch das Erstellen mehrerer Vorlagen können Sie die automatische Erstellung von Genehmigungsregeln so gestalten, dass für unterschiedliche Branches die jeweils erforderlichen Genehmigungen vorausgesetzt und Kontrollen bereitgestellt werden. Sie können beispielsweise unterschiedliche Vorlagen für Produktions- und Entwicklungs-Branches erstellen und diese Vorlagen einzelnen oder mehreren Repositorys zuweisen. Wenn Benutzer Pull-Anforderungen in diesen Repositorys erstellen, wird die Anforderung anhand dieser Vorlagen ausgewertet. Wenn die Anforderung den Bedingungen in den zugewiesenen Vorlagen entspricht, werden Genehmigungsregeln für die Pull-Anforderung erstellt.

Sie können die Konsole oder die AWS CLI verwenden, um Genehmigungsregelvorlagen zu erstellen. Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unterBerechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln undVon AWS verwaltete Richtlinien für CodeCommit.

Themen

- So erstellen Sie eine Genehmigungsregelvorlage (Konsole)
- So erstellen Sie eine Genehmigungsregelvorlage (AWS CLI)

So erstellen Sie eine Genehmigungsregelvorlage (Konsole)

Genehmigungsregelvorlagen sind standardmäßig keinem Repository zugeordnet. Sie können eine Zuordnung zwischen einer Vorlage und einzelnen oder mehreren Repositorys herstellen, wenn Sie die Vorlage erstellen. Sie können die Zuordnungen aber auch zu einem späteren Zeitpunkt vornehmen.

So erstellen Sie eine Genehmigungsregelvorlage (Konsole)

1. Öffnen Sie dieCodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie Approval rule templates (Genehmigungsregelvorlagen) und dann Create template (Vorlage erstellen) aus.
3. Geben Sie in das Feld Approval rule template name (Name der Genehmigungsregelvorlage) einen aussagekräftigen Namen an, der den Zweck beschreibt. Wenn Sie beispielsweise möchten, dass eine Person aus einer Gruppe leitender Entwickler eine Pull-Anforderung genehmigen muss, bevor sie zusammengeführt werden kann, können Sie die Regel beispielsweise **`Require 1 approver from a senior developer`** nennen.
4. (Optional) Geben Sie in das Feld Description (Beschreibung) eine Beschreibung des Zwecks dieser Vorlage ein. Dies erleichtert anderen Benutzern die Entscheidung, ob diese Vorlage für ihre Repositorys geeignet ist.
5. Geben Sie in das Feld Number of approvals needed (Anzahl erforderlicher Genehmigungen) die gewünschte Anzahl ein. Der Standardwert ist 1.
6. (Optional) Wenn Sie voraussetzen möchten, dass die Genehmigungen für eine Pull-Anforderung von einer bestimmten Gruppe von Benutzern stammen, wählen Sie unter Approval rule members (Genehmigungsregelmitglieder) die Option Add (Hinzufügen) aus. Wählen Sie unter Approver type (Genehmigertyp) eine der folgenden Optionen aus:

   - IAM-Benutzername oder angenommene Rolle: Diese Option füllt die Amazon Web Services Services-Konto-ID für das Konto, mit dem Sie sich angemeldet haben, vorab aus und erfordert nur einen Namen. Es kann sowohl für IAM-Benutzer als auch für Verbundzugriffsbenutzer verwendet werden, deren Name mit dem angegebenen Namen übereinstimmt. Dies ist eine sehr leistungsfähige Option, die hohe Flexibilität bietet. Wenn Sie beispielsweise diese Option wählen und mit dem Amazon Web Services Services-Konto 123456789012 angemeldet sind und angeben**`Mary_Major`**, werden alle folgenden Genehmigungen als Genehmigungen dieses Benutzers gezählt:
     - Ein IAM-Benutzer im Konto (`arn:aws:iam::123456789012:user/Mary_Major`)
     - Ein Verbundbenutzer in IAM mit dem Namen Mary_Major (`arn:aws:sts::123456789012:federated-user/Mary_Major`)

     Diese Option erkennt keine aktive Sitzung eines Benutzers, der die Rolle **`CodeCommitReview`** angenommen hat und den Rollensitzungsnamen Mary_Major (`arn:aws:sts::123456789012:assumed-role/CodeCommitReview/Mary_Major`)

verwendet, sofern Sie kein Platzhalterzeichen angeben (`*Mary_Major`). Sie können den Rollennamen auch explizit angeben (`CodeCommitReview/Mary_Major`).

- Vollqualifizierter ARN: Mit dieser Option können Sie den vollqualifizierten Amazon Resource Name (ARN) des IAM-Benutzers oder der IAM-Rolle angeben. Diese Option unterstützt auch angenommene Rollen, die von anderen AWS-Services wie AWS Lambda und AWS CodeBuild verwendet werden. Bei angenommenen Rollen sollte das ARN-Format bei Rollen „`arn:aws:sts::`*`AccountID`*`:assumed-role/`*`RoleName`*“ und bei Funktionen „`arn:aws:sts::`*`AccountID`*`:assumed-role/`*`FunctionName`*“ sein.

Wenn Sie den IAM-Benutzernamen oder die übernommene Rolle als Genehmigungstyp ausgewählt haben, geben Sie im Feld Wert den Namen des IAM-Benutzers oder der IAM-Rolle oder den vollständig qualifizierten ARN des Benutzers oder der Rolle ein. Wählen Sie erneut Add (Hinzufügen) aus, um weitere Benutzer oder Rollen hinzuzufügen, bis Sie alle Benutzer oder Rollen hinzugefügt haben, deren Genehmigungen auf die Anzahl der erforderlichen Genehmigungen angerechnet werden sollen.

Für beide Genehmigertypen können Platzhalterzeichen (*) in den Werten verwendet werden. Wenn Sie beispielsweise die Option IAM-Benutzername oder übernommene Rolle wählen und angeben**`CodeCommitReview/*`**,**`CodeCommitReview`** werden alle Benutzer, die die Rolle von übernehmen, im Genehmigungspool gezählt. Die individuellen Rollensitzungsnamen zählen zum Erreichen der erforderlichen Anzahl von Genehmigern. Auf diese Weise zählen sowohl Mary_Major als auch Li_Juan als Genehmigungen, wenn sie angemeldet sind und die Rolle `CodeCommitReview` annehmen. Weitere Informationen zu IAM-ARNs, Platzhaltern und IDs finden Sie unter IAM-IDs.

> **Note**
>
> Genehmigungsregeln unterstützen keine kontoübergreifenden Genehmigungen.

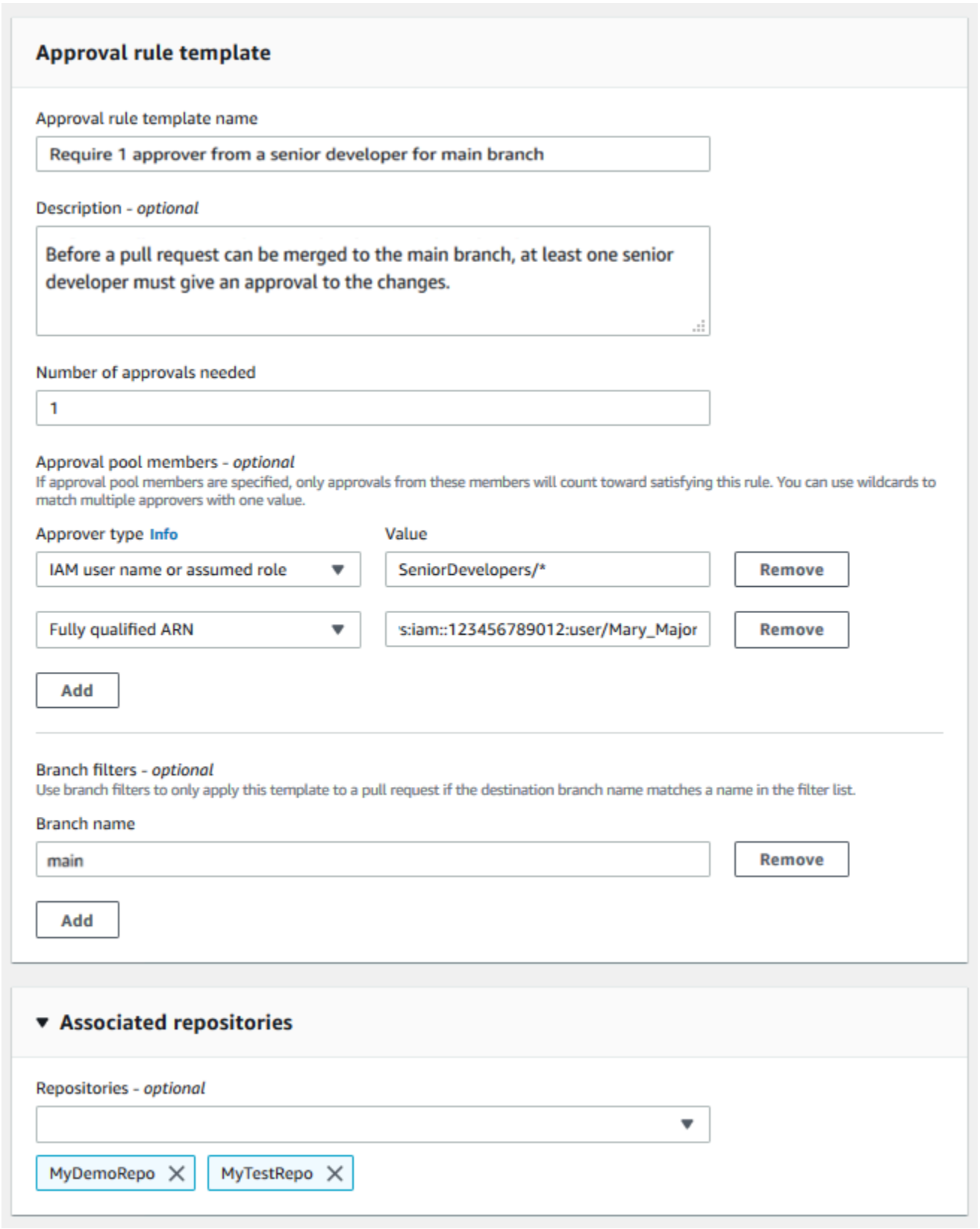
7. (Optional) Geben Sie unter Branch filters (Branch-Filter) die Branch-Zielnamen ein, die zum Filtern der Erstellung von Genehmigungsregeln verwendet werden sollen. Wenn Sie beispielsweise *`main`* angeben, wird eine Genehmigungsregel für Pull-Requests in zugehörigen Repositorys nur erstellt, wenn der Ziel-Branch für den Pull-Request ein Branch mit dem Namen *`main`* ist. Sie können Platzhalterzeichen (*) in Branch-Namen verwenden, um Genehmigungsregeln allen Branch-Namen zuzuweisen, die den Namen mit Platzhaltern entsprechen. Sie dürfen Platzhalterzeichen jedoch nicht am Anfang eines Branch-Namens

verwenden. Sie können bis zu 100 Branch-Namen angeben. Wenn Sie keine Filter angeben, gilt die Vorlage für alle Branches in einem zugeordneten Repository.

8. (Optional) Wählen Sie unter Verknüpfte Repositorys in der Liste Repositorys die Repositorys ausAWS-Region, die Sie dieser Genehmigungsregel zuordnen möchten.

> **Note**
>
> Nach dem Erstellen der Vorlage können Sie Repositorys zuordnen. Weitere Informationen finden Sie unter Eine Genehmigungsregelvorlage einem Repository zuordnen..

9. Wählen Sie Create (Erstellen) aus.


Approval rule template
Approval rule template name
Require 1 approver from a senior developer for main branch
Description - optional
Before a pull request can be merged to the main branch, at least one senior developer must give an approval to the changes.
Number of approvals needed
1
Approval pool members - optional
If approval pool members are specified, only approvals from these members will count toward satisfying this rule. You can use wildcards to match multiple approvers with one value.
Approver type Info
Value
IAM user name or assumed role
SeniorDevelopers/*
Remove
Fully qualified ARN
s:iam::123456789012:user/Mary_Major
Remove
Add
Branch filters - optional
Use branch filters to only apply this template to a pull request if the destination branch name matches a name in the filter list.
Branch name
main
Remove
Add
Associated repositories
Repositories - optional
MyDemoRepo
MyTestRepo

So erstellen Sie eine Genehmigungsregelvorlage (AWS CLI)

Sie können die AWS CLI zum Erstellen von Genehmigungsregelvorlagen verwenden. Wenn Sie die AWS CLI verwenden, können Sie Zielverweise für die Vorlage angeben, damit sie nur für Pull-Anforderungen gilt, deren Ziel-Branches denen in der Vorlage entsprechen.

So erstellen Sie eine Genehmigungsregelvorlage (AWS CLI)

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl create-approval-rule-template unter Angabe der folgenden Informationen aus:

   - Name der Genehmigungsregelvorlage. Empfohlen wird ein Name, der ihren Zweck beschreibt.
   - Beschreibung der Genehmigungsregelvorlage. Wie beim Namen sollte die Eingabe aussagekräftig sein.
   - JSON-Struktur der Genehmigungsregelvorlage. Diese Struktur kann Anforderungen für Zielverweise umfassen, d. h. die Ziel-Branches für Pull-Anforderungen, für die die Genehmigungsregel gilt, und Genehmigungs-Pool-Mitglieder, also Benutzer, deren Genehmigungen für die Anzahl erforderlicher Genehmigungen gezählt werden.

   Beim Erstellen des Inhalts der Genehmigungsregel können Sie Genehmiger in einem Genehmigungs-Pool auf zwei Arten angeben:

   - CodeCommitApprovers: Für diese Option sind nur ein Amazon Web Services Services-Konto und eine Ressource erforderlich. Es kann sowohl für IAM-Benutzer als auch für Verbundzugriffsbenutzer verwendet werden, deren Name mit dem angegebenen Ressourcennamen übereinstimmt. Dies ist eine sehr leistungsfähige Option, die hohe Flexibilität bietet. Wenn Sie beispielsweise dasAWS Konto 123456789012 angeben und**`Mary_Major`** alle folgenden Genehmigungen als Genehmigungen dieses Benutzers gezählt werden:
     - Ein IAM-Benutzer im Konto (`arn:aws:iam::123456789012:user/Mary_Major`)
     - Ein Verbundbenutzer in IAM mit dem Namen Mary_Major (`arn:aws:sts::123456789012:federated-user/Mary_Major`)

     Diese Option erkennt keine aktive Sitzung von jemandem, der die Rolle von *`SeniorDevelopers`*mit dem Rollensitzungsnamen *`Mary_Major`* (`arn:aws:sts::123456789012:assumed-role/`*`SeniorDevelopers`*`/`*`Mary_Major`*) übernimmt, es sei denn, Sie fügen einen Platzhalter (`*Mary_Major`) ein.

- Vollqualifizierter ARN: Mit dieser Option können Sie den vollqualifizierten Amazon Resource Name (ARN) des IAM-Benutzers oder der IAM-Rolle angeben.

Weitere Informationen zu IAM-ARNs, Platzhaltern und IDs finden Sie unter IAM-IDs.

Im folgenden Beispiel wird eine Genehmigungsregelvorlage namens **`2-approver-rule-for-main`** mit der Beschreibung **`Requires two developers from the team to approve the pull request if the destination branch is main`** erstellt. Für die Vorlage müssen zwei Benutzer, die die Rolle **`CodeCommitReview`** angenommen haben, alle Pull-Anforderungen genehmigen, bevor sie im Branch **`main`** zusammengeführt werden können:

```
aws codecommit create-approval-rule-template --approval-rule-template-name 2-
approver-rule-for-main --approval-rule-template-description "Requires two
 developers from the team to approve the pull request if the destination branch
 is main" --approval-rule-template-content "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "approvalRuleTemplate": {
        "approvalRuleTemplateName": "2-approver-rule-for-main",
        "creationDate": 1571356106.936,
        "approvalRuleTemplateId": "dd8b17fe-EXAMPLE",
        "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "approvalRuleTemplateDescription": "Requires two developers from the team
 to approve the pull request if the destination branch is main",
        "lastModifiedDate": 1571356106.936,
        "ruleContentSha256": "4711b576EXAMPLE"
    }
}
```

Eine Genehmigungsregelvorlage einem Repository zuordnen.

Vorlagen für Genehmigungsregeln werden in einem bestimmtenAWS-Region Bereich erstellt, wirken sich jedochAWS-Region erst dann auf Repositorys aus, wenn sie verknüpft sind. Um eine Vorlage einzelnen oder mehreren Repositorys zuzuweisen, müssen Sie die Vorlage den betreffenden Repositorys zuordnen. Sie können eine einzelne Vorlage auf mehrere Repositorys in einem anwendenAWS-Region. Auf diese Weise können Sie den Entwicklungs-Workflow in Repositorys automatisieren und standardisieren, indem Sie konsistente Bedingungen für das Genehmigen und das Zusammenführen von Pull-Anforderungen schaffen.

Sie können eine Genehmigungsregelvorlage nur den Repositorys zuordnen, inAWS-Region denen die Vorlage für Genehmigungsregeln erstellt wurde.

Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unterBerechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln undVon AWS verwaltete Richtlinien für CodeCommit.

Themen

- Eine Genehmigungsregelvorlage zuordnen (Konsole)
- Eine Genehmigungsregelvorlage zuordnen (AWS CLI)

Eine Genehmigungsregelvorlage zuordnen (Konsole)

Möglicherweise haben Sie einer Genehmigungsregelvorlage bereits bei der Erstellung Repositorys zugeordnet. (Dieser Schritt ist optional.) Sie können Zuordnungen hinzufügen oder entfernen, indem Sie die Vorlage bearbeiten.

So ordnen Sie eine Genehmigungsregelvorlage Repositorys zu

1. Öffnen Sie dieCodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie Approval rule templates (Genehmigungsregelvorlagen) aus. Wählen Sie die Vorlage und dann Edit (Bearbeiten) aus.
3. Wählen Sie unter Associated Repositories (Zugeordnete Repositorys) die Repositorys in der Liste Repositories (Repositorys) aus. Jedes zugeordnete Repository wird unter dem Listenfeld angezeigt.

4. Wählen Sie Speichern. Genehmigungsregeln werden nun allen in den zugeordneten Repositorys erstellten Pull-Anforderungen zugewiesen.

Eine Genehmigungsregelvorlage zuordnen (AWS CLI)

Sie können eine Genehmigungsregelvorlage mit der AWS CLI einzelnen oder mehreren Repositorys zuordnen.

So ordnen Sie eine Vorlage einem einzelnen Repository zu

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl associate-approval-rule-template-with-repository unter Angabe der folgenden Informationen aus:

   - Name der Genehmigungsregelvorlage, die Sie einem Repository zuordnen möchten.
   - Name des Repositorys, das der Genehmigungsregelvorlage zugeordnet werden soll.

   Um beispielsweise eine Vorlage für Genehmigungsregeln mit dem Namen *2-* einem Repositoryapprover-rule-for-main mit dem Namen zu verknüpfen *MyDemoRepo*:

   ```
   aws codecommit associate-approval-rule-template-with-repository --repository-
   name MyDemoRepo --approval-rule-template-name 2-approver-rule-for-main
   ```

2. Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.

So ordnen Sie eine Vorlage mehreren Repositorys zu

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl batch-associate-approval-rule-template-with-repositories unter Angabe der folgenden Informationen aus:

   - Name der Genehmigungsregelvorlage, die Sie einem Repository zuordnen möchten.
   - Namen der Repositorys, die der Genehmigungsregelvorlage zugeordnet werden sollen.

   So ordnen Sie beispielsweise eine Genehmigungsregelvorlage namens **2-approver-rule-for-main** Repositorys mit den Namen **MyDemoRepo** und **MyOtherDemoRepo** zu:

```
aws codecommit batch-associate-approval-rule-template-with-repositories --
repository-names "MyDemoRepo", "MyOtherDemoRepo" --approval-rule-template-name 2-
approver-rule-for-main
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "associatedRepositoryNames": [
        "MyDemoRepo",
        "MyOtherDemoRepo"
    ],
    "errors": []
}
```

Verwalten von Genehmigungsregelvorlagen

Sie können die Vorlagen für Genehmigungsregeln in einer verwaltenAWS-Region, um zu verstehen, wie sie verwendet werden und wofür sie bestimmt sind. Sie können beispielsweise die Namen und Beschreibungen von Genehmigungsregelvorlagen bearbeiten, um andere Benutzer über den Verwendungszweck zu informieren. Sie können alle Vorlagen für Genehmigungsregeln in einerAWS-Region auflisten und Informationen über den Inhalt und die Struktur einer Vorlage abrufen. Sie können ermitteln, welche Vorlagen einem Repository und welche Repositorys einer Vorlage zugeordnet sind.

Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unterBerechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln undVon AWS verwaltete Richtlinien für CodeCommit.

Vorlagen für Genehmigungsregeln verwalten (Konsole)

Sie können Genehmigungsregelvorlagen in der CodeCommit-Konsole anzeigen und verwalten.

So verwalten Sie Genehmigungsregelvorlagen

1. Öffnen Sie dieCodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie Vorlagen für Genehmigungsregeln, um die Liste der Genehmigungsregelvorlagen an demAWS-Region Ort anzuzeigen, an dem Sie angemeldet sind.

> **Note**
>
> Genehmigungsregelvorlagen sind nur in den verfügbar, inAWS-Region denen sie erstellt wurden.

3. Wenn Sie Änderungen an einer Vorlage vornehmen möchten, wählen Sie sie in der Liste und dann Edit (Bearbeiten) aus.
4. Nehmen Sie die gewünschten Änderungen vor und wählen Sie dann Save.

Vorlagen für Genehmigungsregeln verwalten (AWS CLI)

Sie können Genehmigungsregelvorlagen mit den folgenden AWS CLI-Befehlen verwalten:

- list-approval-rule-templates, um eine Liste aller Genehmigungsregelvorlagen in einer anzuzeigen.AWS-Region
- get-approval-rule-template zeigt den Inhalt einer Genehmigungsregelvorlage an.
- update-approval-rule-template-content ändert den Inhalt einer Genehmigungsregelvorlage.
- update-approval-rule-template-name ändert den Namen einer Genehmigungsregelvorlage.
- update-approval-rule-template-description ändert die Beschreibung einer Genehmigungsregelvorlage.
- list-repositories-for-approval-rule-template zeigt alle Repositorys an, die einer Genehmigungsregelvorlage zugeordnet sind.
- list-associated-approval-rule-templates-for-repository zeigt alle Genehmigungsregelvorlagen an, die einem Repository zugeordnet sind.

So listen Sie alle Genehmigungsregelvorlagen in einer aufAWS-Region

1. Führen Sie am Terminal oder über die Befehlszeile den Befehl list-approval-rule-templates aus. So listen Sie beispielsweise alle Genehmigungsregelvorlagen in der Region USA Ost (Ohio) auf:

```
aws codecommit list-approval-rule-templates --region us-east-2
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
```

```
    "approvalRuleTemplateNames": [
        "2-approver-rule-for-main",
        "1-approver-rule-for-all-pull-requests"
    ]
}
```

So rufen Sie den Inhalt einer Genehmigungsregelvorlage ab

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl get-approval-rule-template aus und geben den Namen der Genehmigungsregelvorlage an:

```
aws codecommit get-approval-rule-template --approval-rule-template-name 1-approver-
rule-for-all-pull-requests
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "approvalRuleTemplate": {
        "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",\"Statements
\": [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers
\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "ruleContentSha256": "621181bbEXAMPLE",
        "lastModifiedDate": 1571356106.936,
        "creationDate": 1571356106.936,
        "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan",
        "approvalRuleTemplateId": "a29abb15-EXAMPLE",
        "approvalRuleTemplateDescription": "All pull requests must be approved by
 one developer on the team."
    }
}
```

So aktualisieren Sie den Inhalt einer Genehmigungsregelvorlage

1. Führen Sie am Terminal oder an der Eingabeaufforderung den Befehl update-approval-rule-template-content aus und geben Sie den Namen der Vorlage und den geänderten Inhalt an. So ändern Sie beispielsweise den Inhalt einer Genehmigungsregelvorlage namens **`1-approver-rule`**, um den Genehmigungs-Pool für Benutzer, die die Rolle **`CodeCommitReview`** annehmen, neu zu definieren:

```
aws codecommit update-approval-rule-template-content --approval-rule-template-
name 1-approver-rule --new-rule-content "{\"Version\": \"2018-11-08\",
\"DestinationReferences\": [\"refs/heads/main\"],\"Statements\": [{\"Type
\": \"Approvers\",\"NumberOfApprovalsNeeded\": 2,\"ApprovalPoolMembers\":
 [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}"
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "approvalRuleTemplate": {
        "creationDate": 1571352720.773,
        "approvalRuleTemplateDescription": "Requires 1 approval for all pull
 requests from the CodeCommitReview pool",
        "lastModifiedDate": 1571358728.41,
        "approvalRuleTemplateId": "41de97b7-EXAMPLE",
        "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",\"Statements
\": [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers
\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
        "ruleContentSha256": "2f6c21a5EXAMPLE",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan"
    }
}
```

So aktualisieren Sie den Namen einer Genehmigungsregelvorlage

1. Führen Sie den Befehl update-approval-rule-template-name am Terminal oder an der Eingabeaufforderung aus und geben Sie den aktuellen Namen und den neuen Namen an. So ändern Sie beispielsweise den Namen einer Genehmigungsregelvorlage von **`1-approver-rule`** in **`1-approver-rule-for-all-pull-requests`**:

```
aws codecommit update-approval-rule-template-name --old-approval-rule-template-name
 "1-approver-rule" --new-approval-rule-template-name "1-approver-rule-for-all-pull-
requests"
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "approvalRuleTemplate": {
        "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
```

```
        "lastModifiedDate": 1571358241.619,
        "approvalRuleTemplateId": "41de97b7-EXAMPLE",
        "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",\"Statements
\": [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers
\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "creationDate": 1571352720.773,
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Mary_Major",
        "approvalRuleTemplateDescription": "All pull requests must be approved by
 one developer on the team.",
        "ruleContentSha256": "2f6c21a5cEXAMPLE"
    }
}
```

So aktualisieren Sie die Beschreibung einer Genehmigungsregelvorlage

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl update-approval-rule-template-description aus und geben Sie den Namen der Genehmigungsregelvorlage und die neue Beschreibung an:

```
aws codecommit update-approval-rule-template-description --approval-rule-template-
name "1-approver-rule-for-all-pull-requests" --approval-rule-template-description
 "Requires 1 approval for all pull requests from the CodeCommitReview pool"
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "approvalRuleTemplate": {
        "creationDate": 1571352720.773,
        "approvalRuleTemplateDescription": "Requires 1 approval for all pull
 requests from the CodeCommitReview pool",
        "lastModifiedDate": 1571358728.41,
        "approvalRuleTemplateId": "41de97b7-EXAMPLE",
        "approvalRuleTemplateContent": "{\"Version\": \"2018-11-08\",\"Statements
\": [{\"Type\": \"Approvers\",\"NumberOfApprovalsNeeded\": 1,\"ApprovalPoolMembers
\": [\"arn:aws:sts::123456789012:assumed-role/CodeCommitReview/*\"]}]}",
        "approvalRuleTemplateName": "1-approver-rule-for-all-pull-requests",
        "ruleContentSha256": "2f6c21a5EXAMPLE",
        "lastModifiedUser": "arn:aws:iam::123456789012:user/Li_Juan"
    }
}
```

So listen Sie alle Repositorys auf, die einer Vorlage zugeordnet sind

1. Führen Sie in der Befehlszeile oder am Terminal den Befehl list-repositories-for-approval-rule-template aus und geben Sie den Namen der Vorlage an:

```
aws codecommit list-repositories-for-approval-rule-template --approval-rule-
template-name 2-approver-rule-for-main
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "repositoryNames": [
        "MyDemoRepo",
        "MyClonedRepo"
    ]
}
```

So listen Sie alle Vorlagen auf, die einem Repository zugeordnet sind

1. Führen Sie in der Befehlszeile oder am Terminal den Befehl list-associated-approval-rule-templates-for-repository aus und geben Sie den Namen des Repositorys an:

```
aws codecommit list-associated-approval-rule-templates-for-repository --repository-
name MyDemoRepo
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "approvalRuleTemplateNames": [
        "2-approver-rule-for-main",
        "1-approver-rule-for-all-pull-requests"
    ]
}
```

So trennen Sie eine Genehmigungsregelvorlage.

Wenn die von einer Genehmigungsregelvorlage generierten Genehmigungsregeln für den Workflow Ihres Teams in einem Repository nicht mehr geeignet sind, können Sie die Zuordnung der Vorlage zu diesem Repository aufheben. Durch das Aufheben der Zuordnung einer Vorlage werden keine

Genehmigungsregeln entfernt, die erstellt wurden, während die Vorlage dem Repository zugeordnet war.

Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unterBerechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln undVon AWS verwaltete Richtlinien für CodeCommit.

So trennen Sie eine Genehmigungsregelvorlage (Konsole)

Mit der Konsole können Sie die Zuordnung zwischen einem Repository und einer Genehmigungsregelvorlage aufheben.

So heben Sie die Zuordnung einer Genehmigungsregelvorlage zu Repositorys auf

1. Öffnen Sie dieCodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie Approval rule templates (Genehmigungsregelvorlagen) aus. Wählen Sie die Vorlage aus, deren Zuordnung zu einzelnen oder mehreren Repositorys aufgehoben werden soll. Wählen Sie dann Edit (Bearbeiten) aus.
3. Wählen Sie unter Associated repositories (Zugeordnete Repositorys) das X neben den Repositorys aus, deren Zuordnung aufgehoben werden soll. Die Repository-Namen werden nicht mehr angezeigt.
4. Wählen Sie Speichern. Genehmigungsregeln werden auf in diesen Repositorys erstellte Pull-Anforderungen nicht angewendet. Die Regeln werden weiterhin auf Pull-Anforderungen angewendet, die bei bestehender Zuordnung getätigt wurden.

Eine Genehmigungsregelvorlage aufheben (AWS CLI)

Sie können die AWS CLI verwenden, um die Zuordnung einzelner oder mehrerer Repositorys zu einer Genehmigungsregelvorlage aufzuheben.

So heben Sie die Zuordnung einer Genehmigungsregelvorlage zu einem Repository auf

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl disassociate-approval-rule-template-from-repository unter Angabe der folgenden Informationen aus:
   - Name der Genehmigungsregelvorlage.
   - Name des Repositorys.

So heben Sie beispielsweise die Zuordnung einer Genehmigungsregelvorlage namens **1-approver-rule-for-all-pull-requests** zu einem Repository namens **MyDemoRepo** auf:

```
aws codecommit disassociate-approval-rule-template-from-repository --repository-name MyDemoRepo --approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

2. Bei erfolgreicher Ausführung gibt dieser Befehl nichts zurück.

So heben Sie die Zuordnung einer Genehmigungsregelvorlage zu mehreren Repositorys auf

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl batch-disassociate-approval-rule-template-from-repositories unter Angabe der folgenden Informationen aus:

   - Name der Genehmigungsregelvorlage.
   - Namen der Repositorys.

   So heben Sie beispielsweise die Zuordnung einer Genehmigungsregelvorlage namens **1-approver-rule-for-all-pull-requests** zu den Repositorys **MyDemoRepo** und **MyOtherDemoRepo** auf:

```
aws codecommit batch-disassociate-approval-rule-template-from-repositories --repository-names "MyDemoRepo", "MyOtherDemoRepo" --approval-rule-template-name 1-approver-rule-for-all-pull-requests
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "disassociatedRepositoryNames": [
        "MyDemoRepo",
        "MyOtherDemoRepo"
    ],
    "errors": []
}
```

Eine Genehmigungsregelvorlage löschen

Sie können eine Genehmigungsregelvorlage löschen, wenn Sie sie nicht in Repositorys verwenden. Das Löschen nicht verwendeter Genehmigungsregelvorlagen erleichtert das Verwalten der und die Suche nach Vorlagen, die für Ihre Workflows geeignet sind.

Weitere Informationen zu verwalteten Richtlinien und Berechtigungen für Genehmigungsregelvorlagen finden Sie unterBerechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln undVon AWS verwaltete Richtlinien für CodeCommit.

Themen

- Eine Genehmigungsregelvorlage löschen (Konsole)
- Eine Genehmigungsregelvorlage löschen (AWS CLI)

Eine Genehmigungsregelvorlage löschen (Konsole)

Sie können eine Genehmigungsregelvorlage löschen, wenn sie für Ihre Entwicklungsarbeit nicht mehr benötigt wird. Wenn Sie eine Genehmigungsregelvorlage mit der Konsole löschen, wird beim Löschen die Zuordnung zu allen Repositorys aufgehoben.

So löschen Sie eine Genehmigungsregelvorlage

1. Öffnen Sie dieCodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie Approval rule templates (Genehmigungsregelvorlagen) aus. Wählen Sie die zu löschende Vorlage aus und klicken Sie auf Delete (Löschen).

Eine Genehmigungsregelvorlage löschen (AWS CLI)

Sie können die AWS CLI verwenden, um eine Genehmigungsregel zu löschen, nachdem deren Zuordnung zu allen Repositorys aufgehoben wurde. Weitere Informationen finden Sie unter Eine Genehmigungsregelvorlage aufheben (AWS CLI).

So löschen Sie eine Genehmigungsregelvorlage

1. Führen Sie an einem Terminal oder in der Befehlszeile den Befehl delete-approval-rule-template aus und geben Sie den Namen der zu löschenden Genehmigungsregelvorlage an:

```
aws codecommit delete-approval-rule-template --approval-rule-template-name 1-approver-for-all-pull-requests
```

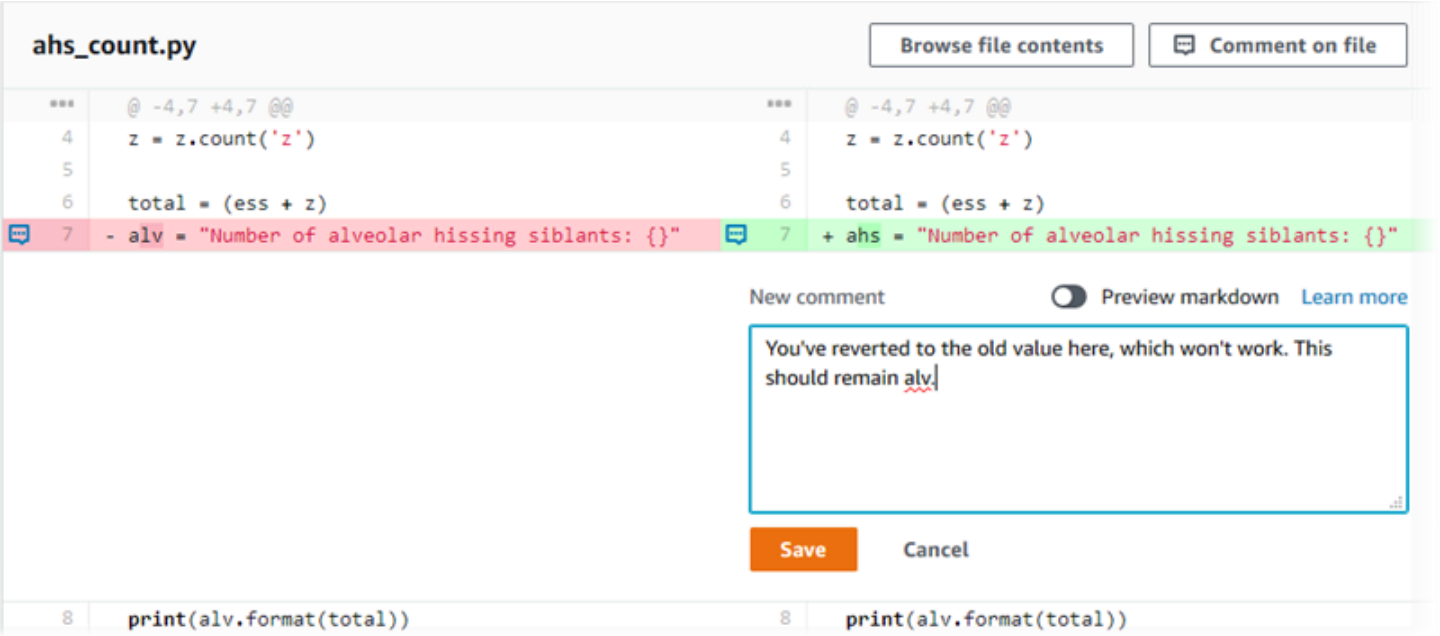
2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte. Wenn die Genehmigungsregelvorlage bereits gelöscht wurde, gibt dieser Befehl keinen Wert zurück.

```
{
    "approvalRuleTemplateId": "41de97b7-EXAMPLE"
}
```

Mit Commits in Repositorys arbeiten AWS CodeCommit

Commits sind Snapshots der Inhalte und der Änderungen, die an den Inhalten des Repositorys vorgenommen wurden. Jedes Mal, wenn ein Benutzer einen Commit ausführt und eine Änderung anstößt, werden diese Informationen gespeichert. Zudem wird gespeichert, wer die Änderung ausgeführt hat, sowie das Datum und die Uhrzeit des Commits und die Änderungen, die mit dem Commit ausgeführt wurden. Zur einfachen Identifizierung spezifischer Commits lassen sich Tags hinzufügen. In CodeCommit können Sie:

- Commits überprüfen
- Den Verlauf von Commits in einem Diagramm darstellen
- Einen Commit mit dem übergeordneten Commit oder einem anderen Spezifizierer vergleichen
- Ihren Commits Kommentare hinzufügen und auf Kommentare anderer Benutzer antworten



Bevor Sie Commits in ein CodeCommit Repository übertragen können, müssen Sie Ihren lokalen Computer so einrichten, dass er eine Verbindung zum Repository herstellt. Die einfachste Methode dazu finden Sie unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden.

Hinweise zur Arbeit mit anderen Aspekten Ihres Repositorys finden Sie unter CodeCommitArbeiten mit Repositorien,Arbeiten mit Dateien, Verwenden von Pull-AnforderungenArbeiten mit Zweigen, undArbeiten mit Benutzereinstellungen.

Themen

- Erstellen Sie einen Commit in AWS CodeCommit
- Commit-Details anzeigen in AWS CodeCommit
- Vergleiche Commits in AWS CodeCommit
- Kommentieren Sie einen Commit in AWS CodeCommit
- Erstellen Sie ein Git-Tag in AWS CodeCommit
- Git-Tag-Details anzeigen in AWS CodeCommit
- Lösche ein Git-Tag in AWS CodeCommit

Erstellen Sie einen Commit in AWS CodeCommit

Wenn Sie den ersten Commit für ein neues Repository erstellen, verwenden Sie den Befehl AWS CLI und den put-file Befehl. Dadurch wird der erste Commit erstellt und Sie können den Standard-Branch für Ihr neues Repository erstellen und angeben. Du kannst Git oder the verwenden AWS CLI , um einen Commit in einem CodeCommit Repository zu erstellen. Wenn das lokale Repo mit einem CodeCommit Repository verbunden ist, verwendest du Git, um den Commit vom lokalen Repo in das Repository zu übertragen. CodeCommit Informationen zum Erstellen eines Commits direkt in der CodeCommit Konsole finden Sie unter Erstellen oder Hinzufügen einer Datei zu einemAWS CodeCommitEndlager und. Bearbeiten Sie den Inhalt einer Datei in einemAWS CodeCommit Repository

> **Note**
>
> Als bewährte Methode empfehlen wir, die neuesten unterstützten Versionen von AWS CLI, Git und anderer Software zu verwenden. Wenn Sie die verwenden AWS CLI, stellen Sie sicher, dass Sie eine aktuelle Version installiert haben, um sicherzustellen, dass Sie eine Version verwenden, die den `create-commit` Befehl enthält.

Themen

- Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI
- Einen Commit mit einem Git-Client erstellen
- Erstelle einen Commit dem AWS CLI

Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI

Sie können den Befehl AWS CLI und den `put-file` Befehl verwenden, um Ihren ersten Commit für ein Repository zu erstellen. Mit put-file Using wird ein erster Commit erstellt, der eine Datei zu Ihrem leeren Repository hinzufügt, und es wird ein Branch mit dem von Ihnen angegebenen Namen erstellt. Es bestimmt den neuen Branch als Standard-Branch für dein Repository.

> **Note**
>
> Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den. AWS CLI Weitere Informationen finden Sie unter Befehlszeilenreferenz.

Um den ersten Commit für ein Repository mit dem zu erstellen AWS CLI

1. Erstellen Sie auf Ihrem lokalen Computer die Datei, die Sie als erste Datei zum CodeCommit Repository hinzufügen möchten. Eine gängige Praxis besteht darin, eine `README.md` Markdown-Datei zu erstellen, die anderen Repository-Benutzern den Zweck dieses Repositorys erklärt. Wenn Sie eine `README.md` Datei hinzufügen, wird der Inhalt der Datei automatisch unten auf der Codeseite für Ihr Repository in der CodeCommit Konsole angezeigt.
2. Führen Sie am Terminal oder in der Befehlszeile den Befehl put-file unter Angabe der folgenden Informationen aus:

   - Der Name des Repositorys, dem Sie die erste Datei hinzufügen möchten.
   - Der Name des Branches, den Sie als Standard-Branch erstellen möchten.
   - Der lokale Speicherort der Datei Die Syntax für diesen Standort hängt von Ihrem lokalen Betriebssystem ab.
   - Der Name der Datei, die Sie hinzufügen möchten, einschließlich des Pfads, in dem die aktualisierte Datei im Repository gespeichert ist.
   - Der Benutzername und die E-Mail-Adresse, die Sie dieser Datei zuordnen möchten.
   - Eine Commit-Nachricht, die erklärt, warum Sie diese Datei hinzugefügt haben

   Der Benutzername, die E-Mail-Adresse und die Commit-Nachricht sind optional, können aber anderen Benutzern helfen, zu erfahren, wer die Änderung vorgenommen hat und warum. Wenn Sie keinen Benutzernamen angeben, wird CodeCommit standardmäßig Ihr IAM-Benutzername oder eine Ableitung Ihres Konsolen-Logins als Autorenname verwendet.

Zum Beispiel, um eine Datei mit dem Namen *README.md* mit dem Inhalt „Willkommen in unserem Team-Repository!“ hinzuzufügen *zu einem Repository, das MyDemoReponach einem Zweig namens development benannt ist:*

```
aws codecommit put-file --repository-name MyDemoRepo --branch-name development --
file-path README.md --file-content "Welcome to our team repository!" --name "Mary
 Major" --email "mary_major@example.com" --commit-message "I added a quick readme
 for our new team repository."
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "commitId": "724caa36EXAMPLE",
    "blobId": "a8a94062EXAMPLE",
    "treeId": "08b2fc73EXAMPLE"
}
```

Einen Commit mit einem Git-Client erstellen

Du kannst Commits mit einem Git-Client erstellen, der auf deinem lokalen Computer installiert ist, und diese Commits dann in dein CodeCommit Repository übertragen.

1. Sorgen Sie dafür, dass die Voraussetzungen erfüllt sind, einschließlich Einrichtung .

   > **Important**
   >
   > Wenn Sie die Einrichtung nicht abgeschlossen haben, können Sie mit Git weder eine Verbindung herstellen noch einen Commit für das Repository durchführen.

2. Vergewissern Sie sich, dass Sie den Commit für den richtigen Branch erstellen. Um eine Liste der verfügbaren Branches zu sehen und festzustellen, welchen Branch Sie derzeit verwenden, führen Sie git branch aus. Alle Branches werden angezeigt. Ihr aktueller Branch ist mit einem Sternchen (*) gekennzeichnet. Wenn Sie zu einem anderen Branch wechseln möchten, führen Sie git checkout ***branch-name*** aus. Wenn dies dein erster Commit ist, führe den git config Befehl aus, um deinen Git-Client so zu konfigurieren, dass er einen ersten Branch mit dem Namen erstellt, den du für diesen Branch verwenden möchtest. Wenn du zum Beispiel möchtest, dass dein Standard-Branch den Namen *development* hat:

```
git config --local init.defaultBranch development
```

> **Tip**
>
> Dieser Befehl ist nur in Git v.2.28 und höher verfügbar.
> Du kannst diesen Befehl auch ausführen, um deinen Standard-Branch-Namen **`development`** für alle neu erstellten Repositorys auf festzulegen:
>
> ```
> git config --global init.defaultBranch development
> ```

3. Ändern Sie den Branch (z. B. durch Hinzufügen, Ändern oder Löschen einer Datei).

   Erstellen Sie beispielsweise im lokalen Repository eine Datei `bird.txt` mit dem folgenden Text:

   ```
   bird.txt
   --------
   Birds (class Aves or clade Avialae) are feathered, winged, two-legged, warm-
   blooded, egg-laying vertebrates.
   ```

4. Führen Sie git status aus. Die Befehlsausgabe sollte besagen, dass `bird.txt` noch in keinem schwebenden Commit enthalten ist:

   ```
   ...
   Untracked files:
     (use "git add <file>..." to include in what will be committed)

           bird.txt
   ```

5. Führen Sie git add bird.txt aus, um die neue Datei in den schwebenden Commit aufzunehmen.

6. Wenn Sie den Befehl git status erneut ausführen, sollte die Befehlsausgabe der folgenden ähneln. Damit wird angegeben, dass `bird.txt` nun zum schwebenden Commit gehört oder für den Commit bereitgestellt wird:

   ```
   ...
   Changes to be committed:
     (use "git reset HEAD <file>..." to unstage)
   ```

```
        new file:   bird.txt
```

7. Um den Commit abzuschließen, führen Sie git commit mit der Option `-m` aus (z. B. git commit -m "***Adding bird.txt to the repository.***"). Mit der Option `-m` wird die Commit-Nachricht erstellt.
8. Wenn Sie den Befehl git status erneut ausführen, sollte die Befehlsausgabe der folgenden ähneln. Es zeigt an, dass der Commit bereit ist, vom lokalen Repo in das Repository übertragen zu werden: CodeCommit

```
...
nothing to commit, working directory clean
```

9. Bevor Sie den finalisierten Commit aus dem lokalen Repo in das CodeCommit Repository übertragen, können Sie sehen, was Sie durch Ausführen übertragen, wobei *remote-name* der Spitzname istgit diff --stat ***remote-name***/***branch-name***, den das lokale Repo für das CodeCommit Repository verwendet, und *branch-name der Name des Branches*, der verglichen werden soll.

> **Tip**
>
> Um den Remote-Namen zu erhalten, führen Sie git remote aus. Um eine Liste der Branch-Namen zu erhalten, führen Sie den Befehl git branch aus. Der aktuelle Branch ist mit einem Sternchen (*) gekennzeichnet. Sie können auch git status ausführen, um den aktuellen Branch-Namen abzurufen.

> **Note**
>
> *Wenn Sie das Repository geklont haben, ist remote-name aus der Sicht des lokalen Repositorys nicht der Name des Repositorys.* CodeCommit Beim Klonen eines Repositorys wird *remote-name* automatisch auf `origin` festgelegt.

Beispielsweise würde die Ausgabe von git diff --stat origin/main der folgenden gleichen:

```
bird.txt | 1 +
1 file changed, 1 insertion(+)
```

In der Ausgabe wird davon ausgegangen, dass Sie das lokale Repository bereits mit dem Repository verbunden haben. CodeCommit (Detaillierte Anweisungen finden Sie unter Herstellen einer Verbindung mit einem Repository.)

10. Wenn Sie bereit sind, den Commit vom lokalen Repo in das CodeCommit Repository zu übertragen, führen Sie den Befehl aus. Dabei steht *remote-name* für den Spitznamengit push ***remote-name branch-name***, den das lokale Repo für das CodeCommit Repository verwendet, und *branch-name für den Namen des Branches*, der in das Repository übertragen werden soll. CodeCommit

    Beispielsweise würde die Ausgabe von git push origin main der folgenden gleichen:

    Für HTTPS:

    ```
    Counting objects: 7, done.
    Delta compression using up to 4 threads.
    Compressing objects: 100% (4/4), done.
    Writing objects: 100% (5/5), 516 bytes | 0 bytes/s, done.
    Total 5 (delta 2), reused 0 (delta 0)
    remote:
    To https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
        b9e7aa6..3dbf4dd main -> main
    ```

    Für SSH:

    ```
    Counting objects: 7, done.
    Delta compression using up to 4 threads.
    Compressing objects: 100% (4/4), done.
    Writing objects: 100% (5/5), 516 bytes | 0 bytes/s, done.
    Total 5 (delta 2), reused 0 (delta 0)
    remote:
    To ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo
        b9e7aa6..3dbf4dd main -> main
    ```

    > **Tip**
    >
    > Wenn Sie dem Befehl git push die Option `-u` hinzufügen (z. B. git push -u origin main), müssen Sie künftig nur noch git push ausführen, da die Upstream-Nachverfolgungsdaten

bereits festgelegt wurden. Führen Sie git remote show ***remote-name*** aus (z. B. git remote show origin), um Upstream-Nachverfolgungsdaten abzurufen.

Weitere Optionen findest du in deiner Git-Dokumentation.

Erstelle einen Commit dem AWS CLI

Sie können den Befehl AWS CLI und den `create-commit` Befehl verwenden, um einen Commit für ein Repository an der Spitze eines bestimmten Branches zu erstellen. Sie können auch einen nicht referenzierten Mergecommit für die Zusammenführung von zwei Commit-Spezifizierern erstellen. Weitere Informationen finden Sie unter Erstellen eines nicht referenzierten Commits.

> **Note**
>
> Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

So erstellen Sie einen Commit

1. Nehmen Sie auf Ihrem lokalen Computer die Änderungen vor, die Sie in das CodeCommit Repository übernehmen möchten.
2. Führen Sie am Terminal oder in der Befehlszeile den Befehl create-commit unter Angabe der folgenden Informationen aus:

   - Das Repository, an das Sie die Änderungen übertragen möchten.
   - Der Branch, an den Sie die Änderungen übertragen möchten.
   - Die vollständige Commit-ID des letzten an diesem Branch vorgenommenen Commits (auch als Spitzen- oder Head-Commit- oder übergeordnete Commit-ID bezeichnet).
   - Gibt an, ob leere Ordner beibehalten werden, wenn die von Ihnen vorgenommenen Änderungen den Inhalt dieser Ordner löschen. Der Standardwert ist "false".
   - Die Informationen zu den Dateien, die Sie hinzufügen, ändern oder löschen möchten.
   - Der Benutzername und die E-Mail-Adresse, die Sie mit diesen Änderungen verknüpfen möchten.
   - Eine Commit-Nachricht, die erklärt, warum Sie diese Änderungen vorgenommen haben.

Der Benutzername, die E-Mail-Adresse und die Commit-Nachricht sind optional. Sie machen anderen Benutzern aber leichter verständlich, von wem die Änderungen aus welchem Grund vorgenommen wurden. Wenn Sie keinen Benutzernamen angeben, wird CodeCommit standardmäßig Ihr IAM-Benutzername oder eine Ableitung Ihres Konsolen-Logins als Autorenname verwendet.

Zum Beispiel, um einen Commit für ein Repository zu erstellen, der eine README.md Datei zu einem Repository hinzufügt, das MyDemoRepoim Hauptzweig benannt ist. Der Inhalt der Datei ist Base64 und lautet „Willkommen in unserem Team-Repository!“ :

```
aws codecommit create-commit --repository-name MyDemoRepo --
branch-name main --parent-commit-id 4c925148EXAMPLE --put-files
 "filePath=README.md,fileContent=V2VsY29tZSB0byBvdXIgdGVhbSByZXBvc2l0b3J5IQo="
```

> **Tip**
>
> Führen Sie den Befehl get-branch aus, um die übergeordnete Commit-ID abzurufen.

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
    "commitId": "4df8b524-EXAMPLE",
    "treeId": "55b57003-EXAMPLE",
    "filesAdded": [
        {
            "blobId": "5e1c309dEXAMPLE",
            "absolutePath": "meeting.md",
            "fileMode": "NORMAL"
        }
    ],
    "filesDeleted": [],
    "filesUpdated": []
}
```

Um einen Commit zu erstellen, der Änderungen an den Dateien file1.py und file2.txt vornimmt, eine Datei von picture.png in image1.png

umbenennt und sie aus einem Verzeichnis namens pictures in ein Verzeichnis namens images verschiebt und eine Datei mit dem Namen ExampleSolution.py in einem Repository löscht, das nach einem Zweig benannt ist, MyFeatureBranchdessen neuester Commit die ID 4C925148 hat. Beispiel: MyDemoRepo

```
aws codecommit create-commit --repository-name MyDemoRepo --branch-
name MyFeatureBranch --parent-commit-id 4c925148EXAMPLE --name "Saanvi Sarkar"
 --email "saanvi_sarkar@example.com" --commit-message "I'm creating this commit to
 update a variable name in a number of files."
 --keep-empty-folders false  --put-files '{"filePath": "file1.py", "fileMode":
 "EXECUTABLE", "fileContent": "bucket_name = sys.argv[1] region = sys.argv[2]"}'
'{"filePath": "file2.txt", "fileMode": "NORMAL", "fileContent": "//Adding a comment
 to explain the variable changes in file1.py"}' '{"filePath": "images/image1.png",
"fileMode": "NORMAL", "sourceFile": {"filePath": "pictures/picture.png", "isMove":
 true}}' --delete-files filePath="ExampleSolution.py"
```

> **Note**
>
> Die Syntax für das Segment hängt von Ihrem Betriebssystem ab--put-files. Das obige Beispiel ist für Linux-, MacOS- oder Unix-Benutzer und Windows-Benutzer mit einem Bash-Emulator optimiert. Windows-Benutzer in der Befehlszeile oder in Powershell sollten eine für diese Systeme geeignete Syntax verwenden.

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
   "commitId": "317f8570EXAMPLE",
   "treeId": "347a3408EXAMPLE",
   "filesAdded": [
        {
        "absolutePath": "images/image1.png",
        "blobId": "d68ba6ccEXAMPLE",
        "fileMode": "NORMAL"
        }
    ],
    "filesUpdated": [
        {
        "absolutePath": "file1.py",
```

```
        "blobId": "0a4d55a8EXAMPLE",
        "fileMode": "EXECUTABLE"
        },
        {
        "absolutePath": "file2.txt",
        "blobId": "915766bbEXAMPLE",
        "fileMode": "NORMAL"
        }
    ],
    "filesDeleted": [
        {
        "absolutePath": "ExampleSolution.py",
        "blobId": "4f9cebe6aEXAMPLE",
        "fileMode": "EXECUTABLE"
        },
        {
        "absolutePath": "pictures/picture.png",
        "blobId": "fb12a539EXAMPLE",
        "fileMode": "NORMAL"
        }
    ]
}
```

Commit-Details anzeigen in AWS CodeCommit

Sie können die AWS CodeCommit Konsole verwenden, um den Verlauf der Commits in einem Repository zu durchsuchen. Auf diese Weise können Sie Änderungen an einem Repository erkennen und auch:

- Wann und von wem die Änderungen vorgenommen wurden.
- Wann bestimmte Commits in einen Branch zusammengeführt wurden.

Wenn Sie den Verlauf von Commits für einen Branch anzeigen, erhalten Sie zudem einen besseren Einblick in die Unterschiede zwischen den Branches. Wenn Sie das Markieren verwenden, können Sie außerdem schnell einen Commit anzeigen, der mit einem bestimmten Tag markiert wurde, und Sie sehen dessen übergeordnete Commits. In der Befehlszeile kannst du Git verwenden, um Details zu den Commits in einem lokalen Repo oder einem CodeCommit Repository anzuzeigen.

Durchsuche Commits in einem Repository

Sie können die AWS CodeCommit Konsole verwenden, um den Verlauf der Commits in einem Repository zu durchsuchen. Zudem können Sie eine Grafik der Commits im Repository und in dessen Branches im Zeitverlauf anzeigen. Auf diese Weise können Sie die Vorgänge des Repository im Verlauf erkennen. Sie sehen auch, wann Änderungen vorgenommen wurden.

> **Note**
>
> Wenn Sie mit dem Befehl git rebase einen Rebase für ein Repository ausführen, ändert sich der Verlauf des Repositorys. Commits werden dann möglicherweise nicht mehr in der richtigen Reihenfolge angezeigt. Weitere Informationen finden Sie unter Git Branching-Rebasing oder in Ihrer Git-Dokumentation.

Themen

- Den Commit-Verlauf eines Repositorys durchsuchen
- Sehen Sie sich ein Diagramm des Commit-Verlaufs eines Repositorys an

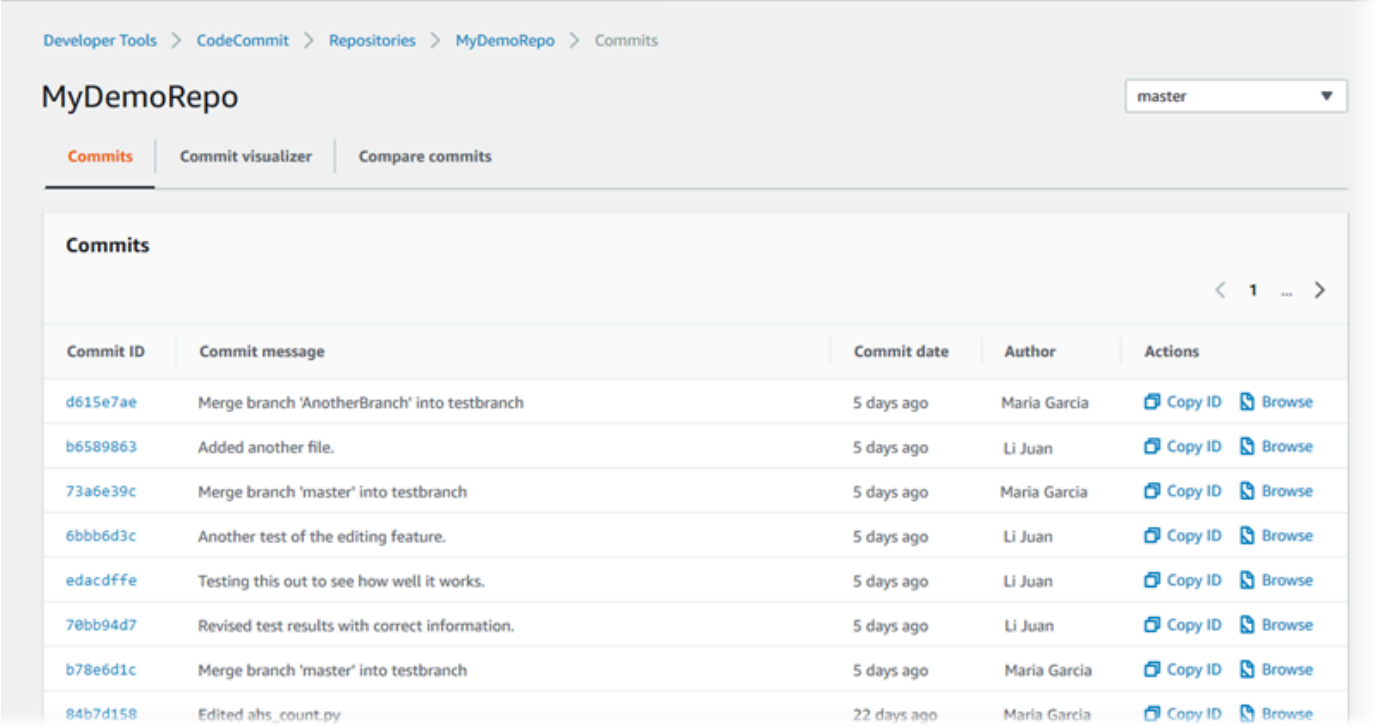
Den Commit-Verlauf eines Repositorys durchsuchen

Sie können den Commit-Verlauf für einen bestimmten Branch oder Tag des Repositorys durchsuchen, u. a. können Sie Informationen zum Committer und die Commit-Nachricht einsehen. Sie können auch den Code für einen Commit anzeigen.

So durchsuchen Sie den Commit-Verlauf

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, für das Sie den Commit-Verlauf prüfen möchten.
3. Wählen Sie im Navigationsbereich Commits aus. In der Ansicht des Commit-Verlaufs wird der Verlauf der Commits für das Repository im Standard-Branch in umgekehrter chronologischer Reihenfolge des Commit-Datums angezeigt. Datum und Uhrzeit entsprechen der Zeitzone Coordinated Universal Time (UTC). Sie können den Commit-Verlauf eines anderen Branches anzeigen, indem Sie auf die Schaltfläche der Ansichtsauswahl klicken und einen anderen Branch in der Liste auswählen. Wenn Sie Tags in Ihrem Repository verwenden, können Sie einen

Commit mit einem bestimmten Tag und dessen übergeordnete Commits anzeigen. Klicken Sie dazu auf die Schaltfläche der Ansichtsauswahl.



4. Um den Unterschied zwischen einem Commit und seinem übergeordneten Commit sowie alle Kommentare zu den Änderungen anzuzeigen, wählen Sie die abgekürzte Commit-ID. Weitere Informationen finden Sie unter Einen Commit mit seinem übergeordneten Commit vergleichen und Kommentar zu einem Commit. Wenn Sie den Unterschied zwischen einem Commit und einem beliebigen anderen Commit-Spezifizierer, einschließlich Branch, Tag und Commit-ID, anzeigen möchten, finden Sie unter Vergleichen von zwei beliebigen Commit-Spezifizierern weitere Informationen.
5. Führen Sie eine oder mehrere der folgenden Aktionen aus:

   - Um das Datum und die Uhrzeit einer Änderung anzuzeigen, bewegen Sie den Mauszeiger über das Commit-Datum.
   - Um die vollständige Commit-ID anzuzeigen, kopieren Sie sie und fügen Sie sie in einem Texteditor oder an anderer Stelle ein. Um sie zu kopieren, wählen Sie Copy ID (ID kopieren) aus.
   - Um den Code so anzuzeigen, wie er zum Zeitpunkt des Commits war, wählen Sie Browse (Durchsuchen) aus. Der Inhalt des Repositorys zum Zeitpunkt des Commits wird in der Ansicht Code angezeigt. Die Schaltfläche der Ansichtsauswahl zeigt die gekürzte Commit-ID an anstelle eines Branches oder Tags.

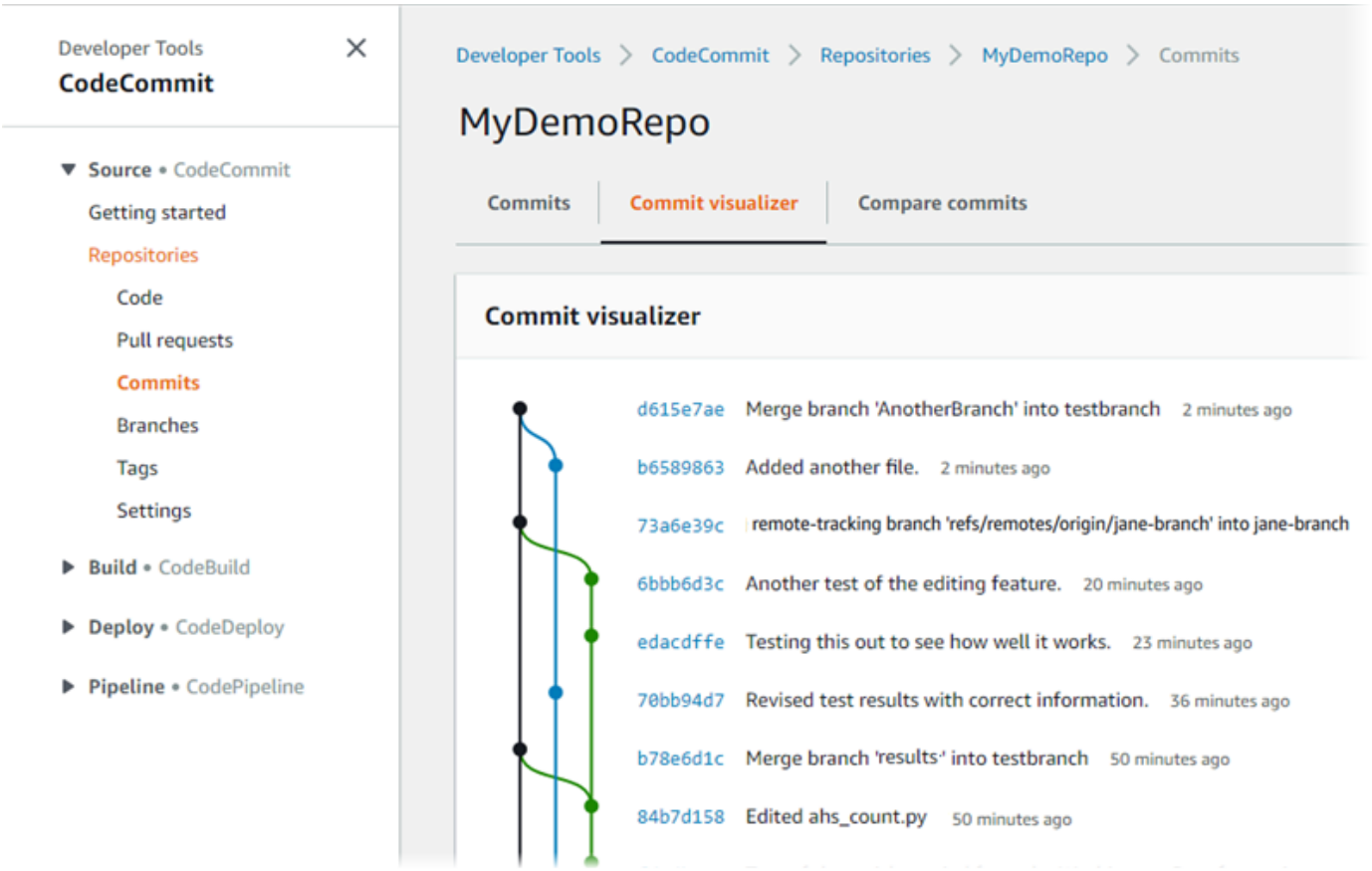
Sehen Sie sich ein Diagramm des Commit-Verlaufs eines Repositorys an

Sie können eine Grafik der Commits anzeigen, die für ein Repository durchgeführt wurden. Die Ansicht Commit Visualizer ist eine DAG-Darstellung (Directed Acyclic Graph, gerichteter azyklischer Graph) aller Commits, die für einen Branch eines Repository durchgeführt wurden. Anhand dieser grafische Darstellung können Sie besser erkennen, wann Commits und verbundene Funktionen hinzugefügt oder zusammengeführt wurden. Sie erhalten damit auch einen genaueren Einblick, wann eine Änderung im Relation zu anderen Änderungen vorgenommen wurde.

> **Note**
>
> Commits, die mithilfe der Fast-Forward-Methode zusammengeführt wurden, werden in der Commit-Grafik nicht in gesonderten Zeilen aufgeführt.

So zeigen Sie eine Commit-Grafik an

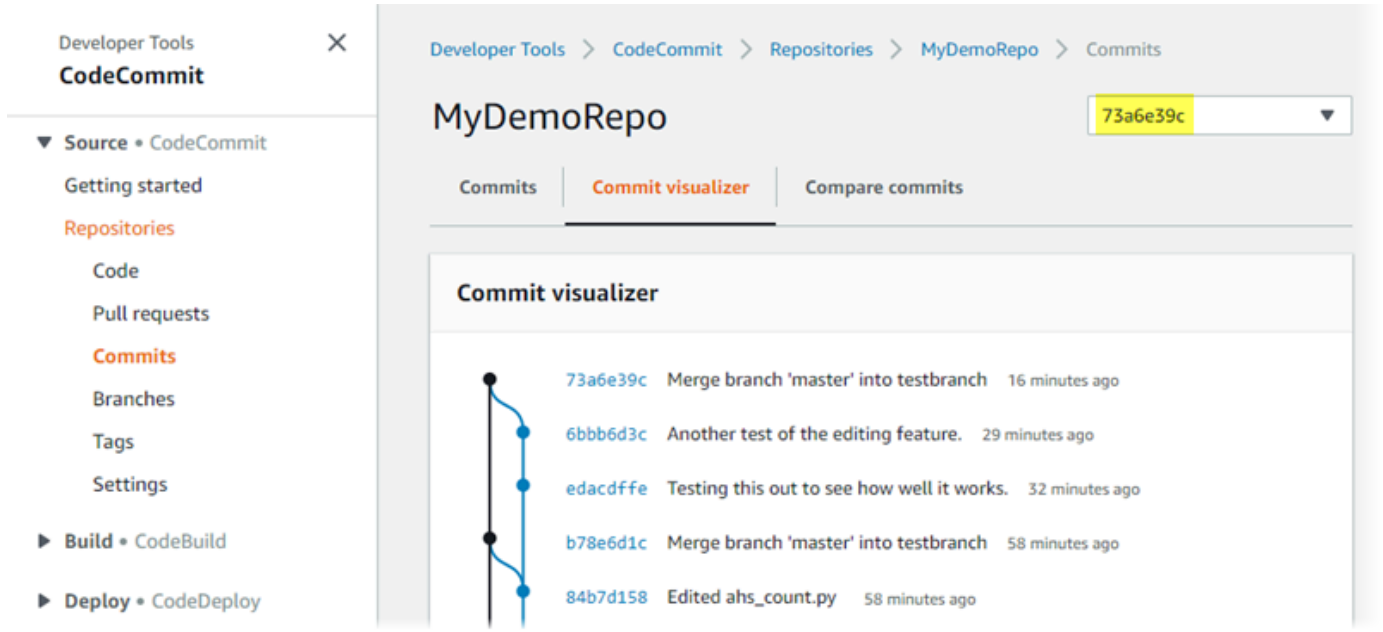
1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, für das Sie eine Commit-Grafik anzeigen möchten.
3. Wählen Sie im Navigationsbereich Commits und dann die Registerkarte Commit Visualizer aus.



In der Commit-Grafik wird die gekürzte Commit-ID und der Betreff jeder Commit-Nachricht neben diesem Punkt in der Grafik angezeigt.

> **Note**
>
> Die Grafik kann bis zu 35 Branches auf einer Seite anzeigen. Falls es mehr als 35 Branches gibt, ist die Grafik zu komplex für eine Anzeige. Es gibt zwei Methoden zum Vereinfachen der Ansicht:
>
> - Indem Sie die Grafik mithilfe der Schaltfläche der Ansichtsauswahl für einen bestimmten Branch anzeigen.
> - Indem Sie eine vollständige Commit-ID in das Suchfeld kopieren, um die Grafik für diesen Commit zu rendern.

4. Um eine neue Grafik von einem Commit zu rendern, wählen Sie den Punkt in der Grafik aus, der diesem Commit entspricht. Die Schaltfläche der Ansichtsauswahl ändert sich in die gekürzte Commit-ID.



Commit-Details anzeigen (AWS CLI)

Mit Git können Sie die Details zu Commits anzeigen. Sie können den auch verwenden AWS CLI , um Details zu den Commits in einem lokalen Repo oder in einem CodeCommit Repository anzuzeigen, indem Sie die folgenden Befehle ausführen:

- Um Informationen zu einem Commit anzuzeigen, führen Sie aws codecommit get-commit aus.
- Um Informationen zu mehreren Commits anzuzeigen, führen Sie aws codecommit batch-get-commits aus.
- Führen Sie zum Anzeigen von Informationen zu einem Merge-Commit aws codecommit get-merge-commit aus.
- Um Informationen zu Änderungen eines Commit-Spezifizierers (Branch, Tag, HEAD, oder andere vollständig qualifizierte Referenzen wie Commit-IDs) anzuzeigen, führen Sie aws codecommit get-differences aus.
- Um den base64-codierten Inhalt eines einzelnen Git-Blob-Objekts in einem Repository anzuzeigen, führen Sie aws codecommit get-blob aus.

So zeigen Sie Informationen zu einem Commit an

1. Führen Sie den Befehl aws codecommit get-commit aus und geben Sie Folgendes an:

- Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
- Die vollständige Commit-ID

Um beispielsweise Informationen über einen Commit mit der ID `317f8570EXAMPLE` in einem CodeCommit Repository mit dem Namen anzuzeigen`MyDemoRepo`:

```
aws codecommit get-commit  --repository-name MyDemoRepo  --commit-id
 317f8570EXAMPLE
```

2. Ist der Befehl erfolgreich, enthält die Ausgabe Folgendes:

- Informationen zum Autor des Commits (wie in Git konfiguriert), einschließlich des Datums im Zeitstempelformat und der Abweichung zur Coordinated Universal Time (UTC).
- Informationen zum Committer (wie in Git konfiguriert), einschließlich des Datums im Zeitstempelformat und der Abweichung zur UTC.
- Die ID der Git-Struktur, in der sich der Commit befindet.
- Die Commit-ID des übergeordneten Commits.
- Die Commit-Nachricht.

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
    "commit": {
        "additionalData": "",
        "committer": {
            "date": "1484167798 -0800",
            "name": "Mary Major",
            "email": "mary_major@example.com"
        },
        "author": {
            "date": "1484167798 -0800",
            "name": "Mary Major",
            "email": "mary_major@example.com"
        },
        "treeId": "347a3408EXAMPLE",
        "parents": [
            "4c925148EXAMPLE"
        ],
```

```
        "message": "Fix incorrect variable name"
    }
}
```

So zeigen Sie Informationen zu einem Commit für die Zusammenführung an

1. Führen Sie den Befehl get-merge-commit aus und geben Sie Folgendes an:

   - Einen Commit-Spezifizierer für die Quelle der Zusammenführung (mit der Option --source-commit-specifier).
   - Einen Commit-Spezifizierer für das Ziel der Zusammenführung (mit der Option --destination-commit-specifier).
   - Die Zusammenführungsoption, die Sie verwenden möchten (mit der Option --merge-option).
   - Der Name des Repositorys (mit der Option --repository-name).

   Um beispielsweise Informationen über einen Merge-Commit für den Quell-Branch namens *bugfix-bug1234* mit einem Ziel-Branch namens *main* unter Verwendung der *THREE_WAY_MERGE-Strategie* in einem Repository mit dem Namen anzuzeigen: *MyDemoRepo*

   ```
   aws codecommit get-merge-commit --source-commit-specifier bugfix-bug1234 --
   destination-commit-specifier main --merge-option THREE_WAY_MERGE --repository-
   name MyDemoRepo
   ```

2. Ist der Befehl erfolgreich, gibt seine Ausgabe Informationen zurück, die wie folgt aussehen sollten:

   ```
   {
       "sourceCommitId": "c5709475EXAMPLE",
       "destinationCommitId": "317f8570EXAMPLE",
       "baseCommitId": "fb12a539EXAMPLE",
       "mergeCommitId": "ffc4d608eEXAMPLE"
   }
   ```

So zeigen Sie Informationen zu mehreren Commits an

1. Führen Sie den Befehl batch-get-commits aus und geben Sie Folgendes an:

- Der Name des Repositorys (mit der Option). CodeCommit `--repository-name`
- Eine Liste der vollständigen Commit-IDs für jeden Commit, zu dem Sie Informationen anzeigen möchten.

Zum Beispiel, um Informationen über Commits mit den IDs `317f8570EXAMPLE` und `4c925148EXAMPLE` in einem CodeCommit Repository mit dem Namen `MyDemoRepo` anzuzeigen:

```
aws codecommit batch-get-commits  --repository-name MyDemoRepo  --commit-ids
 317f8570EXAMPLE 4c925148EXAMPLE
```

2. Ist der Befehl erfolgreich, enthält die Ausgabe Folgendes:

- Informationen zu den Autoren des Commits (wie in Git konfiguriert), einschließlich des Datums im Zeitstempelformat und der Abweichung zur Coordinated Universal Time (UTC).
- Informationen zu Committern (wie in Git konfiguriert), einschließlich des Datums im Zeitstempelformat und der Abweichung zur UTC.
- Die IDs der Git-Struktur, in der sich der Commit befindet.
- Die Commit-IDs des übergeordneten Commits.
- Die Commit-Nachrichten.

Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
    "commits": [
      {
        "additionalData": "",
        "committer": {
            "date": "1508280564 -0800",
            "name": "Mary Major",
            "email": "mary_major@example.com"
        },
        "author": {
            "date": "1508280564 -0800",
            "name": "Mary Major",
            "email": "mary_major@example.com"
        },
        "commitId": "317f8570EXAMPLE",
```

```
        "treeId": "1f330709EXAMPLE",
        "parents": [
            "6e147360EXAMPLE"
        ],
        "message": "Change variable name and add new response element"
    },
    {
        "additionalData": "",
        "committer": {
            "date": "1508280542 -0800",
            "name": "Li Juan",
            "email": "li_juan@example.com"
        },
        "author": {
            "date": "1508280542 -0800",
            "name": "Li Juan",
            "email": "li_juan@example.com"
        },
        "commitId": "4c925148EXAMPLE",
        "treeId": "1f330709EXAMPLE",
        "parents": [
            "317f8570EXAMPLE"
        ],
        "message": "Added new class"
    }
}
```

So zeigen Sie Informationen zu den Änderungen eines Commit-Spezifizierers an

1. Führen Sie den Befehl aws codecommit get-differences aus und geben Sie Folgendes an:

   - Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
   - Der Commit-Spezifizierer, zu dem Sie Informationen erhalten möchten. Nur `--after-commit-specifier` ist erforderlich. Wenn Sie `--before-commit-specifier` nicht angeben, werden alle aktuellen Dateien ab `--after-commit-specifier` angezeigt.

   Um beispielsweise Informationen zu den Unterschieden zwischen Commits mit den IDs `317f8570EXAMPLE` und `4c925148EXAMPLE` in einem CodeCommit Repository mit dem Namen `MyDemoRepo` anzuzeigen:

```
aws codecommit get-differences  --repository-name MyDemoRepo  --before-commit-
specifier 317f8570EXAMPLE --after-commit-specifier 4c925148EXAMPLE
```

2. Ist der Befehl erfolgreich, enthält die Ausgabe Folgendes:

   - Eine Liste der Unterschiede, einschließlich Änderungstyp: "A" für Added (hinzugefügt), "D" für Deleted (gelöscht) oder "M" für Modified (geändert).
   - Der Modus des Dateiänderungstyps.
   - Die ID des Git-Blobs, das die Änderung enthält.

   Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

```
{
    "differences": [
        {
            "afterBlob": {
                "path": "blob.txt",
                "blobId": "2eb4af3bEXAMPLE",
                "mode": "100644"
            },
            "changeType": "M",
            "beforeBlob": {
                "path": "blob.txt",
                "blobId": "bf7fcf28fEXAMPLE",
                "mode": "100644"
            }
        }
    ]
}
```

So zeigen Sie Informationen zu einem Git-Blob an

1. Führen Sie den Befehl aws codecommit get-blob aus und geben Sie Folgendes an:

   - Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
   - Die ID des Git-Blobs (mit der Option `--blob-id`).

Um beispielsweise Informationen über einen Git-Blob mit der ID von `2eb4af3bEXAMPLE` in einem CodeCommit Repository mit dem Namen `MyDemoRepo` anzuzeigen:

```
aws codecommit get-blob  --repository-name MyDemoRepo  --blob-id 2eb4af3bEXAMPLE
```

2. Ist der Befehl erfolgreich, enthält die Ausgabe Folgendes:

   - Der base64-kodierte Inhalt des Blobs ist in der Regel eine Datei.

   Beispielsweise könnte die Ausgabe des vorherigen Befehls etwa so aussehen:

```
{
    "content": "QSBCaW5hcnkgTGFyToEXAMPLE="
}
```

Commit-Details anzeigen (Git)

Bevor Sie diese Schritte ausführen, sollten Sie das lokale Repository bereits mit dem CodeCommit Repository verbunden und die Änderungen übernommen haben. Anweisungen finden Sie unter Herstellen einer Verbindung mit einem Repository.

Um die Änderungen für den letzten Commit in ein Repository anzuzeigen, führen Sie den git show Befehl aus.

```
git show
```

Die Ausgabe des Befehls ähnelt der folgenden:

```
commit 4f8c6f9d
Author: Mary Major <mary.major@example.com>
Date:   Mon May 23 15:56:48 2016 -0700

    Added bumblebee.txt

diff --git a/bumblebee.txt b/bumblebee.txt
new file mode 100644
index 0000000..443b974
--- /dev/null
```

```
+++ b/bumblebee.txt
@@ -0,0 +1 @@
+A bumblebee, also written bumble bee, is a member of the bee genus Bombus, in the
 family Apidae.
\ No newline at end of file
```

> **Note**
>
> In diesem und den folgenden Beispielen wurden die Commit-IDs gekürzt. Die vollständigen Commit-IDs werden nicht angezeigt.

Sie können den Befehl git show mit der Commit-ID verwenden, um die vorgenommenen Änderungen anzuzeigen:

```
git show 94ba1e60

commit 94ba1e60
Author: John Doe <johndoe@example.com>
Date:   Mon May 23 15:39:14 2016 -0700

    Added horse.txt

diff --git a/horse.txt b/horse.txt
new file mode 100644
index 0000000..080f68f
--- /dev/null
+++ b/horse.txt
@@ -0,0 +1 @@
+The horse (Equus ferus caballus) is one of two extant subspecies of Equus ferus.
```

Um die Unterschiede zwischen zwei Commits zu sehen, führen Sie den git diff Befehl aus und geben Sie die beiden Commit-IDs an.

```
git diff ce22850d 4f8c6f9d
```

In diesem Beispiel besteht der Unterschied zwischen den Commits darin, dass zwei Dateien hinzugefügt wurden. Die Ausgabe des Befehls ähnelt der folgenden:

```
diff --git a/bees.txt b/bees.txt
```

```
new file mode 100644
index 0000000..cf57550
--- /dev/null
+++ b/bees.txt
@@ -0,0 +1 @@
+Bees are flying insects closely related to wasps and ants, and are known for their
 role in pollination and for producing honey and beeswax.
diff --git a/bumblebee.txt b/bumblebee.txt
new file mode 100644
index 0000000..443b974
--- /dev/null
+++ b/bumblebee.txt
@@ -0,0 +1 @@
+A bumblebee, also written bumble bee, is a member of the bee genus Bombus, in the
 family Apidae.
\ No newline at end of file
```

Um Git zu verwenden, um Details zu den Commits in einem lokalen Repo anzuzeigen, führe den git log folgenden Befehl aus:

```
git log
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
commit 94ba1e60
Author: John Doe <johndoe@example.com>
Date:   Mon May 23 15:39:14 2016 -0700

    Added horse.txt

commit 4c925148
Author: Jane Doe <janedoe@example.com>
Date:   Mon May 22 14:54:55 2014 -0700

    Added cat.txt and dog.txt
```

Führen Sie den Befehl git log --pretty=oneline aus, um nur Commit-IDs und Nachrichten anzuzeigen:

```
git log --pretty=oneline
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
94ba1e60 Added horse.txt
4c925148 Added cat.txt and dog.txt
```

Weitere Optionen findest du in deiner Git-Dokumentation.

Vergleiche Commits in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um die Unterschiede zwischen Commit-Spezifizierern in einem CodeCommit Repository anzuzeigen. Sie können den Unterschied zwischen einem Commit und dem übergeordneten Commit auf einen Blick erkennen. Zudem können Sie zwei beliebige Referenzen, darunter auch Commit-IDs, vergleichen.

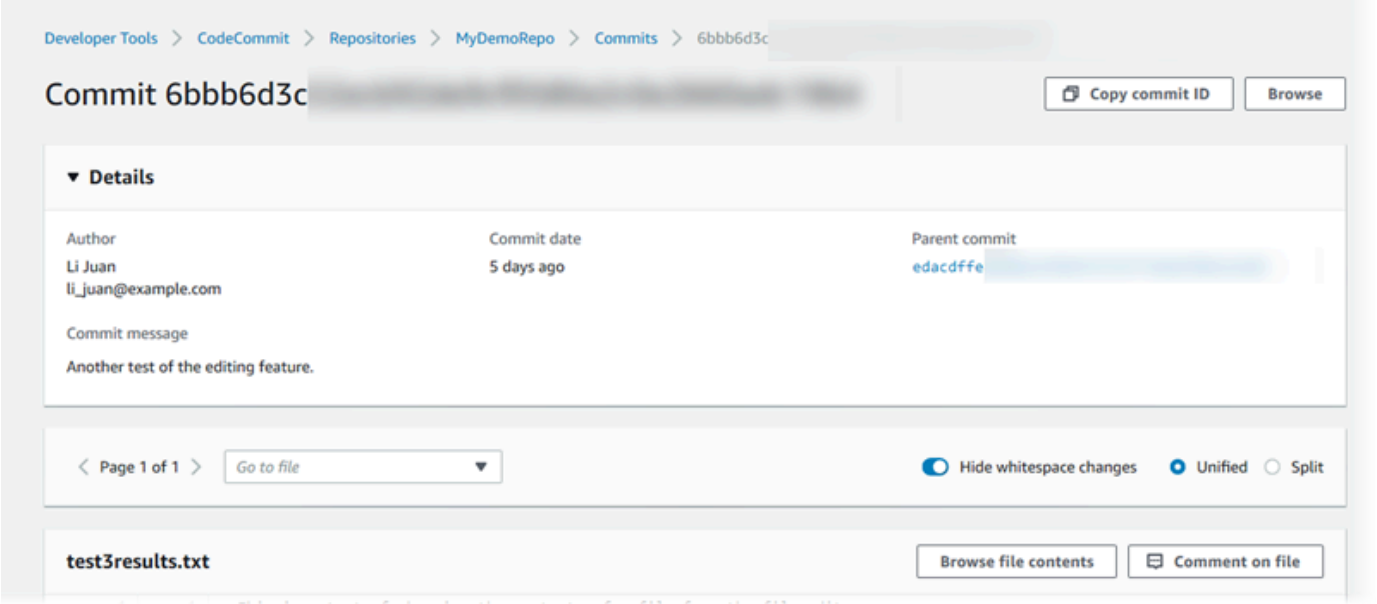
Themen

- Einen Commit mit seinem übergeordneten Commit vergleichen
- Vergleichen von zwei beliebigen Commit-Spezifizierern

Einen Commit mit seinem übergeordneten Commit vergleichen

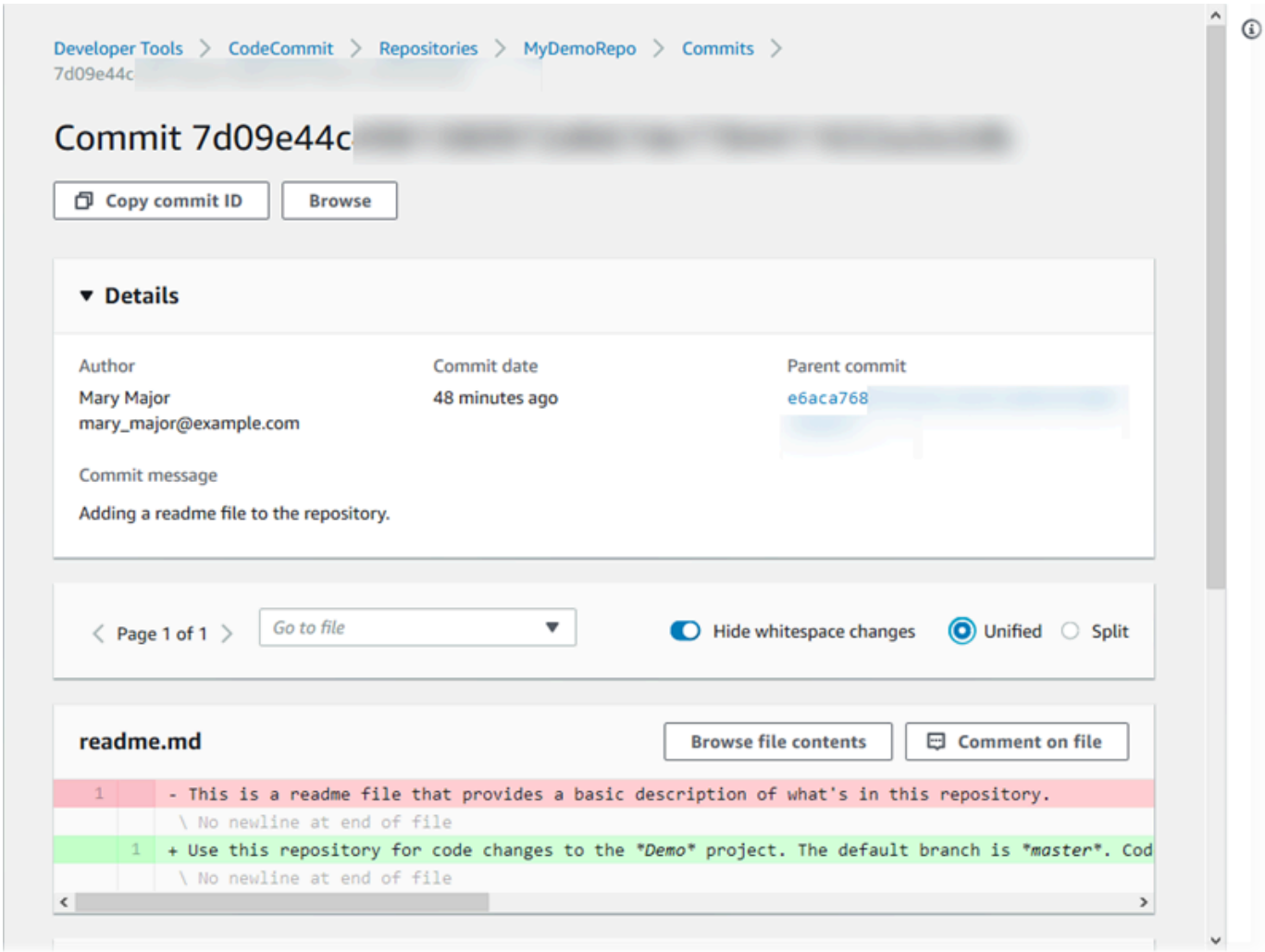
Sie können den Unterschied zwischen einem Commit und dem übergeordneten Commit auf einen Blick erkennen und die Commit-Nachricht, den Committer und die Änderung prüfen.

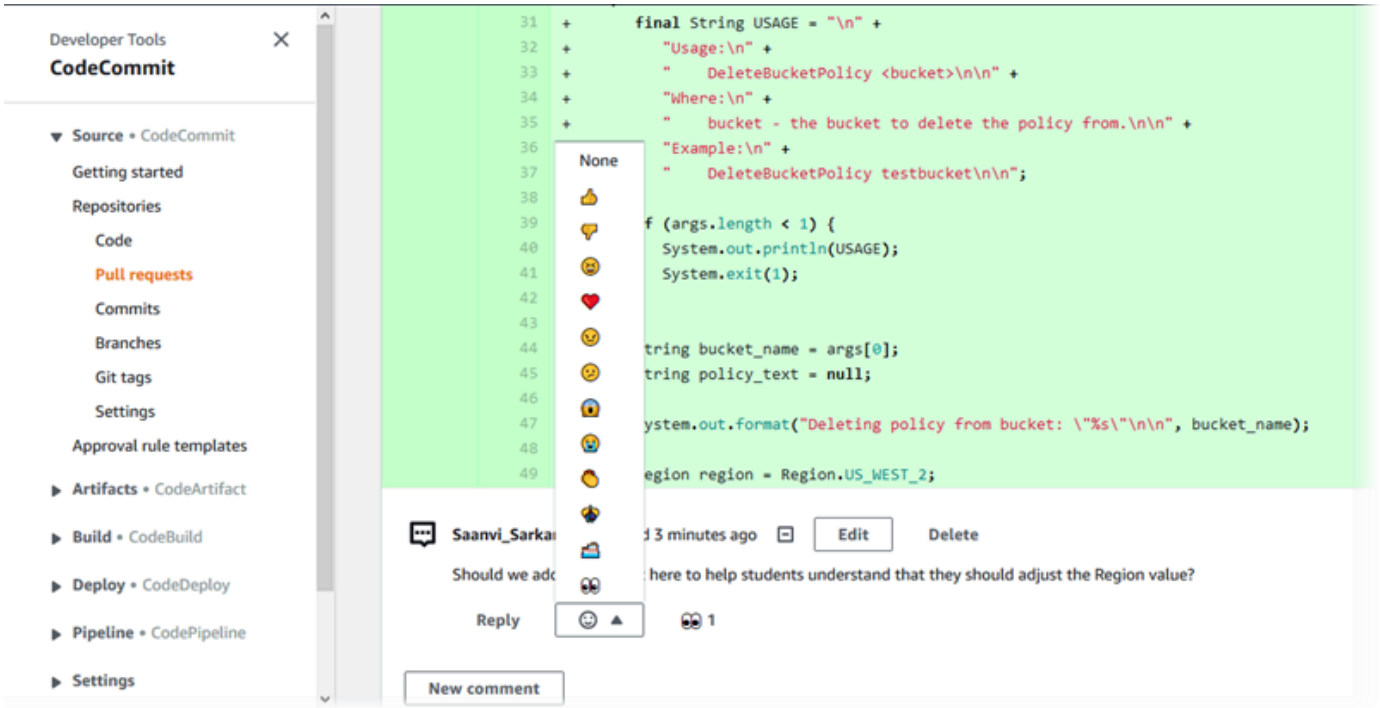
1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie auf der Seite Repositories (Repositorys) das Repository aus, bei dem Sie den Unterschied zwischen einem Commit und dem übergeordneten Commit anzeigen möchten.
3. Wählen Sie im Navigationsbereich Commits aus.
4. Wählen Sie die gekürzte Commit-ID eines beliebigen Commits in der Liste aus. Die Ansicht wird geändert und zeigt nun Details zu diesem Commit an, darunter auch die Unterschiede zum übergeordneten Commit.



Sie können die Änderungen nebeneinander (Ansicht Split) oder inline (Ansicht Unified) anzeigen. Außerdem können Sie geänderte Leerzeichen ein- oder ausblenden. Sie können auch Kommentare hinzufügen. Weitere Informationen finden Sie unter Kommentar zu einem Commit.

> **Note**
>
> Ihre Präferenzen für die Anzeige des Codes sowie andere Konsoleneinstellungen werden als Browser-Cookies gespeichert, wenn Sie sie ändern. Weitere Informationen finden Sie unter Arbeiten mit Benutzereinstellungen.


Developer Tools > CodeCommit > Repositories > MyDemoRepo > Commits >
7d09e44c
Commit 7d09e44c
Copy commit ID
Browse
Details
Author
Mary Major
mary_major@example.com
Commit date
48 minutes ago
Parent commit
e6aca768
Commit message
Adding a readme file to the repository.
Page 1 of 1
Go to file
Hide whitespace changes
Unified
Split
readme.md
Browse file contents
Comment on file
1 - This is a readme file that provides a basic description of what's in this repository.
\ No newline at end of file
1 + Use this repository for code changes to the *Demo* project. The default branch is *master*. Cod
\ No newline at end of file



> **Note**
>
> Abhängig von der Art der Zeilenenden, Ihrem Codeeditor und weiteren Faktoren sehen Sie möglicherweise nicht die spezifischen Änderungen in einer Zeile, sondern ganze hinzugefügte oder gelöschte Zeilen. Die Details entsprechen dem, was der Befehl git show oder git diff zurückgibt.

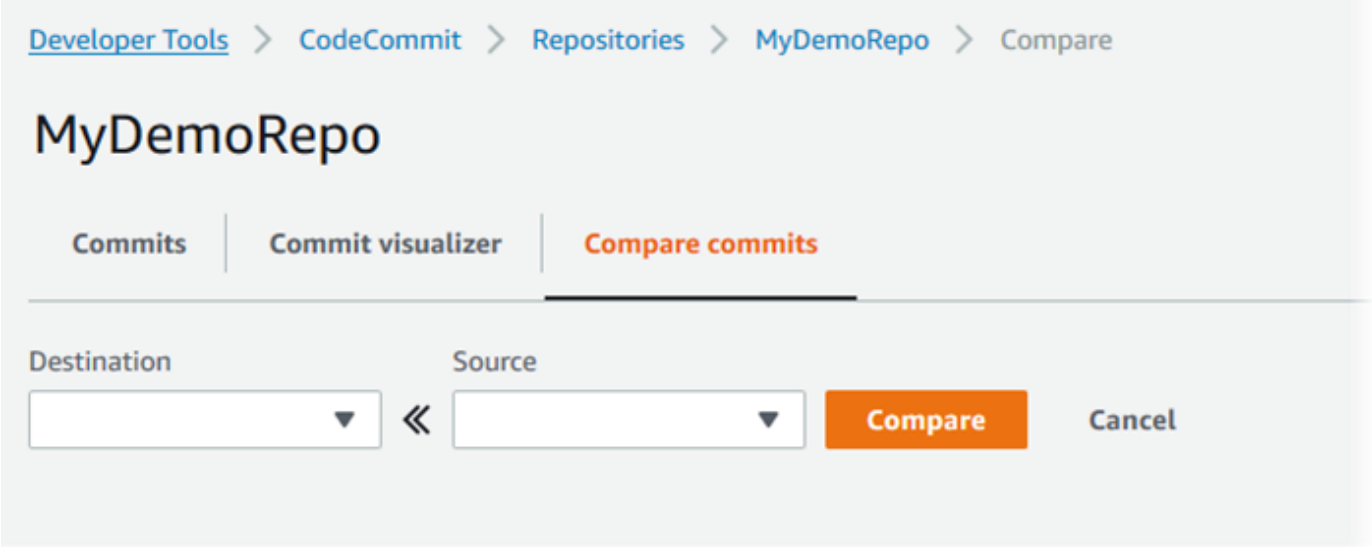
5. Für den Vergleich eines Commits mit dem übergeordneten Commit von der Commit Visualizer-Registerkarte wählen Sie die abgekürzte Commit-ID. Die Commit-Details werden angezeigt, einschließlich der Änderungen zwischen dem Commit und dem übergeordneten Commit.

Vergleichen von zwei beliebigen Commit-Spezifizierern

Sie können die Unterschiede zwischen zwei beliebigen Commit-Spezifizierern in der CodeCommit Konsole einsehen. Commit-Spezifizierer sind Referenzen, wie z. B. Branches, Tags und Commit-IDs.

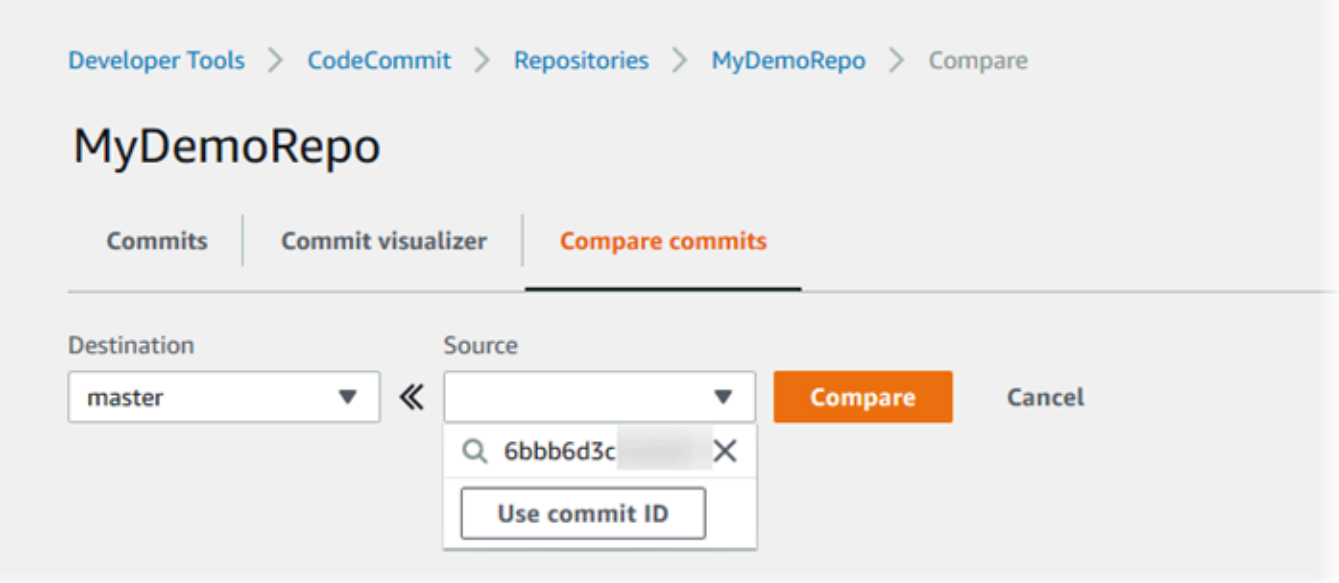
1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie auf der Seite Repositories (Repositorys) das Repository aus, bei dem Sie Commits, Branches oder Commits mit Tags vergleichen möchten.

3. Wählen Sie im Navigationsbereich Commits und dann Compare commits (Commits vergleichen) aus.

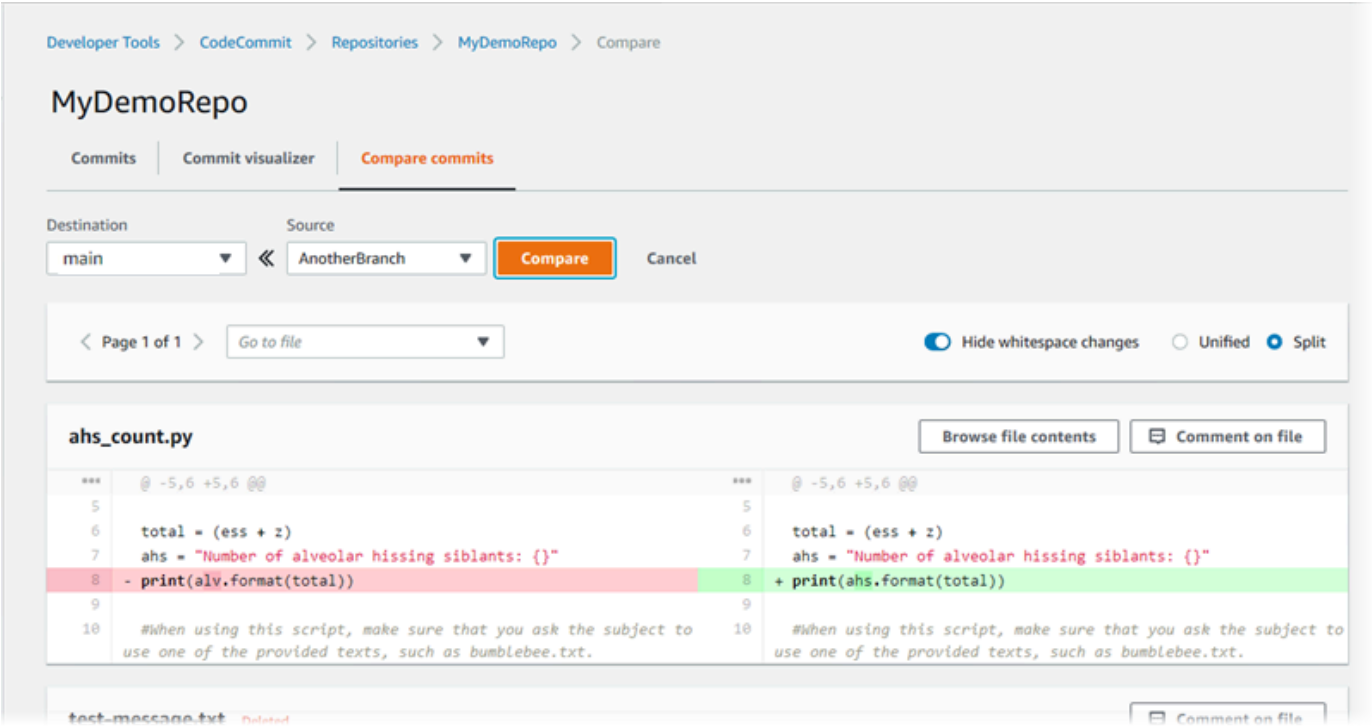


4. Verwenden Sie die Felder, um zwei Commit-Spezifizierer zu vergleichen.

   - Wählen Sie den Branch-Namen aus der Liste aus, um den obersten Commit vom Branch zu vergleichen. Damit wird der neueste Commit von diesem Branch für den Vergleich ausgewählt.
   - Wählen Sie den Tag-Namen aus der Liste aus, sofern vorhanden, um einen Commit mit einem spezifischen zugeordneten Tag zu vergleichen. Damit wird der Commit mit Tag für den Vergleich ausgewählt.
   - Geben oder fügen Sie die Commit-ID in das Feld ein, um einen bestimmten Commit zu vergleichen. Für die vollständige Commit-ID wählen Sie Commits in der Navigationsleiste aus, dann kopieren Sie die Commit-ID aus der Liste. Fügen Sie auf der Seite Compare commits (Commits vergleichen) die vollständige Commit-ID in das Textfeld ein und wählen Sie Use commit ID (Commit-ID verwenden) aus.



5. Wenn Sie die Spezifizierer ausgewählt haben, wählen Sie Compare aus.



Sie können die Unterschiede nebeneinander (Ansicht Split) oder inline (Ansicht Unified) anzeigen. Außerdem können Sie geänderte Leerzeichen ein- oder ausblenden.

6. Um die Auswahl für den Vergleich zu löschen, wählen Sie Cancel (Abbrechen) aus.

Kommentieren Sie einen Commit in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um Commits in einem Repository zu kommentieren und Kommentare anderer Benutzer zu Commits anzusehen und darauf zu antworten. Auf diese Weise können Sie Änderungen an einem Repository diskutiert werden, unter anderem:

- Warum Änderungen vorgenommen wurden.
- Ob weitere Änderungen erforderlich sind.
- Ob Änderungen mit einem anderen Branch zusammengeführt werden sollen.

Sie können einen allgemeinen Commit, eine Datei in einem Commit oder eine bestimmte Zeile oder Änderung in einer Datei kommentieren. Sie können auch auf eine Codezeile verlinken, indem Sie die Zeile auswählen und dann die resultierende URL in Ihren Browser kopieren.

> **Note**
>
> Die besten Ergebnisse erzielen Sie, wenn Sie das Kommentieren verwenden, wenn Sie als IAM-Benutzer angemeldet sind. Die Kommentarfunktion ist nicht optimiert für Benutzer, die sich mit Anmeldeinformationen des Root-Kontos, über einen verbundenen Zugriff oder mit temporären Anmeldeinformationen anmelden.

Themen

- Kommentare zu einem Commit in einem Repository anzeigen
- Kommentare zu einem Commit in einem Repository hinzufügen und darauf antworten
- Kommentare ansehen, hinzufügen, aktualisieren und beantworten ()AWS CLI

Kommentare zu einem Commit in einem Repository anzeigen

Sie können die CodeCommit Konsole verwenden, um Kommentare zu einem Commit anzuzeigen.

So zeigen Sie Kommentare zu einem Commit an

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.

2. Wählen Sie unter Repositories (Repositorys) das Repository aus, für das Sie Kommentare zu Commits prüfen möchten.
3. Wählen Sie im Navigationsbereich Commits aus. Wählen Sie die Commit-ID des Commits aus, in dem Sie Kommentare anzeigen möchten.

   Die Seite für den betreffenden Commit wird mit den vorhandenen Kommentaren angezeigt.

Kommentare zu einem Commit in einem Repository hinzufügen und darauf antworten

Sie können die CodeCommit Konsole verwenden, um Kommentare zum Vergleich zwischen einem Commit und einem übergeordneten Commit oder zum Vergleich zwischen zwei angegebenen Commits hinzuzufügen. Du kannst auf Kommentare auch mit Emojis, mit deinen eigenen Kommentaren oder beidem antworten.

Kommentare zu einem Commit hinzufügen und darauf antworten (Konsole)

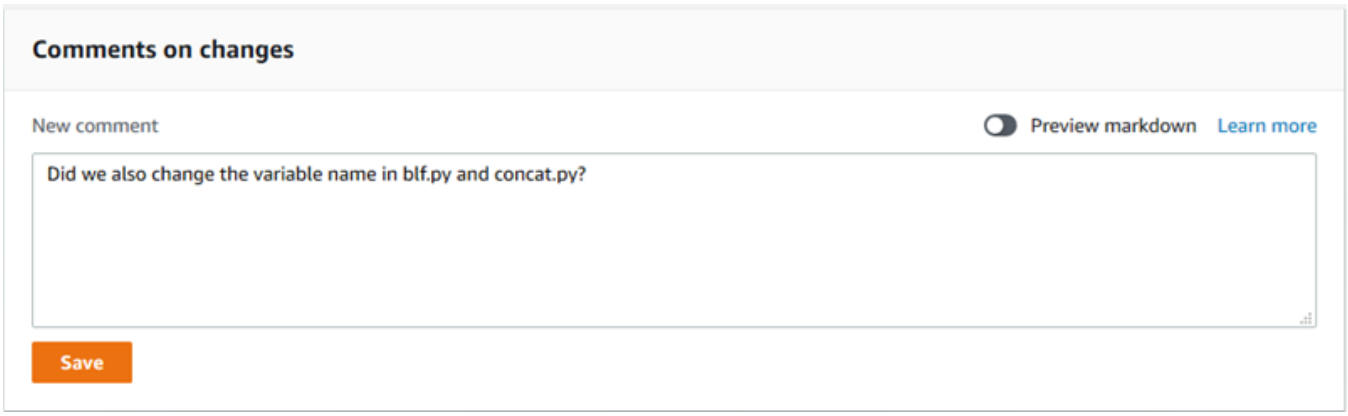
Sie können Kommentare zu einem Commit mit Text und Emojis hinzufügen und darauf antworten. Ihre Kommentare und Emojis sind so gekennzeichnet, dass sie dem IAM-Benutzer oder der IAM-Rolle gehören, mit der Sie sich bei der Konsole angemeldet haben.

So fügen Sie Kommentare zu einem Commit hinzu und antworten darauf

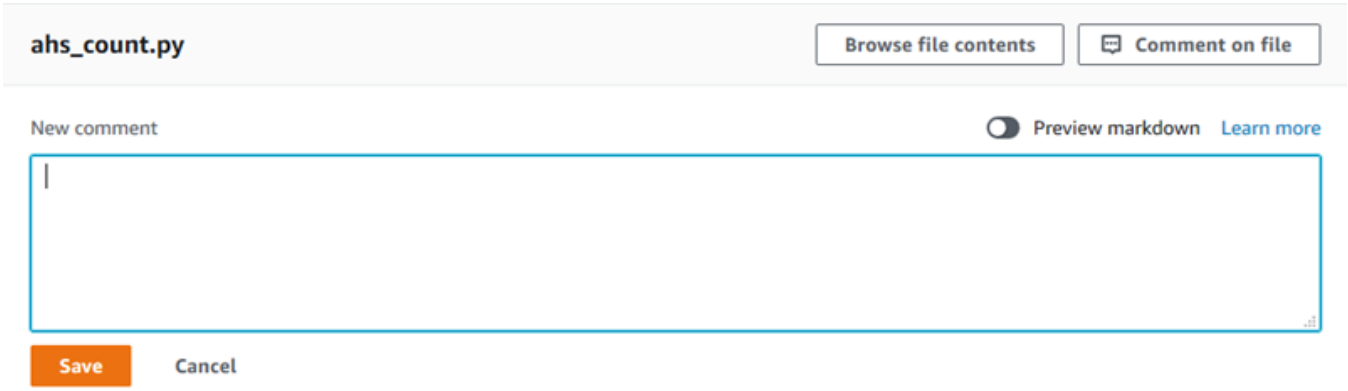
1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie Commits kommentieren möchten.
3. Wählen Sie im Navigationsbereich Commits aus. Wählen Sie die Commit-ID des Commits aus, in dem Sie Kommentare hinzufügen oder beantworten möchten.

   Die Seite für den betreffenden Commit wird mit den vorhandenen Kommentaren angezeigt.
4. Um einen Kommentar hinzuzufügen, gehen Sie wie folgt vor:

   - Zum Hinzufügen eines allgemeinen Kommentars geben Sie in Comments on changes (Kommentare zu Änderungen) einen Kommentar ein und wählen dann Save (Speichern) aus. Sie können Markdown verwenden oder Ihren Kommentar als Klartext eingeben.



- Um einer Datei im Commit einen Kommentar hinzuzufügen, suchen Sie den Namen der Datei. Wählen Sie Comment on file (Kommentar zu Datei) aus, geben Sie Ihren Kommentar ein und wählen Sie dann Save (Speichern) aus.



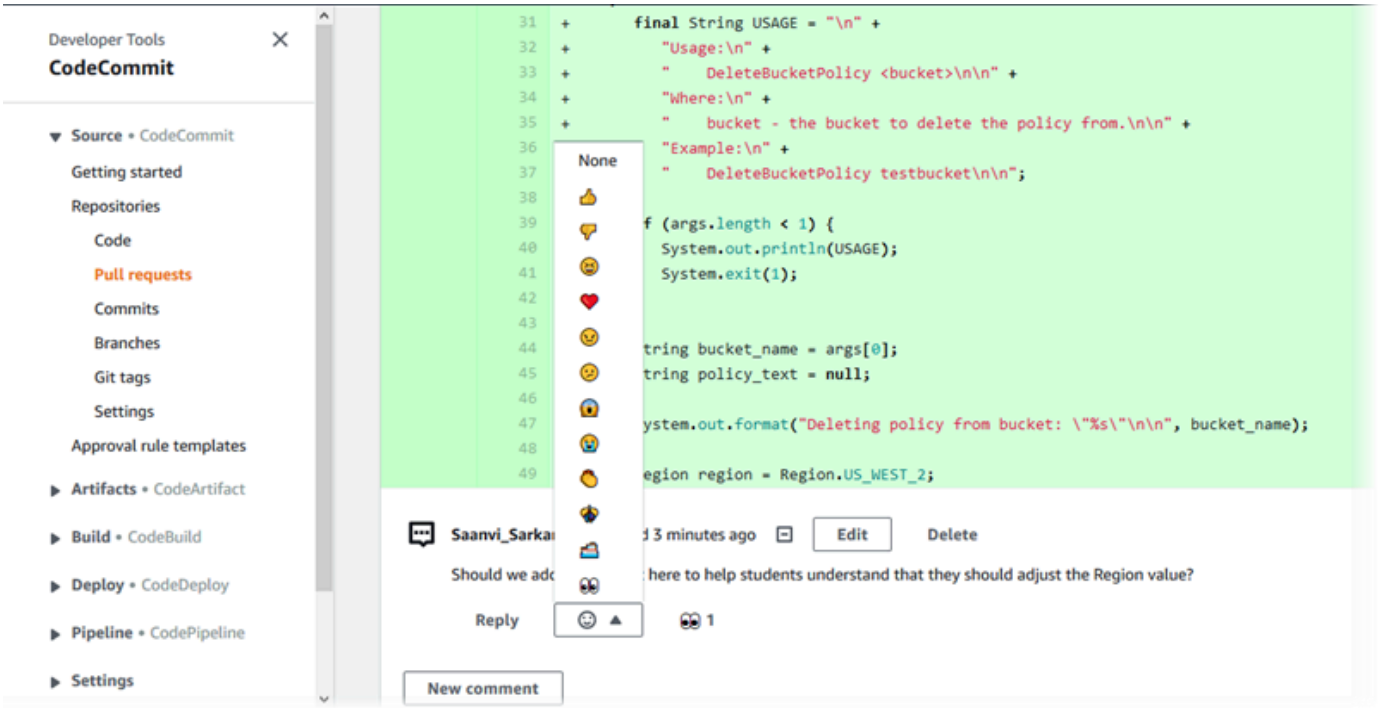
- Um einer geänderten Zeile in dem Commit einen Kommentar hinzuzufügen, suchen Sie die Zeile mit der Änderung. Wählen Sie die Kommentarblase

  aus, geben Sie Ihren Kommentar ein und wählen Sie dann Save (Speichern) aus.

> **Note**
>
> Sie können Ihren Kommentar nach dem Speichern bearbeiten. Sie können den Inhalt auch löschen. Der Kommentar bleibt mit einer Meldung erhalten, dass der Inhalt gelöscht wurde. Verwenden Sie gegebenenfalls den Modus Preview markdown (Markdown-Vorschau) für Ihren Kommentar, bevor Sie ihn speichern.

5. Wählen Sie Reply (Antworten) aus, um Kommentare zu einem Commit zu beantworten. Um auf einen Kommentar mit einem Emoji zu antworten, wählen Sie das gewünschte Emoji aus der Liste aus. Du kannst nur ein Emoji pro Kommentar auswählen. Wenn du deine Emoji-Reaktion ändern möchtest, wähle ein anderes aus der Liste oder wähle Keine, um deine Reaktion zu entfernen.

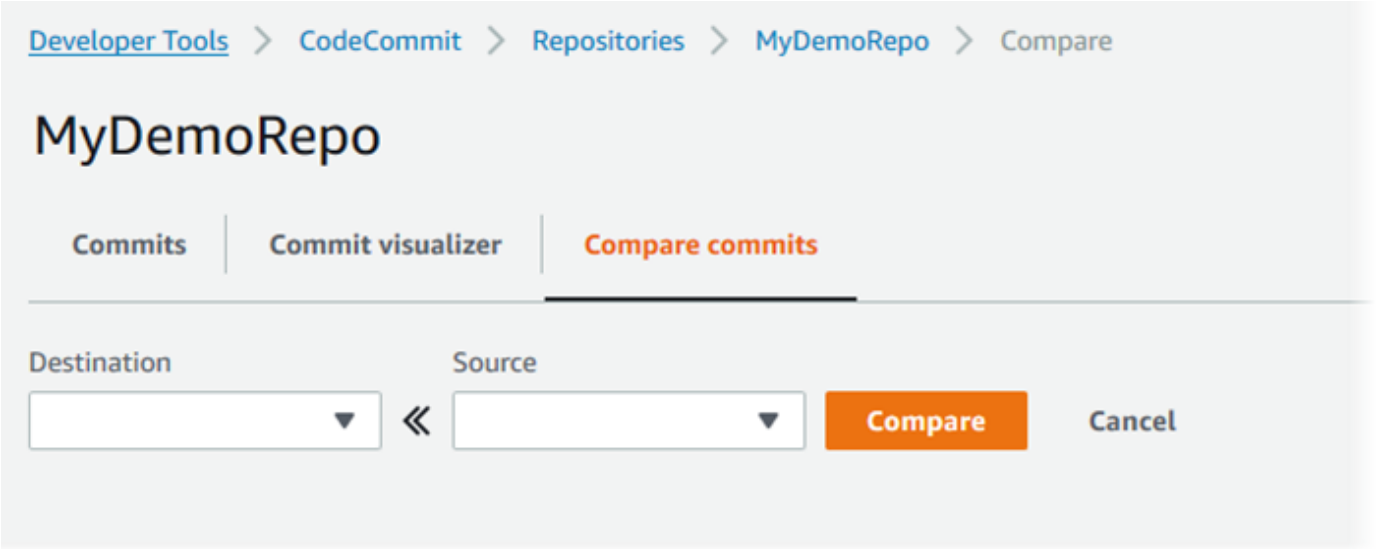


Füge Kommentare hinzu und beantworte sie, wenn du zwei Commit-Spezifizierer vergleichst

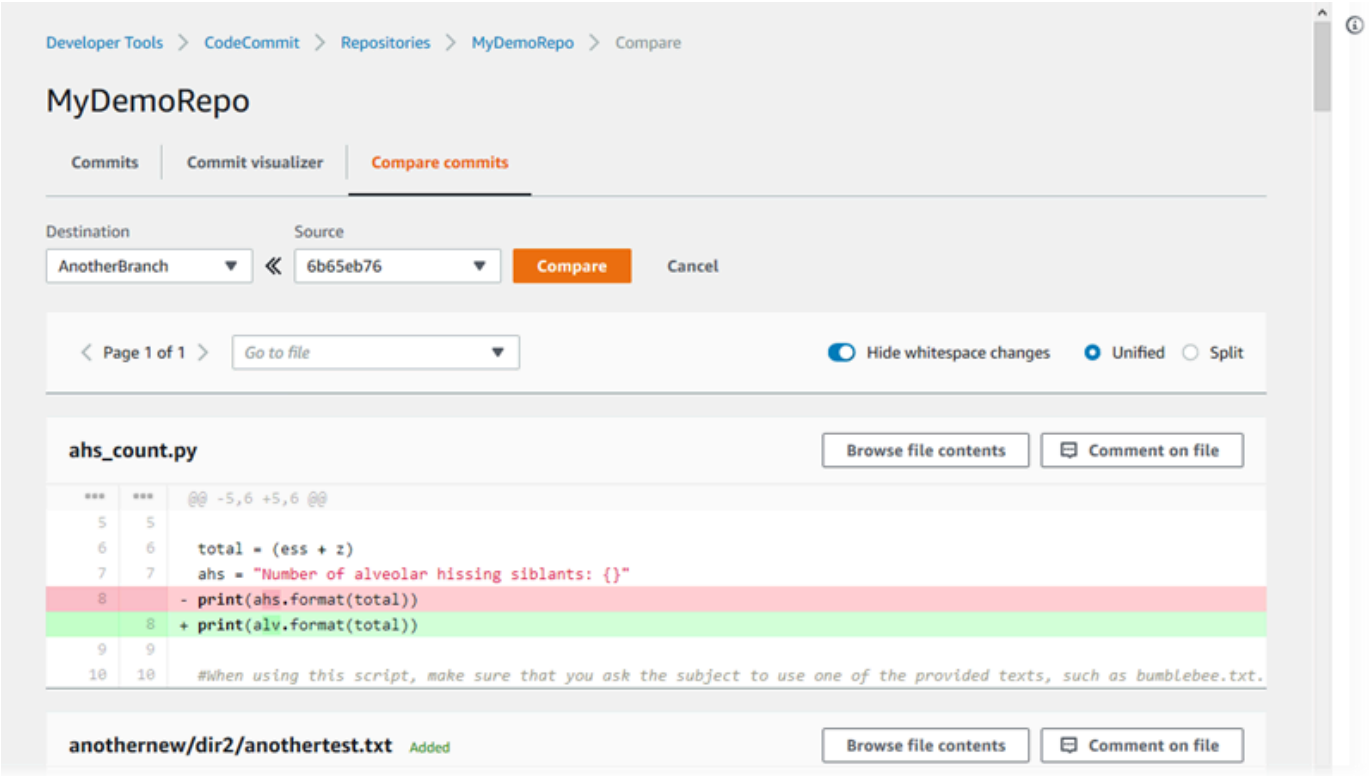
Sie können einem Vergleich zwischen Branches, Tags oder Commits Kommentare hinzufügen.

Beim Vergleich von Commit-Kennungen Kommentare hinzufügen und darauf antworten

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) das Repository aus, in dem Sie Commits, Branches oder Commits mit Tags vergleichen möchten.
3. Wählen Sie im Navigationsbereich Commits und dann die Compare commits (Commits vergleichen)-Registerkarte aus.



4. Verwenden Sie die Felder Ziel und Quelle, um zwei Commit-Kennungen zu vergleichen. Verwenden Sie die Dropdown-Listen oder fügen Sie Commit-IDs ein. Wählen Sie Compare aus.



5. Führen Sie eine oder mehrere der folgenden Aktionen aus:

   - Um Kommentare für Dateien oder Zeilen einzufügen, wählen Sie die Kommentarblase

     aus.

- Um allgemeine Kommentare zu den verglichenen Änderungen hinzuzufügen, wählen Sie Comments on changes.

Kommentare ansehen, hinzufügen, aktualisieren und beantworten ()AWS CLI

Sie können den Inhalt eines Kommentars mit den folgenden Befehlen anzeigen, hinzufügen, beantworten, aktualisieren und löschen:

- Um die Kommentare zum Vergleich zwischen zwei Commits anzuzeigen, führen Sie get-comments-for-compared-commit aus.
- Um Details zu einem bestimmten Kommentar anzuzeigen, führen Sie get-comment aus.
- Um den Inhalt eines Kommentars zu löschen, den Sie erstellt haben, führen Sie delete-comment-content aus.
- Um einen Kommentar zum Vergleich zwischen zwei Commits zu erstellen, führen Sie post-comment-for-compared-commit aus.
- Um einen Kommentar zu aktualisieren, führen Sie update-comment aus.
- Um auf einen Kommentar zu antworten, führen Sie post-comment-replyden Befehl aus.
- Um auf einen Kommentar mit einem Emoji zu antworten, führe put-comment-reactionden Befehl aus.
- Um Emoji-Reaktionen auf einen Kommentar anzuzeigen, führe den Befehl aus. get-comment-reactions

So zeigen Sie Kommentare zu einem Commit an

1. Führen Sie den Befehl get-comments-for-compared-commit aus und geben Sie Folgendes an:
   - Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
   - Die vollständige Commit-ID des AFTER-Commits, um die Richtung des Vergleichs festzulegen (mit der `--after-commit-id option`).
   - Die vollständige Commit-ID des BEFORE-Commits, um die Richtung des Vergleichs festzulegen (mit der `--before-commit-id`-Option).
   - (Optional) Ein Aufzählungs-Token zum Zurückgeben des nächsten Ergebnisstapels (mit der Option `--next-token`).

- (Optional) Eine nicht negative ganze Zahl, um die Anzahl der zurückgegebenen Ergebnisse zu begrenzen (mit der Option `--max-results`).

Zum Beispiel, um Kommentare zu sehen, die zum Vergleich zwischen zwei Commits in einem Repository mit dem Namen *MyDemoRepo*gemacht wurden:

```
aws codecommit get-comments-for-compared-commit --repository-name MyDemoRepo --
before-commit-ID 6e147360EXAMPLE --after-commit-id 317f8570EXAMPLE
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
   "commentsForComparedCommitData": [
      {
         "afterBlobId": "1f330709EXAMPLE",
         "afterCommitId": "317f8570EXAMPLE",
         "beforeBlobId": "80906a4cEXAMPLE",
         "beforeCommitId": "6e147360EXAMPLE",
         "comments": [
            {
               "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
               "clientRequestToken": "123Example",
               "commentId": "ff30b348EXAMPLEb9aa670f",
               "content": "Whoops - I meant to add this comment to the line, not
 the file, but I don't see how to delete it.",
               "creationDate": 1508369768.142,
               "deleted": false,
               "CommentId": "123abc-EXAMPLE",
               "lastModifiedDate": 1508369842.278,
               "callerReactions": [],
               "reactionCounts":
                {
                  "SMILE" : 6,
                  "THUMBSUP" : 1
                }
            },
            {
               "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
               "clientRequestToken": "123Example",
               "commentId": "553b509bEXAMPLE56198325",
               "content": "Can you add a test case for this?",
               "creationDate": 1508369612.240,
```

```
                "deleted": false,
                "commentId": "456def-EXAMPLE",
                "lastModifiedDate": 1508369612.240,
                "callerReactions": [],
                "reactionCounts":
                 {
                   "THUMBSUP" : 2
                 }
             }
         ],
         "location": {
            "filePath": "cl_sample.js",
            "filePosition": 1232,
            "relativeFileVersion": "after"
         },
         "repositoryName": "MyDemoRepo"
      }
   ],
   "nextToken": "exampleToken"
}
```

So zeigen Sie die Details eines Kommentars zu einem Commit an

1. Führen Sie den Befehl get-comment aus und geben Sie die vom System generierte Kommentar-ID an. Beispielsweise:

```
aws codecommit get-comment --comment-id ff30b348EXAMPLEb9aa670f
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
{
   "comment": {
      "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
      "clientRequestToken": "123Example",
      "commentId": "ff30b348EXAMPLEb9aa670f",
      "content": "Whoops - I meant to add this comment to the line, but I don't see
 how to delete it.",
      "creationDate": 1508369768.142,
      "deleted": false,
      "commentId": "",
      "lastModifiedDate": 1508369842.278,
      "callerReactions": [],
```

```
      "reactionCounts":
         {
           "SMILE" : 6,
           "THUMBSUP" : 1
         }
   }
}
```

So löschen Sie den Inhalt eines Kommentars zu einem Commit

1. Führen Sie den Befehl delete-comment-content aus und geben Sie die vom System generierte Kommentar-ID an. Beispielsweise:

```
aws codecommit delete-comment-content --comment-id ff30b348EXAMPLEb9aa670f
```

> **Note**
>
> Sie können den Inhalt eines Kommentars nur löschen, wenn Sie die AWSCodeCommitFullAccess Richtlinie angewendet haben oder wenn Sie die `DeleteCommentContent` Berechtigung auf Zulassen gesetzt haben.

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
   "comment": {
      "creationDate": 1508369768.142,
      "deleted": true,
      "lastModifiedDate": 1508369842.278,
      "clientRequestToken": "123Example",
      "commentId": "ff30b348EXAMPLEb9aa670f",
      "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
      "callerReactions": [],
      "reactionCounts":
         {
           "CLAP" : 1
         }
   }
}
```

So erstellen Sie einen Kommentar zu einem Commit

1. Führen Sie den Befehl post-comment-for-compared-commit aus und geben Sie Folgendes an:

   - Der Name des CodeCommit Repositorys (mit der `--repository-name` Option).
   - Die vollständige Commit-ID des AFTER-Commits, um die Richtung des Vergleichs festzulegen (mit der Option `--after-commit-id`).
   - Die vollständige Commit-ID des BEFORE-Commits, um die Richtung des Vergleichs festzulegen (mit der `--before-commit-id`-Option).
   - Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option --client-request-token).
   - Der Inhalt Ihres Kommentars (mit der Option --content).
   - Eine Liste der Speicherortangaben, wo der Kommentar abgelegt werden soll, unter anderem:
     - Der Name der zu vergleichenden Datei, einschließlich der Erweiterung und des Unterverzeichnisses (mit dem Attribut filePath).
     - Die Zeilennummer der Änderung innerhalb einer verglichenen Datei (mit dem Attribut filePosition).
     - Die Angabe, ob der Kommentar zu der Änderung vor oder nach dem Vergleich zwischen den Quell- und Ziel-Branches steht (mit dem Attribut relativeFileVersion).

   Zum Beispiel, um den Kommentar *„Können Sie dafür einen Testfall hinzufügen?“* hinzuzufügen über die Änderung an der Datei *cl_sample.js* beim Vergleich zwischen zwei Commits in einem Repository mit dem Namen *MyDemoRepo*:

```
aws codecommit post-comment-for-compared-commit --repository-name MyDemoRepo
 --before-commit-id 317f8570EXAMPLE --after-commit-id 5d036259EXAMPLE --client-
request-token 123Example --content "Can you add a test case for this?" --location
 filePath=cl_sample.js,filePosition=1232,relativeFileVersion=AFTER
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
        "afterBlobId": "1f330709EXAMPLE",
        "afterCommitId": "317f8570EXAMPLE",
        "beforeBlobId": "80906a4cEXAMPLE",
        "beforeCommitId": "6e147360EXAMPLE",
        "comment": {
              "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
```

```
            "clientRequestToken": "",
            "commentId": "553b509bEXAMPLE56198325",
            "content": "Can you add a test case for this?",
            "creationDate": 1508369612.203,
            "deleted": false,
            "commentId": "abc123-EXAMPLE",
            "lastModifiedDate": 1508369612.203,
            "callerReactions": [],
            "reactionCounts": []
          },
          "location": {
            "filePath": "cl_sample.js",
            "filePosition": 1232,
            "relativeFileVersion": "AFTER"
          },
        "repositoryName": "MyDemoRepo"
}
```

So aktualisieren Sie einen Kommentar zu einem Commit

1. Führen Sie den Befehl update-comment aus und geben Sie die vom System generierte Kommentar-ID und den Inhalt an, mit dem Sie vorhandenen Inhalt ersetzen möchten.

   Zum Beispiel, um den Inhalt *„Fixed as request“ hinzuzufügen. Ich werde die Pull-Anfrage aktualisieren.“* zu einem Kommentar mit der ID *442b498BExample5756813*:

   ```
   aws codecommit update-comment --comment-id 442b498bEXAMPLE5756813 --content "Fixed
    as requested. I'll update the pull request."
   ```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "comment": {
           "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
           "clientRequestToken": "",
           "commentId": "442b498bEXAMPLE5756813",
           "content": "Fixed as requested. I'll update the pull request.",
           "creationDate": 1508369929.783,
           "deleted": false,
           "lastModifiedDate": 1508369929.287,
   ```

```
        "callerReactions": [],
        "reactionCounts":
          {
            "THUMBSUP" : 2
          }
    }
}
```

So antworten Sie auf einen Kommentar zu einem Commit

1. Um eine Antwort auf einen Kommentar in einer Pull-Anforderung hinzuzufügen, führen Sie den Befehl post-comment-reply aus und geben Folgendes an:

   - Die vom System generierte ID des Kommentars, auf den Sie antworten wollen (mit der Option --in-reply-to).
   - Einzigartiger, vom Client generierter Idempotenz-Token (mit der Option --client-request-token).
   - Der Inhalt Ihrer Antwort (mit der Option --content).

   Zum Beispiel, um die Antwort *„Guter catch“ hinzuzufügen. Ich werde sie entfernen.“ zum Kommentar mit der vom System generierten ID ABCD1234ExampleB5678EFGH:*

   ```
   aws codecommit post-comment-reply --in-reply-to abcd1234EXAMPLEb5678efgh --
   content "Good catch. I'll remove them." --client-request-token 123Example
   ```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "comment": {
           "authorArn": "arn:aws:iam::111111111111:user/Li_Juan",
           "clientRequestToken": "123Example",
           "commentId": "442b498bEXAMPLE5756813",
           "content": "Good catch. I'll remove them.",
           "creationDate": 1508369829.136,
           "deleted": false,
           "CommentId": "abcd1234EXAMPLEb5678efgh",
           "lastModifiedDate": 150836912.221,
           "callerReactions": [],
           "reactionCounts": []
   ```

```
    }
}
```

Um auf einen Kommentar zu einem Commit mit einem Emoji zu antworten

1. Um auf einen Kommentar in einer Pull-Anfrage mit einem Emoji zu antworten oder den Wert deiner Emoji-Reaktion zu ändern, führe den put-comment-reaction Befehl aus und gib Folgendes an:

   - Die vom System generierte ID des Kommentars, auf den Sie mit einem Emoji antworten möchten.
   - Der Wert der Reaktion, die Sie hinzufügen oder aktualisieren möchten. Zu den akzeptablen Werten gehören unterstützte Emojis, Shortcodes und Unicode-Werte.

   Die folgenden Werte werden für Emojis in unterstützt: CodeCommit

Emoji	Shortcode	Unicode
#	:thumbsup:	U+1F44D
#	:thumbsdown:	U+1F44E
#	:smile:	U+1F604
❤	:heart:	U+2764
#	:angry:	U+1F620
#	:confused:	U+1F615
#	:scream:	U+1F631
#	:sob:	U+1F62D
#	:clap:	U+1F44F
#	:confetti_ball:	U+1F38A

Emoji	Shortcode	Unicode
#	:ship:	U+1F6A2
#	:eyes:	U+1F440
	Keine	U+0000

Um zum Beispiel das Emoji:thumbsup: zum Kommentar mit der vom System generierten ID ABCD1234ExampleB5678EFGH hinzuzufügen:

```
aws codecommit put-comment-reaction --comment-id abcd1234EXAMPLEb5678efgh --
reaction-value :thumbsup:
```

2. Bei Erfolg erzeugt dieser Befehl keine Ausgabe.

Um Emoji-Reaktionen auf einen Kommentar zu sehen

1. Um Emoji-Reaktionen auf einen Kommentar anzuzeigen, einschließlich der Benutzer, die mit diesen Emojis reagiert haben, führen Sie den get-comment-reactions Befehl aus und geben Sie die vom System generierte ID des Kommentars an.

   Um beispielsweise Emoji-Reaktionen auf den Kommentar mit der vom System generierten ID ABCD1234ExampleB5678EFGH anzuzeigen:

   ```
   aws codecommit get-comment-reactions --comment-id abcd1234EXAMPLEb5678efgh
   ```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "reactionsForComment": {
           [
              {
                   "reaction": {
                       "emoji:"#",
                       "shortCode": "thumbsup",
                       "unicode": "U+1F44D"
                   },
                   "users": [
   ```

```
                    "arn:aws:iam::123456789012:user/Li_Juan",
                    "arn:aws:iam::123456789012:user/Mary_Major",
                    "arn:aws:iam::123456789012:user/Jorge_Souza"
                ]
            },
            {
                "reaction": {
                    "emoji": "#",
                    "shortCode": "thumbsdown",
                    "unicode": "U+1F44E"
                },
                "users": [
                    "arn:aws:iam::123456789012:user/Nikhil_Jayashankar"
                ]
            },
            {
                "reaction": {
                    "emoji": "#",
                    "shortCode": "confused",
                    "unicode": "U+1F615"
                },
                "users": [
                    "arn:aws:iam::123456789012:user/Saanvi_Sarkar"
                ]
            }
        ]
    }
}
```

Erstellen Sie ein Git-Tag in AWS CodeCommit

Sie können ein Git-Tag verwenden, um ein Commit mit einer Kennung zu versehen, damit andere Repository-Benutzer dessen Bedeutung verstehen. Um ein Git-Tag in einem CodeCommit Repository zu erstellen, können Sie Git von einem lokalen Repo aus verwenden, das mit dem CodeCommit Repository verbunden ist. Nachdem Sie ein Git-Tag im lokalen Repo erstellt haben, können Sie es verwenden, git push --tags um es in das CodeCommit Repository zu übertragen.

Weitere Informationen finden Sie unter Tag-Details anzeigen.

Verwende Git, um ein Tag zu erstellen

Gehen Sie wie folgt vor, um mit Git aus einem lokalen Repository ein Git-Tag in einem CodeCommit Repository zu erstellen.

In diesen Schritten gehen wir davon aus, dass Sie das lokale Repo bereits mit dem Repository verbunden haben. CodeCommit Anweisungen finden Sie unter Herstellen einer Verbindung mit einem Repository.

1. Führen Sie den git tag ***new-tag-name commit-id*** Befehl aus, wobei *new-tag-name*der Name des neuen Git-Tags und *commit-ID die ID* des Commits ist, der dem Git-Tag zugeordnet werden soll.

   Beispielsweise wird mit dem folgenden Befehl das neue Git-Tag `beta` erstellt und mit der Commit-ID `dc082f9a...af873b88` verknüpft:

   ```
   git tag beta dc082f9a...af873b88
   ```

2. Um das neue Git-Tag vom lokalen Repository in das CodeCommit Repository zu übertragen, führen Sie den git push ***remote-name new-tag-name*** Befehl aus, wobei *remote-name der Name* des CodeCommit Repositorys und der Name des neuen Git-Tags *new-tag-name*ist.

   Um beispielsweise ein neues Git-Tag mit dem Namen in ein CodeCommit Repository mit dem Namen `beta` zu übertragen`origin`:

   ```
   git push origin beta
   ```

> **Note**
>
> Um alle neuen Git-Tags von deinem lokalen Repo in das CodeCommit Repository zu übertragen, führe git push --tags den Befehl aus.
> Um sicherzustellen, dass Ihr lokales Repo mit allen Git-Tags im CodeCommit Repository aktualisiert wird, führen Sie den Befehl git fetch gefolgt von git fetch --tags aus.

Weitere Optionen findest du in deiner Git-Dokumentation.

Git-Tag-Details anzeigen in AWS CodeCommit

In Git ist ein Tag ist eine Bezeichnung für eine Referenz wie ein Commit, um es mit Informationen zu kennzeichnen, die möglicherweise wichtig für andere Repository-Benutzer sind. Sie können beispielsweise das Commit, das der Beta-Release-Punkt für ein Projekt war, mit dem Tag **`beta`** kennzeichnen. Weitere Informationen finden Sie unter Verwende Git, um ein Tag zu erstellen. Git-Tags unterscheiden sich von Repository-Tags. Weitere Informationen zur Verwendung von Repository-Tags finden Sie unter Hinzufügen eines Tags zu einem Repository.

Sie können die AWS CodeCommit Konsole verwenden, um Informationen zu Git-Tags in Ihrem Repository anzuzeigen, einschließlich des Datums und der Commit-Nachricht des Commits, auf den jedes Git-Tag verweist. In der Konsole können Sie das Commit, auf das das Tag verweist, mit dem Kopf des Standard-Branch Ihres Repositorys vergleichen. Wie bei jedem anderen Commit können Sie auch den Code am Punkt dieses Git-Tags anzeigen.

Sie können Git auch von Ihrem Terminal oder der Befehlszeile aus verwenden, um Details zu Git-Tags in einem lokalen Repository anzuzeigen.

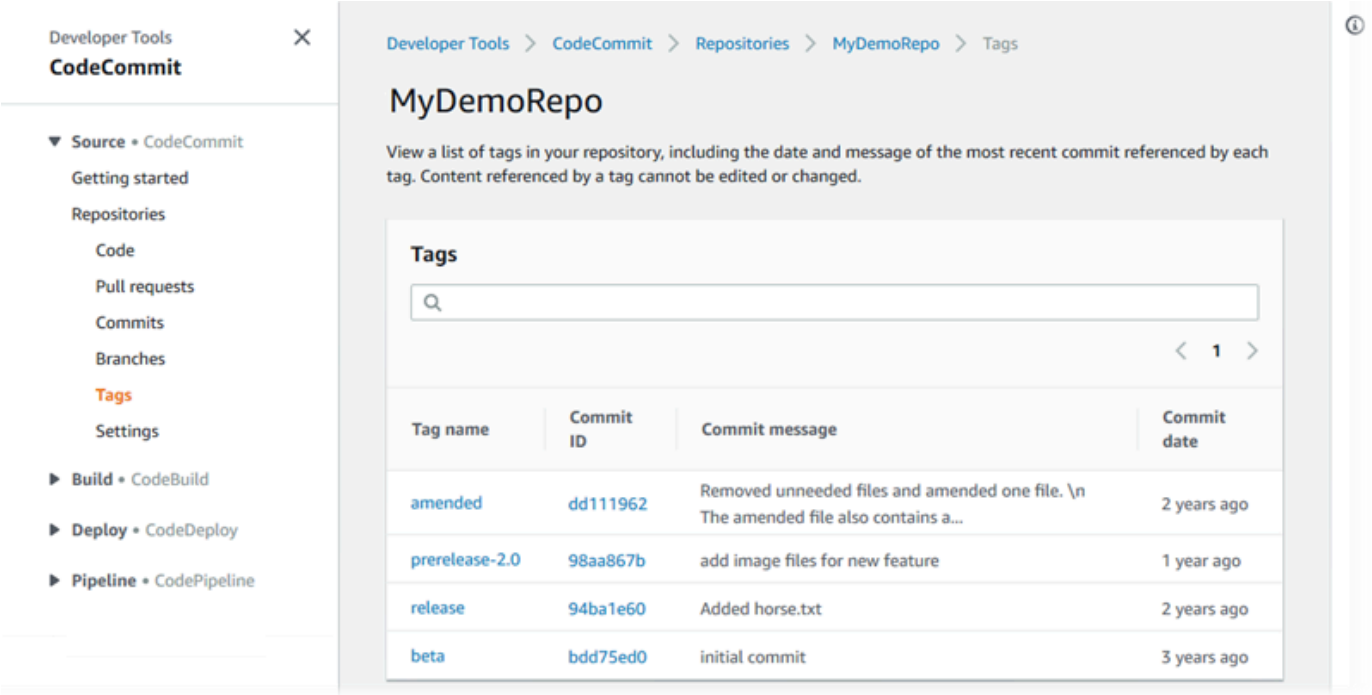
Themen

- Tag-Details anzeigen (Konsole)
- Git-Tag-Details anzeigen (Git)

Tag-Details anzeigen (Konsole)

Verwende die AWS CodeCommit Konsole, um schnell eine Liste der Git-Tags für dein Repository und Details zu den Commits anzuzeigen, auf die in den Git-Tags verwiesen wird.

1. Öffne die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Tags anzeigen möchten.
3. Wählen Sie im Navigationsbereich die Option Git tags (Git-Tags) aus.



4. Führen Sie eine der folgenden Aktionen aus:

   - Zum Anzeigen des Codes, wie er an diesem Commit war, wählen Sie den Git-Tag-Namen aus.
   - Um Details des Commits anzuzeigen, einschließlich der vollständigen Commit-Nachricht, des Committers und des Autors, wählen Sie die gekürzte Commit-ID aus.

Git-Tag-Details anzeigen (Git)

Um Git zu verwenden, um Details zu Git-Tags in einem lokalen Repository anzuzeigen, führen Sie einen der folgenden Befehle aus:

- git tag, um eine Liste der Git-Tag-Namen zu erhalten.
- git show, um Informationen über ein bestimmtes Git-Tag anzuzeigen.
- Git ls-remote, um Informationen über Git-Tags in einem CodeCommit Repository anzuzeigen.

> **Note**
>
> Um sicherzustellen, dass Ihr lokales Repo mit allen Git-Tags im CodeCommit Repository aktualisiert wird, führen Sie den Befehl git fetch gefolgt von git fetch --tags aus.

In den folgenden Schritten gehen wir davon aus, dass Sie das lokale Repo bereits mit einem CodeCommit Repository verbunden haben. Anweisungen finden Sie unter Herstellen einer Verbindung mit einem Repository.

Um eine Liste von Git-Tags in einem lokalen Repo anzuzeigen

1. Führen Sie den Befehl git tag aus:

```
git tag
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
beta
release
```

> **Note**
>
> Wenn keine Tags definiert wurden, gibt git tag nichts zurück.

Weitere Optionen findest du in deiner Git-Dokumentation.

So zeigen Sie Informationen zu einem Git-Tag in einem lokalen Repo an

1. Führen Sie den Befehl git show ***tag-name*** aus. Führen Sie beispielsweise den folgenden Befehl aus, um Informationen zu einem Git-Tag mit dem Namen `beta` anzuzeigen:

```
git show beta
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
commit 317f8570...ad9e3c09
Author: John Doe <johndoe@example.com>
Date:   Tue Sep 23 13:49:51 2014 -0700

    Added horse.txt

diff --git a/horse.txt b/horse.txt
new file mode 100644
index 0000000..df42ff1
--- /dev/null
```

```
+++ b/horse.txt
@@ -0,0 +1 @@
+The horse (Equus ferus caballus) is one of two extant subspecies of Equus ferus
\ No newline at end of file
```

> **Note**
>
> Geben Sie :q ein, um die Ausgabe der Tag-Informationen zu verlassen.

Weitere Optionen findest du in deiner Git-Dokumentation.

So zeigen Sie Informationen zu Git-Tags in einem CodeCommit Repository an

1. Führen Sie den Befehl git ls-remote --tags aus.

```
git ls-remote --tags
```

2. Bei Erfolg erzeugt dieser Befehl als Ausgabe eine Liste der Git-Tags im CodeCommit Repository:

```
129ce87a...70fbffba    refs/tags/beta
785de9bd...59b402d8    refs/tags/release
```

   Wenn keine Git-Tags definiert wurden, gibt git ls-remote --tags eine leere Zeile zurück.

Weitere Optionen findest du in deiner Git-Dokumentation.

Lösche ein Git-Tag in AWS CodeCommit

Um ein Git-Tag in einem CodeCommit Repository zu löschen, verwende Git aus einem lokalen Repo, das mit dem CodeCommit Repository verbunden ist.

Verwende Git, um ein Git-Tag zu löschen

Gehen Sie wie folgt vor, um mit Git aus einem lokalen Repository ein Git-Tag in einem CodeCommit Repository zu löschen.

Diese Schritte wurden unter der Annahme geschrieben, dass Sie das lokale Repository bereits mit dem Repository verbunden haben. CodeCommit Anweisungen finden Sie unter Herstellen einer Verbindung mit einem Repository.

1. Um das Git-Tag aus dem lokalen Repository zu löschen, führen Sie den git tag -d ***tag-name*** Befehl aus, wobei *tag-name der Name* des Git-Tags ist, das Sie löschen möchten.

   > **Tip**
   >
   > Führen Sie den Befehl git tag aus, um eine Liste der Git-Tag-Namen zu erhalten.

   Um beispielsweise ein Git-Tag im lokalen Repo mit dem Namen `beta` zu löschen:

   ```
   git tag -d beta
   ```

2. Um das Git-Tag aus dem CodeCommit Repository zu löschen, führen Sie den git push ***remote-name*** --delete ***tag-name*** Befehl aus, wobei *remote-name* der Spitzname ist, den das lokale Repo für das CodeCommit Repository verwendet, und *tag-name der Name* des Git-Tags ist, das Sie aus dem Repository löschen möchten. CodeCommit

   > **Tip**
   >
   > Führen Sie den Befehl aus, um eine Liste der CodeCommit Repository-Namen und ihrer URLs zu erhalten. git remote -v

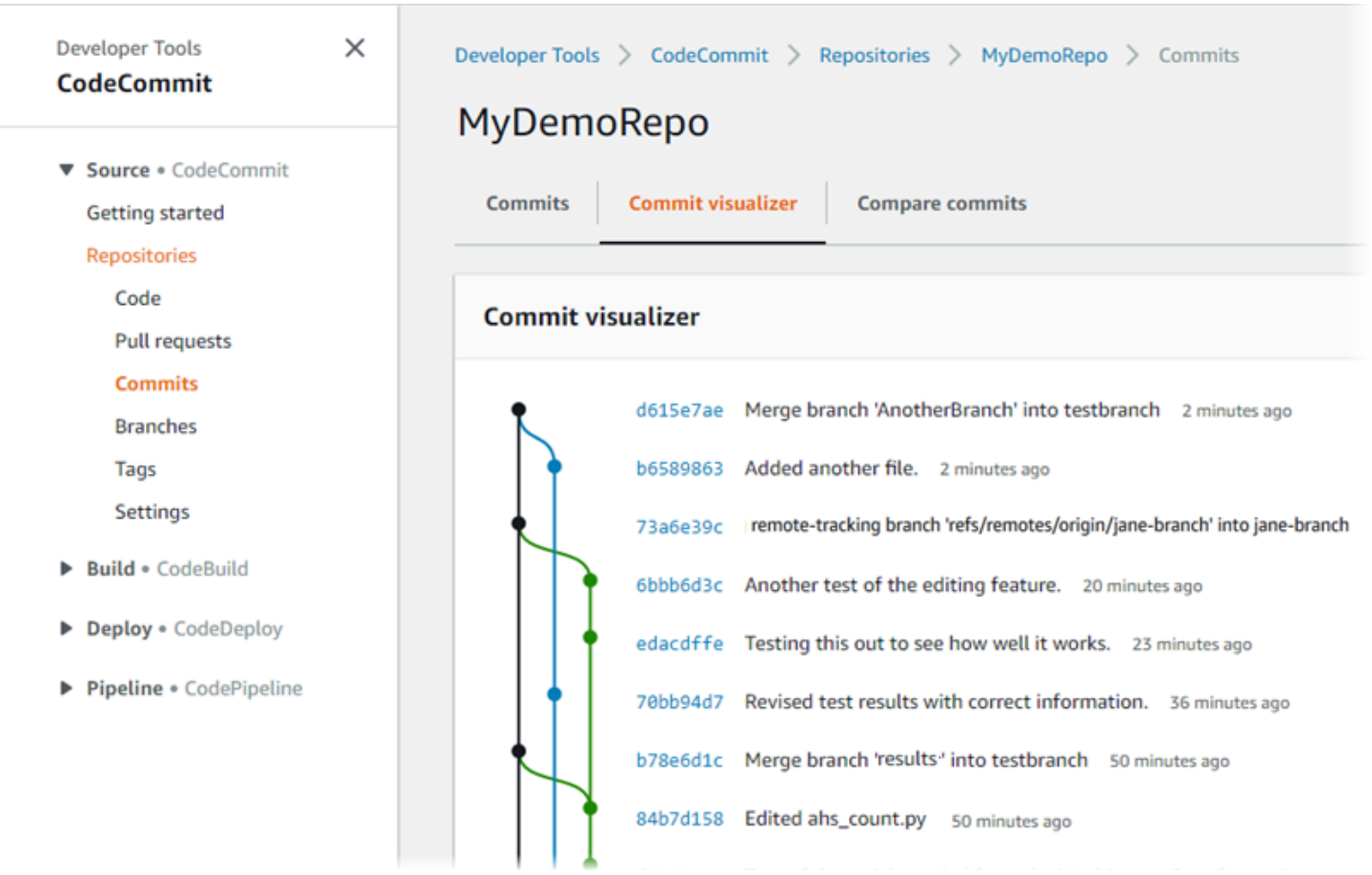
   Um beispielsweise ein Git-Tag mit dem Namen `beta` im CodeCommit Repository mit dem Namen zu löschen`origin`:

   ```
   git push origin --delete beta
   ```

Mit Branches in AWS CodeCommit Repositorys arbeiten

Was ist ein Branch? In Git sind Branches Zeiger oder Verweise auf einen Commit. In der Entwicklung sind sie eine praktische Möglichkeit, die Arbeit zu organisieren. Du kannst Branches verwenden, um die Arbeit an einer neuen oder anderen Version von Dateien voneinander zu trennen, ohne die Arbeit in anderen Branches zu beeinträchtigen. Sie können Branches verwenden, um neue Funktionen zu entwickeln, spezifische Projektversionen von einem bestimmten Commit zu speichern und mehr. Wenn Sie Ihren ersten Commit erstellen, wird ein Standard-Branch für Sie erstellt. Dieser Standard-Branch wird als Basis- oder Standard-Branch in lokalen Repositorys (Repos) verwendet, wenn Benutzer das Repository klonen. Der Name dieses Standard-Branches hängt davon ab, wie Sie Ihren ersten Commit erstellen. Wenn Sie die erste Datei mithilfe der CodeCommit Konsole, des oder eines der SDKs zu Ihrem Repository hinzufügen, lautet der Name dieses Standard-Branches main. AWS CLI Dies ist der Standard-Branch-Name, der in den Beispielen in diesem Handbuch verwendet wird. Wenn du deinen ersten Commit mit einem Git-Client pushst, gibt der Git-Client den Namen des Standard-Branches als Standard an. Erwägen Sie, Ihren Git-Client so zu konfigurieren, dass er main als Namen für den ersten Branch verwendet.

CodeCommitIn kannst du den Standard-Branch für dein Repository ändern. Sie können auch Branches erstellen und löschen und Details über einen Branch anzeigen. Sie können die Unterschiede zwischen einem Branch und dem Standard-Branch (oder zwei anderen Branches) schnell ermitteln. Um den Verlauf der Branches und Merges in deinem Repository anzusehen, kannst du den Commit-Visualizer verwenden, der in der folgenden Grafik dargestellt ist.



Informationen zur Arbeit mit anderen Aspekten Ihres Repositorys finden Sie unter CodeCommit,Arbeiten mit Repositorien, Arbeiten mit Dateien Verwenden von Pull-AnforderungenMit Commits arbeiten, und. Arbeiten mit Benutzereinstellungen

Themen

- Erstellen Sie einen Zweig in AWS CodeCommit
- Beschränken Sie die Anzahl von Pushs und Merges auf Branches in AWS CodeCommit
- Details zum Branch finden Sie in AWS CodeCommit
- Vergleichen und zusammenführen in AWS CodeCommit
- Ändern Sie die Zweigeinstellungen in AWS CodeCommit
- Lösche einen Branch in AWS CodeCommit

Erstellen Sie einen Zweig in AWS CodeCommit

Sie können die CodeCommit Konsole oder die verwenden AWS CLI , um Branches für Ihr Repository zu erstellen. Dies ist eine schnelle Möglichkeit, um separat an einer neuen oder anderen Version von Dateien zu arbeiten, ohne den Standard-Branch zu beeinträchtigen. Nachdem Sie einen Branch in der CodeCommit Konsole erstellt haben, müssen Sie diese Änderung in Ihr lokales Repository übernehmen. Alternativ können Sie lokal einen Branch erstellen und dann Git von einem lokalen Repo aus verwenden, das mit dem CodeCommit Repository verbunden ist, um diese Änderung zu pushen.

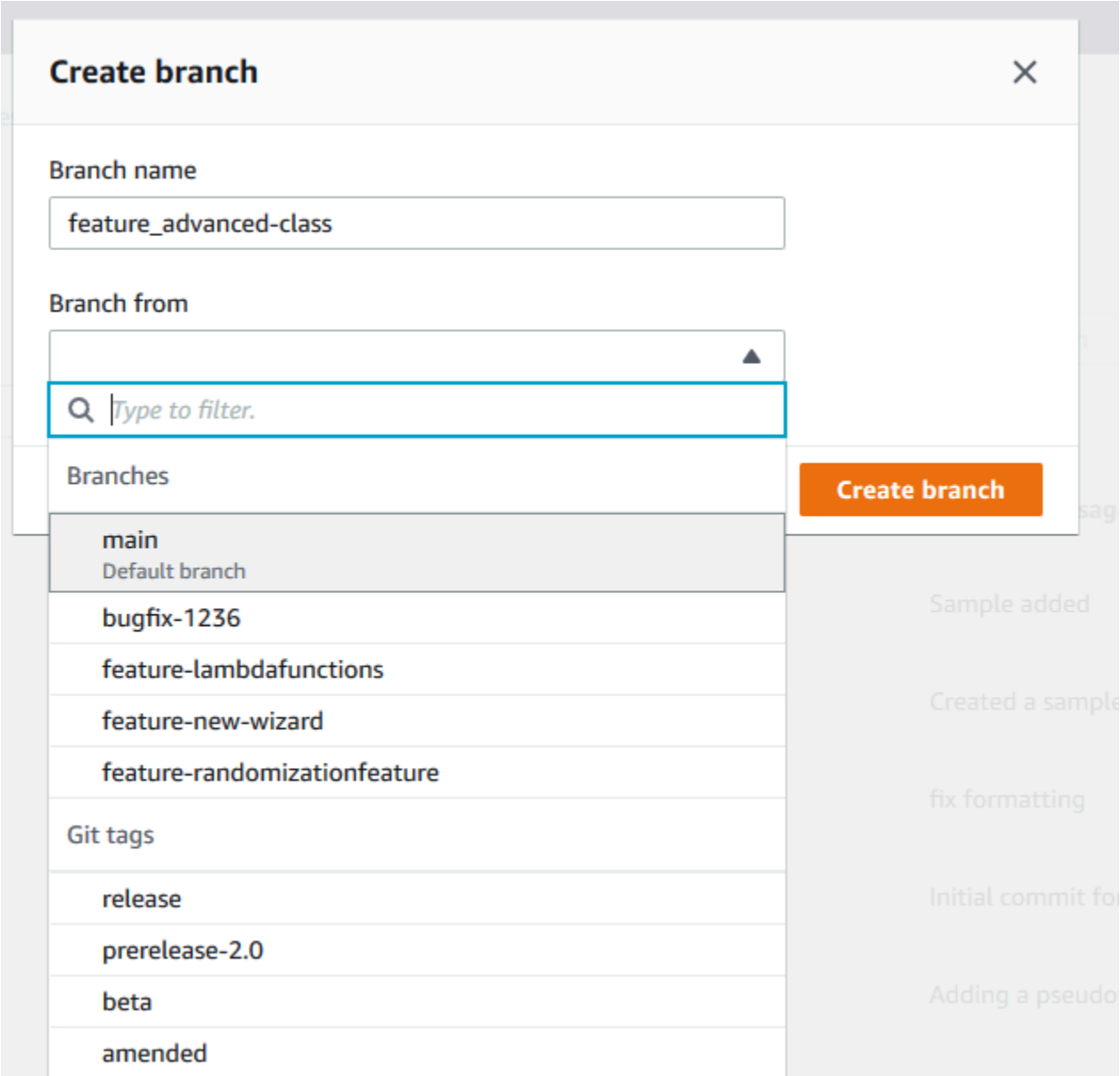
Themen

- Erstelle einen Branch (Konsole)
- Einen Zweig erstellen (Git)
- Erstelle einen Zweig (AWS CLI)

Erstelle einen Branch (Konsole)

Sie können die CodeCommit Konsole verwenden, um einen Branch in einem CodeCommit Repository zu erstellen. Wenn Benutzer das nächste Mal Änderungen vom Repository abrufen, sehen sie den neuen Branch.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie einen Branch erstellen möchten.
3. Wählen Sie im Navigationsbereich Branches aus.
4. Klicken Sie auf Create branch.



Geben Sie im Feld Branch name (Branch-Name) einen Namen für den Branch ein. Wählen Sie unter Branch from (Verzweigen von) entweder einen Branch oder ein Tag aus der Liste aus oder fügen Sie eine Commit-ID ein. Klicken Sie auf Create branch.

Einen Zweig erstellen (Git)

Gehen Sie wie folgt vor, um mit Git von einem lokalen Repository aus einen Branch in einem lokalen Repository zu erstellen und diesen Branch dann in das Repository zu übertragen. CodeCommit

Diese Schritte wurden unter der Annahme geschrieben, dass Sie das lokale Repository bereits mit dem Repository verbunden haben. CodeCommit Anweisungen finden Sie unter Herstellen einer Verbindung mit einem Repository.

1. Erstellen Sie einen Branch in Ihrem lokalen Repository, indem Sie den git checkout -b ***new-branch-name*** Befehl ausführen. Dabei *new-branch-name*steht der Name des neuen Branches.

   Mit dem folgenden Befehl wird beispielsweise ein Branch mit dem Namen `MyNewBranch` im lokalen Repository erstellt:

   ```
   git checkout -b MyNewBranch
   ```

2. Um den neuen Branch vom lokalen Repository in das CodeCommit Repository zu übertragen, führen Sie den git push Befehl aus ***remote-name*** und geben Sie dabei sowohl den als auch den an. ***new-branch-name***

   Um beispielsweise einen neuen Branch im lokalen Repository mit dem Namen in das CodeCommit Repository mit dem Spitznamen `MyNewBranch` zu übertragen: `origin`

   ```
   git push origin MyNewBranch
   ```

> **Note**
>
> Wenn Sie die `-u`-Option zu git push hinzufügen (z. B. git push -u origin main), dann können Sie zukünftig git push ohne *remote-name branch-name* ausführen. Es werden Upstream-Nachverfolgungsdaten festgelegt. Führen Sie git remote show ***remote-name*** aus (z. B. git remote show origin), um Upstream-Nachverfolgungsdaten abzurufen.
> Führen Sie git branch --all aus, um eine Liste aller lokalen und Remote-Nachverfolgungs-Branches anzuzeigen.
> Um einen Zweig im lokalen Repository einzurichten, der mit einem Zweig im CodeCommit Repository verbunden ist, führen Sie den Befehl aus. git checkout ***remote-branch-name***

Weitere Optionen findest du in deiner Git-Dokumentation.

Erstelle einen Zweig (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

Gehen Sie wie folgt vor, um mit dem einen Branch in einem CodeCommit Repository AWS CLI zu erstellen und diesen Branch dann per Push in das CodeCommit Repository zu übertragen. Die Schritte zum Erstellen eines ersten Commits und zum Angeben des Namens des Standard-Branches für ein leeres Repository finden Sie unter Erstellen Sie den ersten Commit für ein Repository mit dem AWS CLI.

1. Führen Sie den Befehl create-branch aus und geben Sie Folgendes an:

   - Der Name des CodeCommit Repositorys, in dem der Branch erstellt wird (mit der --repository-name Option).

     > **Note**
     >
     > Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl list-repositories aus.

   - Der Name des neuen Branches (mit der Option --branch-name).
   - Die ID des Commits, auf den der neue Branch verweist (mit der Option --commit-id).

   Um beispielsweise einen Branch mit dem Namen zu erstellen, der auf `MyNewBranch` die Commit-ID `317f8570EXAMPLE` in einem CodeCommit Repository mit dem Namen verweist: `MyDemoRepo`

   ```
   aws codecommit create-branch --repository-name MyDemoRepo --branch-name MyNewBranch
    --commit-id 317f8570EXAMPLE
   ```

   Dieser Befehl liefert nur eine Ausgabe, wenn Fehler aufgetreten sind.

2. Um die Liste der verfügbaren CodeCommit Repository-Branches in Ihrem lokalen Repository mit dem neuen Namen der Remote-Branches zu aktualisieren, führen Sie git remote update ***`remote-name`*** folgenden Befehl aus.

   Um beispielsweise die Liste der verfügbaren Branches für das CodeCommit Repository mit dem Spitznamen `origin` zu aktualisieren:

```
git remote update origin
```

> **Note**
>
> Alternativ können Sie den Befehl git fetch ausführen. Sie können sich auch alle Remote-Branches ansehengit branch --all, indem Sie den Befehl ausführen, aber bis Sie die Liste Ihres lokalen Repos aktualisiert haben, erscheint der von Ihnen erstellte Remote-Branch nicht in der Liste.
> Weitere Optionen findest du in deiner Git-Dokumentation.

3. Um einen Branch im lokalen Repository einzurichten, der mit dem neuen Branch im CodeCommit Repository verbunden ist, führe git checkout ***remote-branch-name*** folgenden Befehl aus.

> **Note**
>
> Führen Sie den git remote -v Befehl aus, um eine Liste der CodeCommit Repository-Namen und ihrer URLs abzurufen.

Beschränken Sie die Anzahl von Pushs und Merges auf Branches in AWS CodeCommit

Standardmäßig kann jeder CodeCommit Repository-Benutzer, der über ausreichende Berechtigungen verfügt, um Code in das Repository zu übertragen, zu jedem Branch in diesem Repository beitragen. Dies gilt unabhängig davon, wie Sie einen Branch zum Repository hinzufügen: über die Konsole, die Befehlszeile oder Git. Sie können jedoch einen Branch konfigurieren, sodass nur einige Repository-Benutzer Code zu diesem Branch senden oder mit ihm zusammenführen können. Beispielsweise können Sie eine Verzweigung für den Produktionscode so konfigurieren, dass nur bestimmte erfahrene Entwickler Änderungen an dieser Verzweigung vornehmen oder zusammenführen können. Andere Entwickler können etwas aus der Verzweigung abrufen, ihre eigenen Verzweigungen erstellen und Pull-Anforderungen erstellen, aber sie können keine Änderungen an diese Verzweigung senden oder mit dieser zusammenführen. Sie können diesen Zugriff konfigurieren, indem Sie eine bedingte Richtlinie erstellen, die einen Kontextschlüssel für einen oder mehrere Zweige in IAM verwendet.

> **Note**
>
> Um einige der Verfahren in diesem Thema abzuschließen, müssen Sie sich mit einem Administratorbenutzer anmelden, der über ausreichende Berechtigungen zum Konfigurieren und Anwenden von IAM-Richtlinien verfügt. Weitere Informationen finden Sie unter Einen IAM-Admin-Benutzer und eine IAM-Admin-Gruppe erstellen.

Themen

- Konfigurieren Sie eine IAM-Richtlinie, um Pushs und Merges auf einen Branch zu beschränken
- Wenden Sie die IAM-Richtlinie auf eine IAM-Gruppe oder -Rolle an
- Testen Sie die Richtlinie

Konfigurieren Sie eine IAM-Richtlinie, um Pushs und Merges auf einen Branch zu beschränken

Sie können in IAM eine Richtlinie erstellen, die verhindert, dass Benutzer einen Branch aktualisieren, einschließlich der Übertragung von Commits an einen Branch und der Zusammenführung von Pull-Requests an einen Branch. Dazu verwendet Ihre Richtlinie eine bedingte Anweisung, sodass die Wirkung der Anweisung `Deny` nur stattfindet, wenn die Bedingung erfüllt ist. Die APIs, die Sie in die Anweisung `Deny` aufnehmen, bestimmen, welche Aktionen nicht erlaubt sind. Sie können diese Richtlinie so konfigurieren, dass sie nur für einen Branch in einem Repository, eine Reihe von Branches in einem Repository oder für alle Branches gilt, die den Kriterien in allen Repositorys in einem Amazon Web Services Services-Konto entsprechen.

So erstellen Sie eine bedingte Richtlinie für Verzweigungen

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Richtlinien.
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie JSON, und fügen Sie dann die folgende Beispielrichtlinie ein. Ersetzen Sie den Wert von `Resource` durch den ARN des Repositories, das die Verzweigung enthält, für die Sie den Zugriff einschränken möchten. Ersetzen Sie den Wert von `codecommit:References` durch einen Verweis auf den Branch oder die Branches, für die Sie den Zugriff einschränken möchten. Diese Richtlinie verbietet beispielsweise das Senden von Commits, das Zusammenführen

von Branches, das Löschen von Branches, das Zusammenführen von Pull-Requests und das Hinzufügen von Dateien zu einem Branch mit dem Namen und einem Branch mit dem Namen *main*in einem Repository mit dem Namen*prod*: *MyDemoRepo*

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "codecommit:GitPush",
                "codecommit:DeleteBranch",
                "codecommit:PutFile",
                "codecommit:MergeBranchesByFastForward",
                "codecommit:MergeBranchesBySquash",
                "codecommit:MergeBranchesByThreeWay",
                "codecommit:MergePullRequestByFastForward",
                "codecommit:MergePullRequestBySquash",
                "codecommit:MergePullRequestByThreeWay"
            ],
            "Resource": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
            "Condition": {
                "StringEqualsIfExists": {
                    "codecommit:References": [
                        "refs/heads/main",
                        "refs/heads/prod"
                     ]
                },
                "Null": {
                    "codecommit:References": "false"
                }
            }
        }
    ]
}
```

Branches in Git sind einfach Zeiger (Referenzen) auf den SHA-1-Wert des HEAD-Commit, weshalb die Bedingung `References` verwendet. Die `Null` -Anweisung in jeder Richtlinie erforderlich, die ein `Deny` bewirkt, und wobei eine der Aktionen `GitPush` ist. Dies ist aufgrund der Art und Weise erforderlich, wie Git und Git `git-receive-pack` funktionieren, wenn Änderungen von einem lokalen Repo übertragen werden CodeCommit.

> **Tip**
>
> Um eine Richtlinie zu erstellen, die für alle Branches mit dem Namen main in allen Repositorys eines Amazon Web Services Services-Kontos gilt, ändern Sie den Wert `Resource` von einem Repository-ARN in ein Sternchen ()*.

5. Wählen Sie Richtlinie prüfen. Korrigieren Sie alle Fehler in Ihrer Richtlinienanweisung und fahren Sie dann mit Create policy (Richtlinie erstellen) fort.
6. Wenn JSON validiert wird, wird die Seite Seite Create policy (Richtlinie erstellen) angezeigt. Eine Warnung wird im Abschnitt Summary (Zusammenfassung angezeigt, die Sie darauf hinweist, dass diese Richtlinie keine Berechtigungen gewährt. Das ist normal.

   - Geben Sie unter Name einen Namen für diese Richtlinie wie **`DenyChangesToMain`** ein.
   - Geben Sie in Description (Beschreibung) eine Beschreibung zum Zweck der Richtlinie ein. Dies ist zwar optional, wird aber empfohlen.
   - Wählen Sie Richtlinie erstellen aus.

Wenden Sie die IAM-Richtlinie auf eine IAM-Gruppe oder -Rolle an

Sie haben eine Richtlinie erstellt, die Push- und Merge-Vorgänge auf einen Branch beschränkt, aber die Richtlinie ist erst wirksam, wenn Sie sie auf einen IAM-Benutzer, eine IAM-Gruppe oder eine IAM-Rolle anwenden. Es hat sich bewährt, die Richtlinie auf eine IAM-Gruppe oder -Rolle anzuwenden. Die Anwendung von Richtlinien auf einzelne IAM-Benutzer lässt sich nicht gut skalieren.

Anwendung der bedingten Richtlinie auf eine Gruppe oder Rolle

1. Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wenn Sie die Richtlinie auf eine IAM-Gruppe anwenden möchten, wählen Sie im Navigationsbereich Gruppen aus. Wenn Sie die Richtlinie auf eine Rolle anwenden möchten, die Benutzer übernehmen, wählen Sie Rolle aus. Wählen Sie den Namen der Gruppe oder Rolle aus.
3. Wählen Sie auf der Registerkarte Permissions (Berechtigungen) die Option Attach Policy (Richtlinie zuweisen) aus.
4. Wählen Sie die bedingte Richtlinie, die Sie zuvor erstellt haben, aus der Liste der Richtlinien aus, und klicken Sie auf Attach policy (Richtlinie hinzufügen).

Weitere Informationen finden Sie unter IAM-Richtlinien anhängen und trennen.

Testen Sie die Richtlinie

Sie sollten die Auswirkungen der von Ihnen auf die Gruppe oder Rolle angewendeten Richtlinie testen, um sicherzustellen, dass sie wie erwartet funktioniert. Es gibt viele Möglichkeiten, dies zu realisieren. Wenn Sie beispielsweise eine ähnliche Richtlinie wie die oben gezeigte testen möchten, können Sie wie folgt vorgehen:

- Melden Sie sich bei der CodeCommit Konsole mit einem IAM-Benutzer an, der entweder Mitglied einer IAM-Gruppe ist, auf die die Richtlinie angewendet wurde, oder der eine Rolle annimmt, auf die die Richtlinie angewendet wurde. In der Konsole fügen Sie der Verzweigung, für die die Einschränkungen gelten, eine Datei hinzu. Sie sollten eine Fehlermeldung sehen, wenn Sie versuchen, eine Datei in dieser Verzweigung zu speichern oder eine Datei in die Verzweigung hochzuladen. Fügen Sie einer anderen Verzweigung eine Datei hinzu. Die Operation sollte erfolgreich sein.
- Melden Sie sich bei der CodeCommit Konsole mit einem IAM-Benutzer an, der entweder Mitglied einer IAM-Gruppe ist, auf die die Richtlinie angewendet wurde, oder der eine Rolle annimmt, auf die die Richtlinie angewendet wurde. Erstellen Sie eine Pull-Anforderung, die mit dem Branch zusammengeführt wird, für den die Einschränkungen gelten. Sie sollten in der Lage sein, die Pull-Anforderung zu erstellen, erhalten aber einen Fehler, wenn Sie versuchen, sie zusammenzuführen.
- Erstellen Sie über das Terminal oder die Befehlszeile einen Commit für den Branch, für den die Einschränkungen gelten, und übertragen Sie diesen Commit dann per Push in das CodeCommit Repository. Sie sollten eine Fehlermeldung sehen. Commits und Sendeoperationen von anderen Verzweigungen aus sollten wie gewohnt funktionieren.

Details zum Branch finden Sie in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um Details zu den Branches in einem CodeCommit Repository anzuzeigen. Sie können das Datum des letzten Commit für einen Branche, die Commit-Nachrichtung und vieles andere mehr anzeigen. Sie können auch das AWS CLI oder Git von einem lokalen Repo verwenden, das mit dem CodeCommit Repository verbunden ist.

Themen

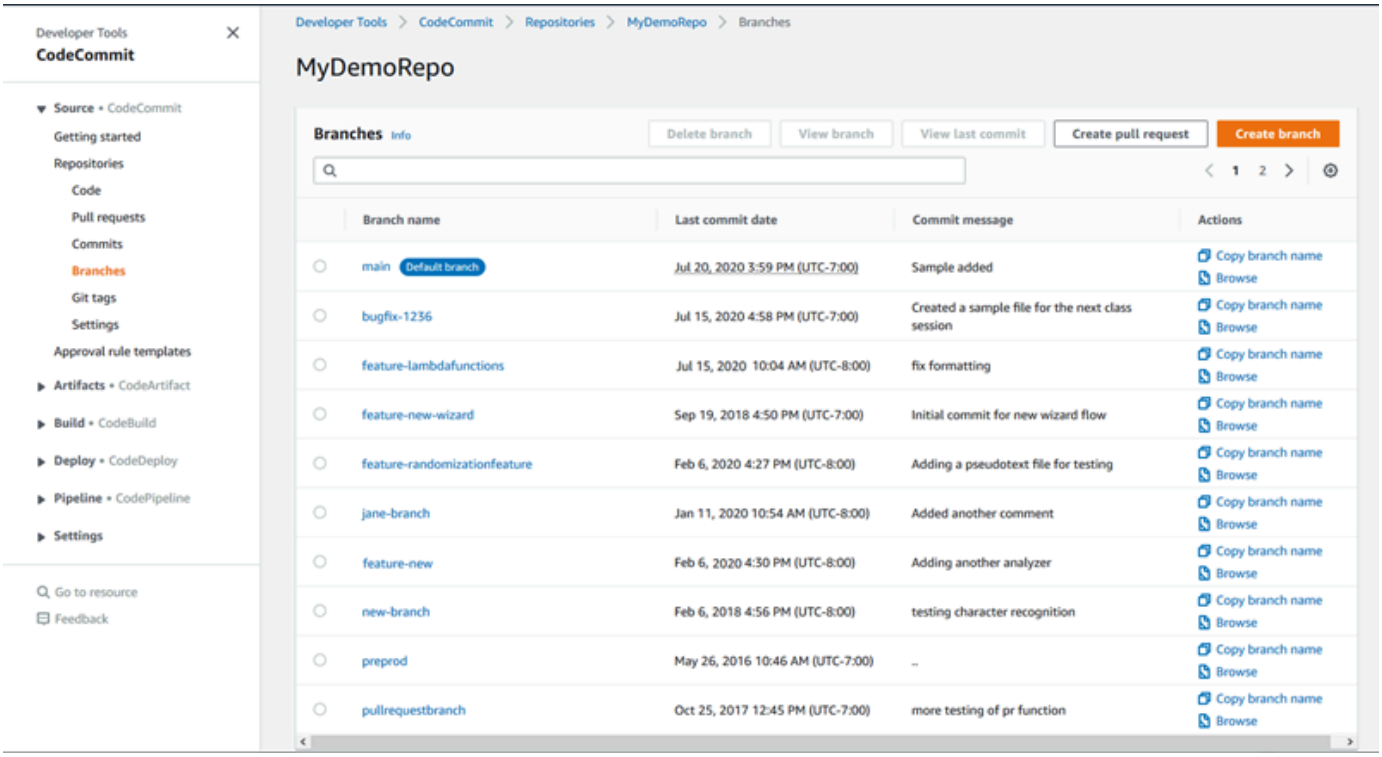
- Branch-Details anzeigen (Konsole)
- Filialdetails anzeigen (Git)

- Details zur Filiale anzeigen (AWS CLI)

Branch-Details anzeigen (Konsole)

Verwenden Sie die CodeCommit Konsole, um schnell eine Liste der Branches für Ihr Repository und Details zu den Branches anzuzeigen.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Branch-Details anzeigen möchten.
3. Wählen Sie im Navigationsbereich Branches aus.



4. Der Name des Branches, der als Standard für das Repository verwendet wird, wird neben Standard-Branch angezeigt. Um Details über das letzte Commit für einen Branch anzuzeigen, wählen Sie den Branch und dann View last commit (Letztes Commit anzeigen) aus. Um die Dateien und den Code in einem Branch anzuzeigen, wählen Sie den Branch-Namen.

Filialdetails anzeigen (Git)

Um Git von einem lokalen Repo aus zu verwenden, um Details sowohl zu den lokalen als auch zu den Remote-Tracking-Branches für ein CodeCommit Repository anzuzeigen, führen Sie den git branch Befehl aus.

Die folgenden Schritte wurden unter der Annahme geschrieben, dass Sie das lokale Repository bereits mit dem Repository verbunden haben. CodeCommit Anweisungen finden Sie unter Herstellen einer Verbindung mit einem Repository.

1. Führen Sie den Befehl git branch aus und geben Sie dabei die Option --all an:

```
git branch --all
```

2. Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die wie folgt aussehen sollte:

```
  MyNewBranch
* main
  remotes/origin/MyNewBranch
  remotes/origin/main
```

   Das Sternchen (*) wird neben dem aktuell geöffneten Branch angezeigt. Die Einträge danach sind Remotenachverfolgungs-Referenzen.

> **Tip**
>
> git branch zeigt lokale Branches an.
> git branch -r zeigt Remote-Branches an.
> git checkout ***existing-branch-name*** wechselt zu dem angegebenen Branch-Namen. Sofern direkt danach git branch ausgeführt wird, wird daneben ein Sternchen angezeigt (*).
> git remote update ***remote-name***aktualisiert Ihr lokales Repo mit der Liste der verfügbaren CodeCommit Repository-Zweige. (Um eine Liste der CodeCommit Repository-Namen und ihrer URLs zu erhalten, führen Sie den git remote -v Befehl aus.)

Weitere Optionen findest du in deiner Git-Dokumentation.

Details zur Filiale anzeigen (AWS CLI)

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

Um Details AWS CLI zu den Branches in einem CodeCommit Repository anzuzeigen, führen Sie einen oder mehrere der folgenden Befehle aus:

- Um eine Liste von Branch-Namen anzuzeigen, führen Sie list-branches aus.
- Um Informationen zu einem bestimmten Zweig anzuzeigen, führen Sie get-branch aus.

So zeigen Sie eine Liste mit Branch-Namen an

1. Führen Sie den list-branches Befehl aus und geben Sie den Namen des CodeCommit Repositorys an (mit der `--repository-name` Option).

   > **Tip**
   >
   > Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl list-repositories aus.

   Um beispielsweise Details zu den Branches in einem CodeCommit Repository mit dem Namen anzuzeigen: `MyDemoRepo`

   ```
   aws codecommit list-branches --repository-name MyDemoRepo
   ```

2. Ist der Befehl erfolgreich, wird ein `branchNameList`-Objekt mit einem Eintrag für jeden Branch ausgegeben.

   Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

   ```
   {
       "branches": [
           "MyNewBranch",
           "main"
       ]
   }
   ```

So zeigen Sie Informationen über einen Branch an

1. Führen Sie den Befehl get-branch aus und geben Sie Folgendes an:

   - Den Repository-Namen (mit der Option --repository-name).
   - Den Branch-Namen (mit der Option --branch-name).

   Um beispielsweise Informationen über einen Branch anzuzeigen, der `MyNewBranch` in einem CodeCommit Repository mit dem Namen`MyDemoRepo`:

   ```
   aws codecommit get-branch --repository-name MyDemoRepo --branch-name MyNewBranch
   ```

2. Ist der Befehl erfolgreich, werden der Name des Branches und die ID des letzten, für den Branch ausgeführten Commits ausgegeben.

   Es folgt eine Beispielausgabe basierend auf dem vorangehenden Beispielbefehl:

   ```
   {
       "branch": {
             "branchName": "MyNewBranch",
             "commitID": "317f8570EXAMPLE"
       }
   }
   ```

Vergleichen und zusammenführen in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um Branches in einem CodeCommit Repository zu vergleichen. Durch den Vergleich von Branches können Sie schnell die Unterschiede zwischen einem Branch und dem Standard-Branch erkennen, oder die Unterschiede zwischen zwei Branches.

Themen

- Vergleichen Sie einen Branch mit dem Standard-Branch
- Vergleichen Sie zwei spezifische Branchen
- Zwei Zweige zusammenführen ()AWS CLI

Vergleichen Sie einen Branch mit dem Standard-Branch

Verwende die CodeCommit Konsole, um dir schnell die Unterschiede zwischen einem Branch und dem Standard-Branch für dein Repository anzusehen.

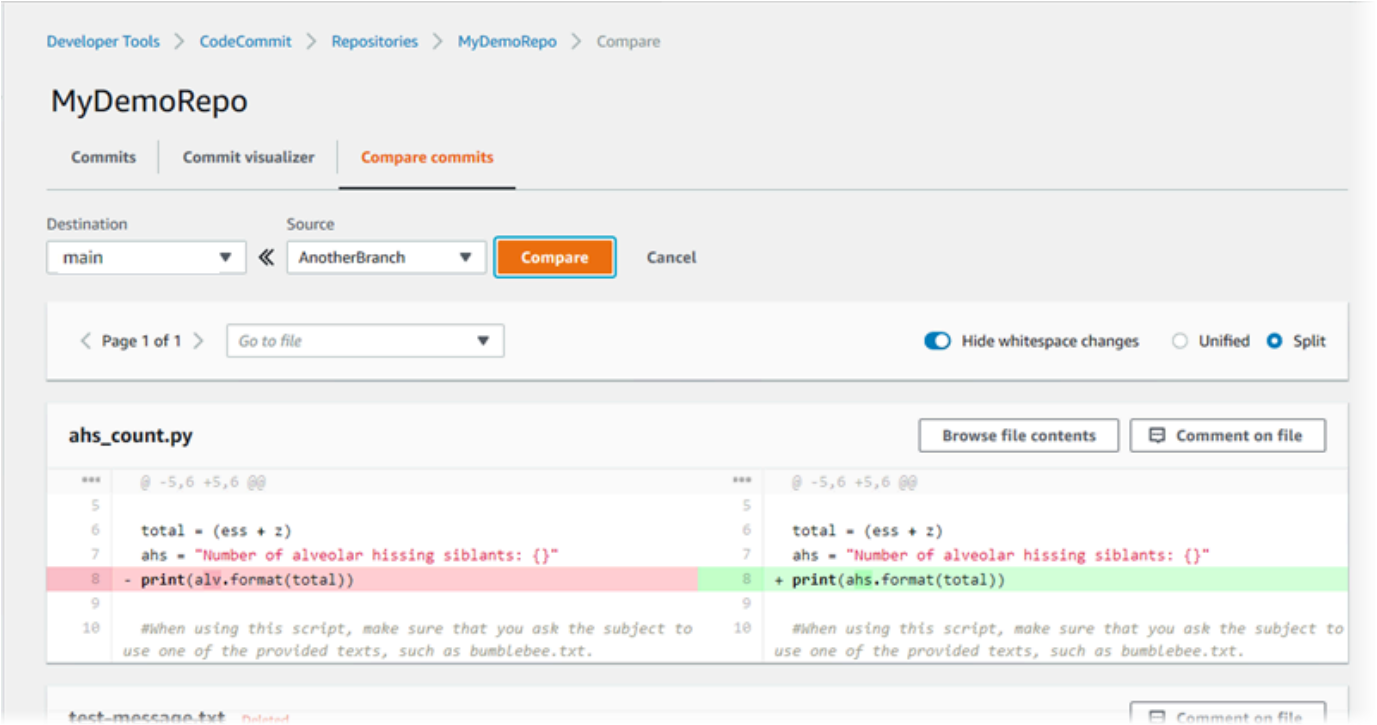
1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Branches vergleichen möchten.
3. Wählen Sie im Navigationsbereich Commits und dann die Compare commits (Commits vergleichen)-Registerkarte aus.
4. Wählen Sie unter Destination (Ziel) den Namen des Standard-Branches aus. Wählen Sie unter Source (Quelle) den Branch aus, den Sie mit dem Standard-Branch vergleichen möchten. Wählen Sie Compare aus.

Vergleichen Sie zwei spezifische Branchen

Verwenden Sie die CodeCommit Konsole, um die Unterschiede zwischen zwei Branchen anzuzeigen, die Sie vergleichen möchten.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie Branches vergleichen möchten.
3. Wählen Sie im Navigationsbereich Commits und dann die Compare commits (Commits vergleichen)-Registerkarte aus.
4. Wählen Sie unter Destination (Ziel) und Source (Quelle) die zwei zu vergleichenden Branches aus und wählen Sie dann Compare (Vergleichen) aus. Um die Liste der geänderten Dateien anzuzeigen, erweitern Sie die Liste der geänderten Dateien. Sie können die Änderungen an den Dateien nebeneinander (Ansicht Split) oder inline (Ansicht Unified) anzeigen.

> **Note**
>
> Wenn Sie als IAM-Benutzer angemeldet sind, können Sie Ihre Einstellungen für die Anzeige von Code und andere Konsoleneinstellungen konfigurieren und speichern. Weitere Informationen finden Sie unter Arbeiten mit Benutzereinstellungen.



Zwei Zweige zusammenführen ()AWS CLI

Sie können zwei Branches in einem CodeCommit Repository zusammenführen, indem Sie eine der verfügbaren Merge-Strategien AWS CLI verwenden, indem Sie einen der folgenden Befehle ausführen:

- Führen Sie den Befehl merge-branches-by-fast-forward aus, um zwei Branches mithilfe der Mergestrategie mit Vorlauf zusammenzuführen.
- Führen Sie den Befehl merge-branches-by-squash aus, um zwei Branches mithilfe der Squashmerge-Strategie zusammenzuführen.
- Führen Sie den Befehl merge-branches-by-three-way aus, um zwei Branches mithilfe der Dreiwegemerge-Strategie zusammenzuführen.

Sie können Zusammenführungen auch testen, indem Sie den Befehl create-unreferenced-merge-commit ausführen. Weitere Informationen finden Sie unter Auflösen von Konflikten in einer Pull-Anforderung.

> **Note**
>
> Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

Um das zu verwenden AWS CLI , um zwei Zweige in einem CodeCommit Repository zusammenzuführen

1. Um zwei Branches mithilfe der Mergestrategie mit Vorlauf zusammenzuführen, führen Sie den Befehl merge-branches-by-fast-forward aus und geben Sie Folgendes an:

   - Den Namen des Quell-Branches, der die Änderungen enthält, die Sie zusammenführen möchten (mit der Option --source-commit-specifier).
   - Den Namen des Ziel-Branches, in dem Sie Ihre Änderungen zusammenführen möchten (mit der Option --destination-commit-specifier).
   - Der Name des Repositorys (mit der Option --repository-name).

   Um zum Beispiel einen Quell-Branch namens *bugfix-1234* mit einem Ziel-Branch namens *preprod* in einem Repository mit dem Namen zusammenzuführen: *MyDemoRepo*

   ```
   aws codecommit merge-branches-by-fast-forward --source-commit-specifier bugfix-bug1234 --destination-commit-specifier preprod --repository-name MyDemoRepo
   ```

   Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

   ```
   {
       "commitId": "4f178133EXAMPLE",
       "treeId": "389765daEXAMPLE"
   }
   ```

2. Um zwei Branches mithilfe der Squashmerge-Strategie zusammenzuführen, führen Sie den Befehl merge-branches-by-squash aus und geben Sie Folgendes an:

   - Den Namen des Quell-Branches, der die Änderungen enthält, die Sie zusammenführen möchten (mit der Option --source-commit-specifier).

- Den Namen des Ziel-Branches, in dem Sie Ihre Änderungen zusammenführen möchten (mit der Option --destination-commit-specifier).
- Der Name des Repositorys (mit der Option --repository-name).
- Die einzuschließende Commit-Nachricht (mit der Option --commit-message).
- Der für den Commit zu verwendende Name (mit der Option --name).
- Die für den Commit zu verwendende E-Mail-Adresse (mit der Option --email).

Um zum Beispiel einen Quell-Branch namens bugfix-bug1234 mit einem Ziel-Branch namens bugfix-quarterly in einem Repository mit dem Namen zusammenzuführen: MyDemoRepo

```
aws codecommit merge-branches-by-squash --source-commit-specifier bugfix-bug1234 --
destination-commit-specifier bugfix-quarterly --author-name "Maria Garcia" --email
 "maria_garcia@example.com" --commit-message "Merging in fix branches to prepare
 for a general patch." --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "commitId": "4f178133EXAMPLE",
    "treeId": "389765daEXAMPLE"
}
```

3. Um zwei Branches mithilfe der Dreiwegemerge-Strategie zusammenzuführen, führen Sie den Befehl merge-branches-by-three-way aus und geben Sie Folgendes an:

- Den Namen des Quell-Branches, der die Änderungen enthält, die Sie zusammenführen möchten (mit der Option --source-commit-specifier).
- Den Namen des Ziel-Branches, in dem Sie Ihre Änderungen zusammenführen möchten (mit der Option --destination-commit-specifier).
- Der Name des Repositorys (mit der Option --repository-name).
- Die einzuschließende Commit-Nachricht (mit der Option --commit-message).
- Der für den Commit zu verwendende Name (mit der Option --name).
- Die für den Commit zu verwendende E-Mail-Adresse (mit der Option --email).

Um zum Beispiel einen Quell-Branch namens main mit einem Ziel-Branch namens bugfix-1234 in einem Repository mit dem Namen zusammenzuführen: MyDemoRepo

```
aws codecommit merge-branches-by-three-way --source-commit-specifier main --
destination-commit-specifier bugfix-bug1234 --author-name "Jorge Souza" --email
 "jorge_souza@example.com" --commit-message "Merging changes from main to bugfix
 branch before additional testing."  --repository-name MyDemoRepo
```

Ist der Befehl erfolgreich, wird eine Ausgabe zurückgegeben, die der folgenden ähnelt:

```
{
    "commitId": "4f178133EXAMPLE",
    "treeId": "389765daEXAMPLE"
}
```

Ändern Sie die Zweigeinstellungen in AWS CodeCommit

Sie können in der AWS CodeCommit Konsole oder mit dem ändern, welcher Zweig als Standardzweig verwendet werden soll AWS CLI. Wenn du beispielsweise deinen ersten Commit mit einem Git-Client erstellt hast, der den Standard-Branch auf master setzt, könntest du einen Branch namens main erstellen und dann die Branch-Einstellungen so ändern, dass der neue Branch als Standard-Branch für das Repository festgelegt wird. Um andere Branch-Einstellungen zu ändern, kannst du Git von einem lokalen Repo aus verwenden, das mit dem CodeCommit Repository verbunden ist.

Themen

- Ändern Sie den Standardzweig (Konsole)
- Ändern Sie den Standardzweig ()AWS CLI

Ändern Sie den Standardzweig (Konsole)

Sie können in der AWS CodeCommit Konsole angeben, welcher Branch der Standard-Branch in einem CodeCommit Repository ist.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, bei dem Sie Einstellungen ändern möchten.
3. Wählen Sie im Navigationsbereich Settings (Einstellungen).
4. Wählen Sie unter Default branch (Standard-Branch) die Dropdown-Liste für Branches aus und wählen Sie dann einen anderen Branch. Wählen Sie Speichern.

> **Tip**
>
> - Wenn Sie in der Dropdownliste keinen anderen Zweig sehen, haben Sie keine zusätzlichen Zweige erstellt. Sie können den Standardzweig eines Repositorys nicht ändern, wenn das Repository nur einen Zweig hat. Weitere Informationen finden Sie unter Erstellen Sie einen Zweig in AWS CodeCommit.
> - Wenn Sie den Abschnitt Standardverzweigung nicht sehen, sondern stattdessen Elemente für Benachrichtigungsregeln und Verbindungen sehen, befinden Sie sich im Menü mit den allgemeinen Einstellungen für die Konsole. Das Einstellungsmenü für Repositorys ist unter Repositorys auf derselben Ebene wie Code - und Pull-Anfragen aufgeführt.

Ändern Sie den Standardzweig ()AWS CLI

Um AWS CLI Befehle mit zu verwenden CodeCommit, installieren Sie den AWS CLI. Weitere Informationen finden Sie unter Befehlszeilenreferenz.

AWS CLI Um die Branch-Einstellungen eines Repositorys in einem CodeCommit Repository zu ändern, führen Sie den folgenden Befehl aus:

- update-default-branch (Damit ändern Sie den Standard-Branch.)

So ändern Sie den Standard-Branch

1. Führen Sie den Befehl update-default-branch aus und geben Sie Folgendes an:
   - Der Name des CodeCommit Repositorys, in dem der Standard-Branch aktualisiert wird (mit der --repository-name Option).

> **Tip**
>
> Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl list-repositories aus.

- Der Name des neuen Standard-Branches (mit der Option --default-branch-name).

> **Tip**
>
> Um den Namen des Branches abzurufen, führen Sie den Befehl list-branches aus.

2. Um beispielsweise den Standard-Branch `MyNewBranch` in ein CodeCommit Repository mit dem Namen zu ändern: `MyDemoRepo`

```
aws codecommit update-default-branch --repository-name MyDemoRepo --default-branch-
name MyNewBranch
```

Dieser Befehl liefert nur eine Ausgabe, wenn Fehler aufgetreten sind.

Weitere Optionen findest du in deiner Git-Dokumentation.

Lösche einen Branch in AWS CodeCommit

Sie können die CodeCommit Konsole verwenden, um einen Branch in einem Repository zu löschen. Wenn Sie einen Branch in löschen, wird dieser Branch in einem lokalen Repository CodeCommit nicht gelöscht, sodass Benutzer möglicherweise weiterhin Kopien dieses Branches haben, bis sie das nächste Mal Änderungen abrufen. Um einen Branch lokal zu löschen und diese Änderung in das CodeCommit Repository zu übertragen, verwende Git von einem lokalen Repo, das mit dem CodeCommit Repository verbunden ist.

Durch das Löschen eines Branch werden keine Commits gelöscht, sondern alle Verweise auf die Commits in diesem Branch. Wenn Sie einen Branch löschen, der Commits enthält, die noch nicht in einen anderen Branch im Repository übernommen wurden, können Sie diese Commits nicht mehr abrufen, es sei denn, Sie haben ihre vollständigen Commit-IDs.

> **Note**
>
> Sie können die Anweisungen in diesem Thema nicht verwenden, um den Standard-Branch eines Repository zu löschen. Wenn Sie den Standard-Branch löschen möchten, müssen Sie einen Branch erstellen, diesen zum Standard-Branch machen und dann den alten Branch löschen. Weitere Informationen finden Sie unter Erstellen eines Zweigs und Ändern Sie die Filialeinstellungen.

Themen

- Lösche einen Branch (Konsole)
- Lösche einen Zweig (AWS CLI)
- Einen Zweig löschen (Git)

Lösche einen Branch (Konsole)

Sie können die CodeCommit Konsole verwenden, um einen Branch in einem CodeCommit Repository zu löschen.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories (Repositorys) den Namen des Repositorys aus, in dem Sie einen Branch löschen möchten.
3. Wählen Sie im Navigationsbereich Branches aus.
4. Suchen Sie den Namen des Branches, den Sie löschen möchten, klicken Sie auf Delete branch (Branch löschen) und bestätigen Sie dies.

Lösche einen Zweig (AWS CLI)

Sie können den verwenden AWS CLI , um einen Branch in einem CodeCommit Repository zu löschen, falls dieser Branch nicht der Standard-Branch für das Repository ist. Weitere Hinweise zur Installation und Verwendung von finden Sie unterBefehlszeilenreferenz. AWS CLI

1. Führen Sie am Terminal oder in der Befehlszeile den Befehl delete-branch unter Angabe der folgenden Informationen aus:

- Der Name des CodeCommit Repositorys, in dem der Branch gelöscht werden soll (mit der --repository-name Option).

  > **Tip**
  >
  > Um den Namen des CodeCommit Repositorys abzurufen, führen Sie den Befehl list-repositories aus.

- Name des zu löschenden Branches (mit der Option branch-name)

  > **Tip**
  >
  > Um den Namen des Branches abzurufen, führen Sie den Befehl list-branches aus.

2. Um beispielsweise einen Branch zu löschen, der `MyNewBranch` in einem CodeCommit Repository mit dem folgenden Namen benannt ist: `MyDemoRepo`

   ```
   aws codecommit delete-branch --repository-name MyDemoRepo --branch-name MyNewBranch
   ```

   Dieser Befehl gibt Informationen über den gelöschten Branch zurück, z. B. den Namen des gelöschten Branch sowie die vollständige Commit-ID des Commits, das Kopf des Branch war. Beispielsweise:

   ```
   "deletedBranch": {
       "branchName": "MyNewBranch",
       "commitId": "317f8570EXAMPLE"
   }
   ```

Einen Zweig löschen (Git)

Gehen Sie wie folgt vor, um mit Git aus einem lokalen Repository einen Branch in einem CodeCommit Repository zu löschen.

Diese Schritte wurden unter der Annahme geschrieben, dass Sie das lokale Repository bereits mit dem Repository verbunden haben. CodeCommit Anweisungen finden Sie unter Herstellen einer Verbindung mit einem Repository.

1. Um den Branch aus dem lokalen Repository zu löschen, führen Sie den git branch -D ***branch-name*** Befehl aus, wobei *Branch-Name der Name* des Branches ist, den Sie löschen möchten.

   > **Tip**
   >
   > Um eine Liste der Branch-Namen zu erhalten, führen Sie den Befehl git branch --all aus.

   Um beispielsweise einen Zweig im lokalen Repository mit dem Namen zu löschen: `MyNewBranch`

   ```
   git branch -D MyNewBranch
   ```

2. Um den Branch aus dem CodeCommit Repository zu löschen, führen Sie den git push ***remote-name*** --delete ***branch-name*** Befehl aus, wobei *remote-name* der Spitzname ist, den das lokale Repository für das CodeCommit Repository verwendet, und *branch-name der Name des Branches* ist, den Sie aus dem Repository löschen möchten. CodeCommit

   > **Tip**
   >
   > Um eine Liste der CodeCommit Repository-Namen und ihrer URLs zu erhalten, führen Sie den Befehl aus. git remote -v

   Um beispielsweise einen Branch zu löschen, der `MyNewBranch` im CodeCommit Repository wie folgt benannt ist`origin`:

   ```
   git push origin --delete MyNewBranch
   ```

   > **Tip**
   >
   > Mit diesem Befehl wird der Branch nicht gelöscht, wenn es sich dabei um den Standard-Branch handelt.

Weitere Optionen findest du in deiner Git-Dokumentation.

Arbeiten mit Benutzereinstellungen

Einige der Standardeinstellungen können in der AWS CodeCommit-Konsole konfiguriert werden. Beispielsweise können Sie Ihre Einstellungen für die Anzeige der Codeänderungen als inline oder in einer geteilten Ansicht festlegen. Wenn Sie an einer dieser Einstellungen eine Änderung vornehmen, setzt die AWS CodeCommit-Konsole in Ihrem Browser einen Cookie, durch den Ihre Auswahl automatisch gespeichert und bei jeder weiteren Verwendung der Konsole erneut verwendet wird. Diese Einstellungen gelten jederzeit für alle Repositorys in allen Regionen und werden bei jedem Zugriff auf die AWS CodeCommit-Konsole durch diesen Browser angewendet. Diese Einstellungen gelten weder nur für ein bestimmtes Repository noch für eine bestimmte Region. Sie haben keine Auswirkung auf Ihre Interaktionen mit der AWS CLI, der AWS CodeCommit-API oder anderen Services, die mit AWS CodeCommit interagieren.

> **Note**
>
> Die Cookies für die Benutzereinstellungen sind browserspezifisch. Wenn Sie die Cookies in Ihrem Browser löschen, werden auch die Einstellungen gelöscht. Greifen Sie mit einem anderen Browser auf ein Repository zu, haben Sie mit diesem Browser keinen Zugriff auf die Cookies des ursprünglichen Browsers. Ihre Einstellungen werden nicht gespeichert.

Benutzereinstellungen umfassen:

- Ob beim Anzeigen von Codeänderungen die Ansicht Unified oder Split verwendet und ob geänderte Leerzeichen ein- oder ausgeblendet werden sollen.
- Ob im Code-Editorfenster beim Anzeigen, Bearbeiten oder Erstellen von Code, ein heller oder dunkler Hintergrund verwendet werden soll.

Es gibt keine spezielle Seite, um all Ihre Einstellungen festzulegen. Stattdessen werden Ihre Änderungen an den Einstellungen in der Konsole, beispielsweise wie Sie Codeänderungen anzeigen möchten, gespeichert und nach Bedarf angewendet.

Migrieren zu AWS CodeCommit

Sie können ein Git-Repository auf verschiedene Arten in ein CodeCommit Repository migrieren: Sie klonen es, spiegeln es, migrieren alle oder nur einige Branches usw. Auch lokale und nicht versionierte Inhalte von Ihrem Computer können Sie in CodeCommit migrieren.

In den folgenden Themen werden einige Methoden für die Repository-Migration vorgestellt. Ihre Schritte sind möglicherweise anders, dies ist abhängig vom Typ, von der Art oder der Komplexität Ihres Repositorys und von den Entscheidungen, die Sie in Bezug auf die zu migrierenden Inhalte und die Migrationsmethode getroffen haben. Bei sehr umfangreichen Repositorys kann eine inkrementelle Migration sinnvoll sein.

> **Note**
>
> Sie können von anderen Versionskontrollsystemen (wie Perforce, Subversion oder TFS) zu CodeCommit migrieren, dafür ist jedoch zuerst eine Migration in Git notwendig.
> Weitere Optionen finden Sie in der Git-Dokumentation.
> Alternativ können Sie die Informationen für eine Migration in Git im Buch Pro Git von Scott Chacon und Ben Straub heranziehen.

Themen

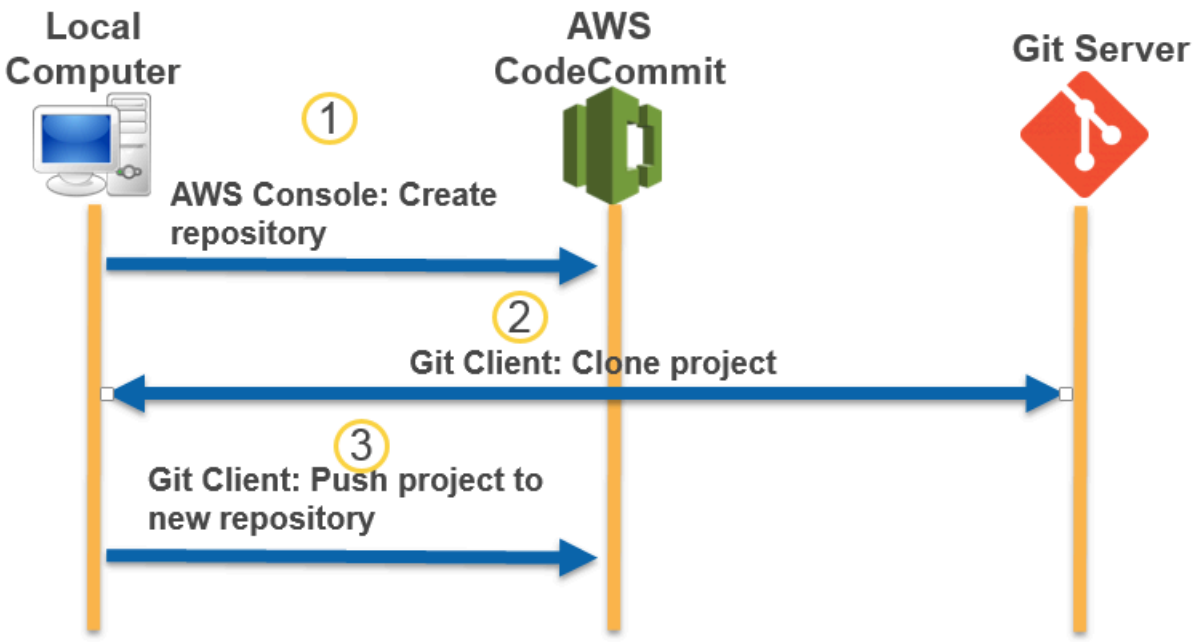
- Migrieren Sie ein Git-Repository zu AWS CodeCommit
- Migrieren Sie lokalen oder unversionierten Inhalt zu AWS CodeCommit
- Ein Repository inkrementell migrieren

Migrieren Sie ein Git-Repository zu AWS CodeCommit

Sie können ein vorhandenes Git-Repository in ein CodeCommit Repository migrieren. Das Verfahren in diesem Thema führt Sie durch die Migration eines auf einem anderen Git-Repository gehosteten Projekts zu CodeCommit. Als Teil dieses Prozesses werden Sie Folgendes durchführen:

- Schließen Sie die Ersteinrichtung ab, die für erforderlich ist CodeCommit.
- Erstellen Sie ein CodeCommit Repository.
- Klonen Sie das Repository und übertragen Sie es dorthin CodeCommit.

- Dateien im CodeCommit Repository anzeigen.
- Teilen Sie das CodeCommit Repository mit Ihrem Team.



Themen

- Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit
- Schritt 1: Erstellen Sie ein Repository CodeCommit
- Schritt 2: Das Repository klonen und zum CodeCommit Repository pushen
- Schritt 3: Dateien anzeigen in CodeCommit
- Schritt 4: Teilen Sie das Repository CodeCommit

Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit

Bevor Sie ein Repository zu migrieren können CodeCommit, müssen Sie einen IAM-Benutzer für den Zugriff erstellen und konfigurieren CodeCommit und Ihren lokalen Computer für den Zugriff konfigurieren. Außerdem sollten Sie die AWS CLI für die Verwaltung von CodeCommit installieren. Obwohl Sie die meisten CodeCommit Aufgaben auch ohne Git ausführen können, AWS CLI bietet es Flexibilität, wenn Sie mit Git über die Befehlszeile oder das Terminal arbeiten.

Wenn Sie bereits dafür eingerichtet sind CodeCommit, können Sie direkt zuSchritt 1: Erstellen Sie ein Repository CodeCommit .

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu http://aws.amazon.com gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen.

   > **Note**
   >
   > CodeCommit erfordertAWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter AWS KMS und Verschlüsselung.

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie AWSCodeCommitPowerUseraus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit.

   Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter Teilen Sie ein Repository und. Authentifizierung und Zugriffskontrolle für AWS CodeCommit

So installieren und konfigurieren Sie die AWS CLI

1. Laden Sie die AWS CLI auf den lokalen Computer herunter und installieren Sie. Dies ist eine Voraussetzung für die Interaktion mit über die CodeCommit Befehlszeile. Wir empfehlen, AWS CLI Version 2 zu installieren. Es ist die neueste Hauptversion von AWS CLI und unterstützt alle aktuellen Funktionen. Es ist die einzige Version vonAWS CLI, die die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mit git-remote-codecommit unterstützt.

   Weitere Informationen finden Sie unter Erste Schritte mit der AWS-Befehlszeilenschnittstelle.

   > **Note**
   >
   > CodeCommit funktioniert nur mit den AWS CLI Versionen 1.7.38 und höher. Als bewährte Methode installieren oder aktualisieren Sie die AWS CLI auf die neueste verfügbare Version. Führen Sie den Befehl aws --version aus, um zu überprüfen, welche Version der AWS CLI installiert ist.
   > Informationen zum Upgraden von einer älteren auf die aktuelle Version der AWS CLI finden Sie unter Installieren der AWS Command Line Interface.

2. Führen Sie diesen Befehl aus, um zu überprüfen, ob die CodeCommit Befehle für installiert AWS CLI sind.

   ```
   aws codecommit help
   ```

   Dieser Befehl gibt eine Liste von CodeCommit Befehlen zurück.

3. Konfigurieren Sie das AWS CLI mit einem Profil, indem Sie den configure Befehl verwenden, wie folgt:.

   ```
   aws configure
   ```

Wenn Sie dazu aufgefordert werden, geben Sie den AWS Zugriffsschlüssel und den AWS geheimen Zugriffsschlüssel des IAM-Benutzers an, mit CodeCommit dem Sie ihn verwenden möchten. Stellen Sie außerdem sicher, dass Sie angeben, AWS-Region wo sich das Repository befindet, z. B. `us-east-2` Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

```
AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press
 Enter
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and
 then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then
 press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen für die AWS CLI finden Sie unter:

- Benannte Profile
- Verwenden einer IAM-Rolle in der AWS CLI
- Befehl „Set“
- Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen Repository herzustellenAWS-Region, müssen Sie das AWS CLI mit dem Standardregionsnamen neu konfigurieren. Zu den unterstützten Standardregionsnamen für CodeCommit gehören:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1
- ap-southeast-2

- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3
- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und finden AWS-Region Sie unterRegionen und Git-Verbindungsendpunkte. Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter Wie erhalte ich Anmeldeinformationen? und Zugriffsschlüssel für IAM-Benutzer verwalten. Weitere Informationen zur AWS CLI und zu Profilen finden Sie unter Benannte Profile.

Als Nächstes müssen Sie Git installieren.

- Für Linux, macOS oder Unix:

Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

Um Git zu installieren, empfehlen wir Websites wie Git Downloads.

> **Note**
>
> Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter anFehlerbehebung.

- Für Windows:

  Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

  Um Git zu installieren, empfehlen wir Websites wie Git für Windows. Wenn du diesen Link verwendest, um Git zu installieren, kannst du alle Standardeinstellungen der Installation akzeptieren, mit Ausnahme der folgenden:

  - Wenn Sie während des Schritts Anpassen Ihrer PATH-Umgebung dazu aufgefordert werden, wählen Sie in der Befehlszeile die Option, Git zu verwenden.
  - (Optional) Wenn Sie beabsichtigen, HTTPS mit dem Credential Helper zu verwenden, der in AWS CLI statt der Konfiguration von Git-Anmeldeinformationen für enthalten ist CodeCommit, auf der Seite Zusätzliche Optionen konfigurieren, stellen Sie sicher, dass die Option Git Credential Manager aktivieren deaktiviert ist. Der Git Credential Manager ist nur kompatibel, CodeCommit wenn IAM-Benutzer Git-Anmeldeinformationen konfigurieren. Weitere Informationen finden Sie unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden und Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert (403).

> **Note**
>
> Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter anFehlerbehebung.

CodeCommit unterstützt sowohl HTTPS- als auch SSH-Authentifizierung. Um die Installation abzuschließen, müssen Sie Git-Anmeldeinformationen für CodeCommit (HTTPS, für die meisten Benutzer empfohlen), ein SSH-Schlüsselpaar für den Zugriff CodeCommit (SSH) git-remote-codecommit (empfohlen für Benutzer, die Verbundzugriff verwenden) oder den AWS CLI in (HTTPS) enthaltenen Credential Helper konfigurieren.

- Informationen zu Git-Anmeldeinformationen auf allen unterstützten Betriebssystemen finden Sie unter Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit.
- Informationen zu SSH unter Linux, macOS oder Unix finden Sie unterSSH und Linux, macOS oder Unix: Richte die öffentlichen und privaten Schlüssel für Git ein undCodeCommit.
- Informationen zu SSH unter Windows finden Sie unter Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein.
- Informationen für git-remote-codecommit finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit.
- Informationen zum Credential Helper unter Linux, macOS oder Unix finden Sie unter Credential Helper einrichten (Linux, macOS oder Unix).
- Informationen zum Hilfsprogramm für Anmeldeinformationen unter Windows finden Sie unter Einrichten des Hilfsprogramms für Anmeldeinformationen (Windows).

Schritt 1: Erstellen Sie ein Repository CodeCommit

In diesem Abschnitt verwenden Sie die CodeCommit Konsole, um das CodeCommit Repository zu erstellen, das Sie für den Rest dieses Tutorials verwenden. Informationen über die Verwendung der AWS CLI zum Erstellen des Repositorys finden Sie unter Erstellen Sie ein Repository ()AWS CLI.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.

2. Wählen Sie in der Regionsauswahl den AWS-Region Ort aus, an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Wählen Sie auf der Seite Repositorys die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) im Feld Repository name (Repository-Name) einen Namen für das Repository ein.

   > **Note**
   >
   > Bei den Repository-Namen wird nach Groß- und Kleinschreibung unterschieden. Der Name muss AWS-Region für Ihr Amazon Web Services Services-Konto eindeutig sein.

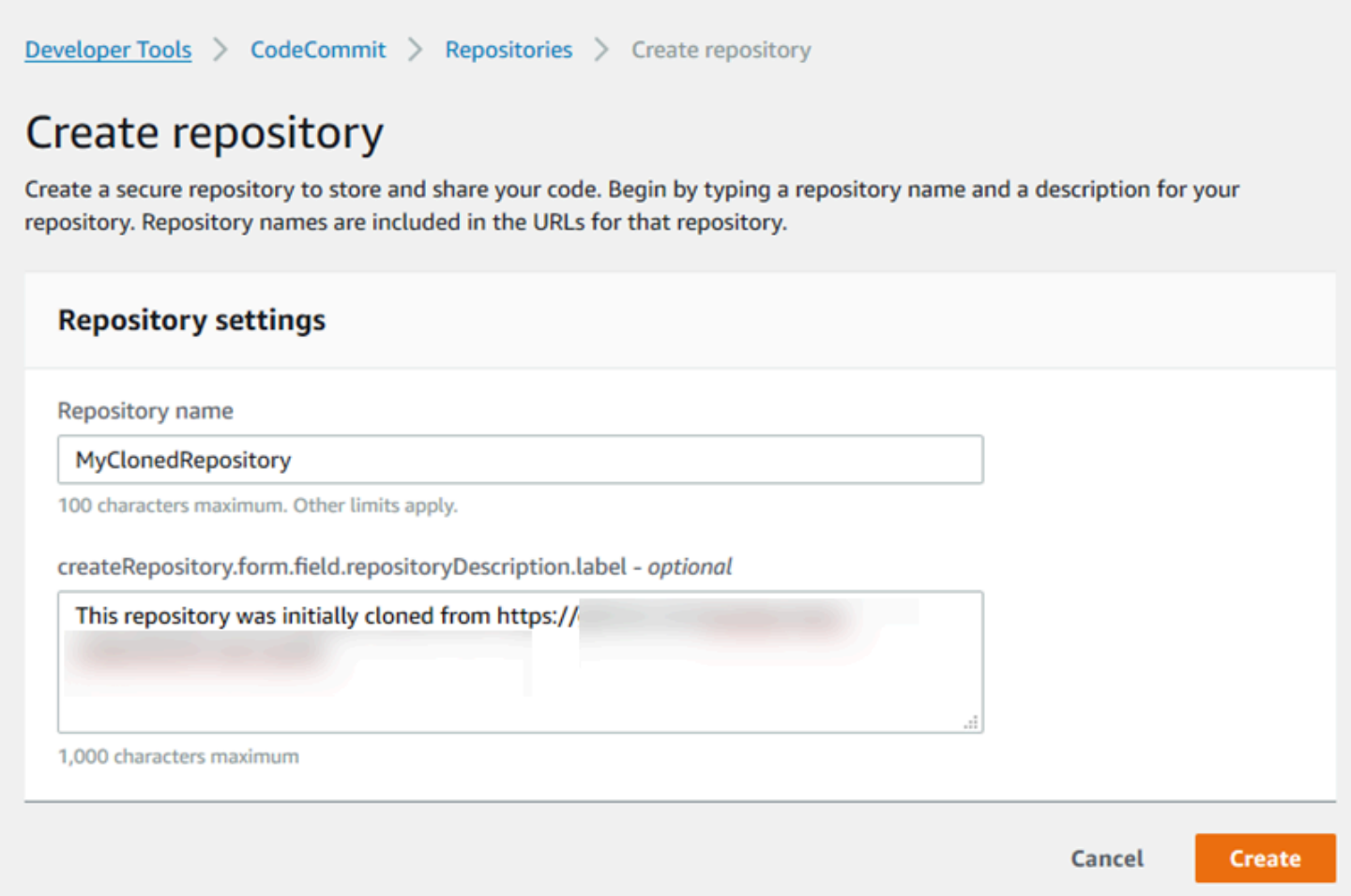
5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung für das Repository ein. Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.

   > **Note**
   >
   > Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie ein Anwendungsentwickler sind, der die `BatchGetRepositories` APIs `GetRepository` oder verwendet und planen, das Repository-Beschreibungsfeld in einem Webbrowser anzuzeigen, finden Sie weitere Informationen in der CodeCommit API-Referenz.

6. (Optional) Wählen Sie Add tag (Tag hinzufügen) aus, um ein oder mehrere Repository-Tags (ein benutzerdefiniertes Attribut-Label, mit dem Sie Ihre AWS-Ressourcen organisieren und verwalten), zu Ihrem Repository hinzuzufügen. Weitere Informationen finden Sie unter Kennzeichnen von Repositorys in AWS CodeCommit.
7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln von Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorys.
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer ihn analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um

Codefehler zu finden und Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.

9. Wählen Sie Erstellen aus.



Nachdem das Repository erstellt wurde, wird es in der Liste Repositories angezeigt. Wählen Sie in der URL-Spalte das Kopiersymbol aus und wählen Sie dann das Protokoll (SSH oder HTTPS) aus, das Sie zum Herstellen der Verbindung mit CodeCommit verwenden möchten. Kopieren Sie die URL.

Wenn Sie beispielsweise Ihrem Repository einen Namen gegeben haben *MyClonedRepository*und Git-Anmeldeinformationen mit HTTPS in der Region USA Ost (Ohio) verwenden, sieht die URL wie folgt aus:

```
https://git-codecommit.us-east-2.amazonaws.com/MyClonedRepository
```

Sie benötigen diese URL später in Schritt 2: Das Repository klonen und zum CodeCommit Repository pushen.

Schritt 2: Das Repository klonen und zum CodeCommit Repository pushen

In diesem Abschnitt klonen Sie ein Git-Repository auf Ihren lokalen Computer und erstellen somit ein lokales Repository. Anschließend übertragen Sie den Inhalt des lokalen Repos in das CodeCommit Repository, das Sie zuvor erstellt haben.

1. Führen Sie über das Terminal oder die Befehlszeile auf Ihrem lokalen Computer den git clone Befehl mit der `--mirror` Option aus, eine bloße Kopie des Remote-Repositorys in einen neuen Ordner mit dem Namen *aws-codecommit-demo*zu klonen. Hierbei handelt es sich um ein einfaches Repository nur für die Migration. Es ist nicht das lokale Repo für die Interaktion mit dem migrierten Repository in. CodeCommit Sie können es später erstellen, nachdem die Migration zu abgeschlossen CodeCommit ist.

   Im folgenden Beispiel wird eine auf GitHub (*https://github.com/awslabs/ aws-demo-php-simple -app.git*) gehostete Demo-Anwendung in ein lokales Repo in einem Verzeichnis mit dem Namen geklont. *aws-codecommit-demo*

   ```
   git clone --mirror https://github.com/awslabs/aws-demo-php-simple-app.git aws-codecommit-demo
   ```

2. Ändern Sie die Verzeichnisse in das Verzeichnis, in dem Sie den Klon erstellt haben.

   ```
   cd aws-codecommit-demo
   ```

3. Führen Sie den git push Befehl aus und geben Sie die URL und den Namen des CodeCommit Ziel-Repositorys sowie die Option an. --all (Dies ist die URL, die Sie in Schritt 1: Erstellen Sie ein Repository CodeCommit kopiert haben.)

   Wenn Sie beispielsweise Ihr Repository benannt haben *MyClonedRepository*und für die Verwendung von HTTPS eingerichtet sind, würden Sie den folgenden Befehl ausführen:

   ```
   git push https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyClonedRepository
    --all
   ```

   > **Note**
   >
   > Die Option --all überträgt per Push ausschließlich alle Branches für das Repository. Andere Referenzen, z. B. Tags, werden nicht übertragen. Wenn Sie Tags per Push

übertragen möchten, warten Sie, bis der ursprüngliche Push-Vorgang abgeschlossen ist, und führen Sie einen weiteren aus, diesmal unter Verwendung der Option --tags:

```
git push ssh://git-codecommit.us-east-2.amazonaws.com/v1/
repos/MyClonedRepository --tags
```

Weitere Informationen finden Sie unter Git Push auf der Git-Website. Informationen zur Push-Übertragung großer Repositorys, insbesondere wenn alle Referenzen gleichzeitig per Push übertragen werden sollen, (z. B. mit der Option --mirror), finden Sie unter Migrieren Sie ein Repository schrittweise.

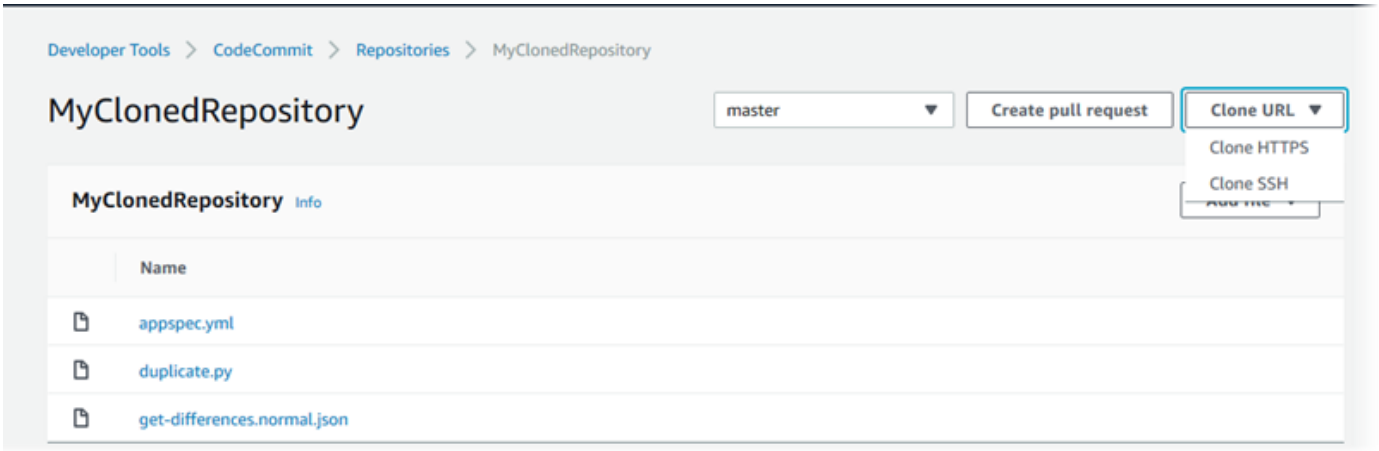
Sie können den *aws-codecommit-demo*Ordner und seinen Inhalt löschen, nachdem Sie das Repository zu CodeCommit migriert haben. Um ein lokales Repo mit allen korrekten Referenzen für die Arbeit mit dem Repository zu erstellen CodeCommit, führen Sie den `git clone` Befehl ohne die `--mirror` Option aus:

```
git clone https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyClonedRepository
```

Schritt 3: Dateien anzeigen in CodeCommit

Nachdem Sie den Inhalt Ihres Verzeichnisses per Push übertragen haben, können Sie die CodeCommit Konsole verwenden, um schnell alle Dateien in diesem Repository anzuzeigen.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories den Namen des Repositorys aus (z. B. *MyClonedRepository*).
3. Zeigen Sie die Dateien im Repository an, um die Branches, die Klon-URLs, die Einstellungen und vieles mehr zu sehen.



Schritt 4: Teilen Sie das Repository CodeCommit

Wenn Sie ein Repository in erstellen CodeCommit, werden zwei Endpunkte generiert: einer für HTTPS-Verbindungen und einer für SSH-Verbindungen. Beide bieten sichere Verbindungen über ein Netzwerk. Ihre Benutzer können beide Protokolle verwenden. Beide Endpunkte bleiben aktiv, unabhängig davon, welches Protokoll Sie Ihren Benutzern empfehlen. Bevor Sie Ihr Repository mit anderen teilen können, müssen Sie IAM-Richtlinien erstellen, die anderen Benutzern Zugriff auf Ihr Repository gewähren. Stellen Sie Ihren Benutzern diese Zugriffsanweisungen zur Verfügung.

Erstellen einer kundenverwalteten Richtlinie für das Repository

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Dashboard die Optionen Policies und dann Create Policy aus.
3. Wählen Sie auf der Seite „Richtlinie erstellen“ die Option Verwaltete Richtlinie importieren aus.
4. Geben Sie auf der Seite Verwaltete Richtlinien importieren im Feld Richtlinien filtern den Text ein**`AWSCodeCommitPowerUser`**. Klicken Sie auf die Schaltfläche neben dem Richtliniennamen und wählen Sie dann Importieren aus.
5. Wählen Sie auf der Seite Create policy (Richtlinie erstellen) die Option JSON aus. Ersetzen Sie den Teil „*“ in der `Resource` Zeile für CodeCommit Aktionen durch den Amazon-Ressourcennamen (ARN) des CodeCommit Repositorys, wie hier gezeigt:

```
"Resource": [
 "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"
 ]
```

> **Tip**
>
> Um den ARN für das CodeCommit Repository zu finden, gehen Sie zur CodeCommit Konsole, wählen Sie den Repository-Namen aus der Liste aus und wählen Sie dann Einstellungen. Weitere Informationen finden Sie unter Repository-Details anzeigen.

Wenn Sie diese Richtlinie auf mehr als ein Repository anwenden möchten, fügen Sie jedes Repository als Ressource hinzu, indem Sie seinen ARN angeben. Fügen Sie zwischen den Ressourcenanweisungen ein Komma ein, wie hier dargestellt:

```
"Resource": [
 "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
 "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo"
 ]
```

Wenn Sie mit der Bearbeitung fertig sind, wählen Sie „Richtlinie überprüfen“.

6. Geben Sie auf der Seite „Richtlinie überprüfen“ im Feld Name einen neuen Namen für die Richtlinie ein (z. B. *AWSCodeCommitPowerUser- MyDemoRepo*). Geben Sie optional eine Beschreibung für diese Richtlinie ein.
7. Wählen Sie Create Policy (Richtlinie erstellen) aus.

Um den Zugriff auf Ihr Repository zu verwalten, erstellen Sie eine IAM-Gruppe für deren Benutzer, fügen Sie dieser Gruppe IAM-Benutzer hinzu und fügen Sie dann die vom Kunden verwaltete Richtlinie hinzu, die Sie im vorherigen Schritt erstellt haben. Fügen Sie alle anderen Richtlinien hinzu, die für den Zugriff erforderlich sind, z. B. IAMuserSSHKeys oder. IAMSelfManageServiceSpecificCredentials

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Dashboard die Option Groups und dann Create New Group aus.
3. Geben Sie auf der Seite Set Group Name (Name für Gruppe festlegen) im Feld Group Name (Gruppenname) einen Namen für die Gruppe ein (zum Beispiel *MyDemoRepoGroup*) und wählen Sie dann Next Step (Nächster Schritt) aus. Sie sollten den Repository-Namen als Teil des Gruppennamens einfügen.

> **Note**
>
> Dieser Name muss in einem Amazon Web Services Services-Konto eindeutig sein.

4. Aktivieren Sie das Kontrollkästchen neben der kundenverwalteten Richtlinie, die Sie im vorherigen Abschnitt erstellt haben (z. B. AWSCodeCommitPowerUser-MyDemoRepo).
5. Klicken Sie auf der Seite Review auf Create Group. IAM erstellt diese Gruppe mit den angegebenen Richtlinien, die bereits angehängt sind. Die Gruppe erscheint in der Liste der Gruppen, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind.
6. Wählen Sie Ihre Gruppe in der Liste aus.
7. Wählen Sie auf der Gruppenübersichtsseite die Registerkarte Users und anschließend die Option Add Users to Groups aus. Markieren Sie in der Liste, in der alle mit Ihrem Amazon Web Services Services-Konto verknüpften Benutzer angezeigt werden, die Kästchen neben den Benutzern, denen Sie Zugriff auf das CodeCommit Repository gewähren möchten, und wählen Sie dann Benutzer hinzufügen.

> **Tip**
>
> Sie können über das Suchfeld schnell nach Benutzern anhand ihres Namens suchen.

8. Wenn Sie Ihre Benutzer hinzugefügt haben, schließen Sie die IAM-Konsole.

Nachdem Sie mit der von Ihnen konfigurierten Richtliniengruppe und den Richtlinien einen IAM-Benutzer für den CodeCommit Zugriff erstellt haben, senden Sie diesem Benutzer die Informationen, die für die Verbindung mit dem Repository erforderlich sind.

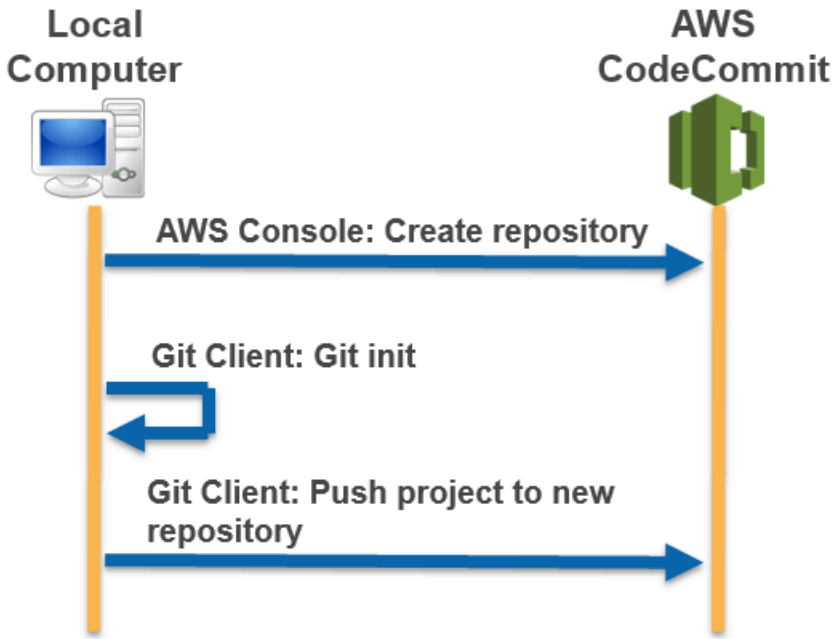
1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositorys sind spezifisch für ein. AWS-Region Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Wählen Sie auf der Seite Repositories (Repositorys) das Repository aus, das Sie freigeben möchten.
4. Wählen Sie unter Clone URL (Klon-URL) das Protokoll aus, das die Benutzer verwenden sollen. Dadurch wird die Klon-URL für das Verbindungsprotokoll kopiert.

5. Übermitteln Sie den Benutzern die Klon-URL und andere erforderliche Anweisungen, wie z. B. zum Installieren der AWS CLI, Konfigurieren eines Profils oder Installieren von Git. Stellen Sie sicher, dass Sie die Konfigurationsinformationen für das Verbindungsprotokoll (z. B. HTTPS) angeben.

Migrieren Sie lokalen oder unversionierten Inhalt zu AWS CodeCommit

Die Verfahren in diesem Thema zeigen Ihnen, wie Sie ein vorhandenes Projekt oder lokale Inhalte auf Ihrem Computer in ein CodeCommit Repository migrieren. Als Teil dieses Prozesses werden Sie Folgendes durchführen:

- Schließen Sie die Ersteinrichtung ab, die für erforderlich ist CodeCommit.
- Erstellen Sie ein CodeCommit Repository.
- Platzieren Sie einen lokalen Ordner unter der Git-Versionskontrolle und übertragen Sie den Inhalt dieses Ordners in das CodeCommit Repository.
- Dateien im CodeCommit Repository anzeigen.
- Teilen Sie das CodeCommit Repository mit Ihrem Team.



Themen

- Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit
- Schritt 1: Erstellen Sie ein Repository CodeCommit
- Schritt 2: Migrieren Sie lokale Inhalte in das CodeCommit Repository
- Schritt 3: Dateien anzeigen in CodeCommit
- Schritt 4: Teilen Sie das Repository CodeCommit

Schritt 0: Für den Zugriff auf ist eine Einrichtung erforderlich CodeCommit

Bevor Sie lokale Inhalte migrieren können CodeCommit, müssen Sie einen IAM-Benutzer für den Zugriff erstellen und konfigurieren CodeCommit und Ihren lokalen Computer für den Zugriff konfigurieren. Außerdem sollten Sie die AWS CLI für die Verwaltung von CodeCommit installieren. Obwohl Sie die meisten CodeCommit Aufgaben auch ohne Git ausführen können, AWS CLI bietet es Flexibilität bei der Arbeit mit Git.

Wenn Sie bereits dafür eingerichtet sind CodeCommit, können Sie direkt zuSchritt 1: Erstellen Sie ein Repository CodeCommit .

Um einen IAM-Benutzer für den Zugriff zu erstellen und zu konfigurieren CodeCommit

1. Erstellen Sie ein Amazon Web Services Services-Konto, indem Sie zu http://aws.amazon.com gehen und Sign Up wählen.
2. Erstellen Sie einen IAM-Benutzer oder verwenden Sie einen vorhandenen in Ihrem Amazon Web Services Services-Konto. Stellen Sie sicher, dass Sie über eine Zugriffsschlüssel-ID und einen geheimen Zugriffsschlüssel verfügen, die diesem IAM-Benutzer zugeordnet sind. Weitere Informationen finden Sie unter Einen IAM-Benutzer in Ihrem Amazon Web Services Services-Konto erstellen.

> **Note**
>
> CodeCommit erfordertAWS Key Management Service. Wenn Sie einen vorhandenen IAM-Benutzer verwenden, stellen Sie sicher, dass dem Benutzer keine Richtlinien zugeordnet sind, die ausdrücklich die von CodeCommit erforderlichen AWS KMS Aktionen verweigern. Weitere Informationen finden Sie unter AWS KMS und Verschlüsselung.

3. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
4. Wählen Sie in der IAM-Konsole im Navigationsbereich Benutzer und dann den IAM-Benutzer aus, den Sie für den Zugriff konfigurieren möchten. CodeCommit
5. Wählen Sie auf der Registerkarte Permissions die Option Add Permissions.
6. Wählen Sie unter Grant permissions die Option Attach existing policies directly aus.
7. Wählen Sie AWSCodeCommitPowerUseraus der Liste der Richtlinien eine andere verwaltete Richtlinie für CodeCommit den Zugriff aus. Weitere Informationen finden Sie unter Von AWS verwaltete Richtlinien für CodeCommit.

   Nachdem Sie die Richtlinie ausgewählt haben, die Sie anhängen möchten, wählen Sie Weiter: Überprüfen aus, um die Liste der Richtlinien zu überprüfen, die an den IAM-Benutzer angehängt werden sollen. Ist die Liste korrekt, wählen Sie Add permissions aus.

   Weitere Informationen zu CodeCommit verwalteten Richtlinien und zur gemeinsamen Nutzung des Zugriffs auf Repositorys mit anderen Gruppen und Benutzern finden Sie unter Teilen Sie ein Repository und. Authentifizierung und Zugriffskontrolle für AWS CodeCommit

So installieren und konfigurieren Sie die AWS CLI

1. Laden Sie die AWS CLI auf den lokalen Computer herunter und installieren Sie. Dies ist eine Voraussetzung für die Interaktion mit über die CodeCommit Befehlszeile. Wir empfehlen, AWS CLI Version 2 zu installieren. Es ist die neueste Hauptversion von AWS CLI und unterstützt alle aktuellen Funktionen. Es ist die einzige Version vonAWS CLI, die die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mit git-remote-codecommit unterstützt.

   Weitere Informationen finden Sie unter Erste Schritte mit der AWS-Befehlszeilenschnittstelle.

   > **Note**
   >
   > CodeCommit funktioniert nur mit den AWS CLI Versionen 1.7.38 und höher. Als bewährte Methode installieren oder aktualisieren Sie die AWS CLI auf die neueste verfügbare Version. Führen Sie den Befehl aws --version aus, um zu überprüfen, welche Version der AWS CLI installiert ist.

> Informationen zum Upgraden von einer älteren auf die aktuelle Version der AWS CLI finden Sie unter Installieren der AWS Command Line Interface.

2. Führen Sie diesen Befehl aus, um zu überprüfen, ob die CodeCommit Befehle für installiert AWS CLI sind.

```
aws codecommit help
```

Dieser Befehl gibt eine Liste von CodeCommit Befehlen zurück.

3. Konfigurieren Sie das AWS CLI mit einem Profil, indem Sie den configure Befehl verwenden, wie folgt:.

```
aws configure
```

Wenn Sie dazu aufgefordert werden, geben Sie den AWS Zugriffsschlüssel und den AWS geheimen Zugriffsschlüssel des IAM-Benutzers an, mit CodeCommit dem Sie ihn verwenden möchten. Stellen Sie außerdem sicher, dass Sie angeben, AWS-Region wo sich das Repository befindet, z. B. `us-east-2` Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

```
AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press
 Enter
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and
 then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then
 press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen für die AWS CLI finden Sie unter:

- Benannte Profile
- Verwenden Sie eine IAM-Rolle in der AWS CLI
- Befehl „Set“
- Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen Repository herzustellenAWS-Region, müssen Sie das AWS CLI mit dem Standardregionsnamen neu konfigurieren. Zu den unterstützten Standardregionsnamen für CodeCommit gehören:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1
- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3

- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und finden AWS-Region Sie unterRegionen und Git-Verbindungsendpunkte. Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter Wie erhalte ich Anmeldeinformationen? und Zugriffsschlüssel für IAM-Benutzer verwalten. Weitere Informationen zur AWS CLI und zu Profilen finden Sie unter Benannte Profile.

Als Nächstes müssen Sie Git installieren.

- Für Linux, macOS oder Unix:

  Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

  Um Git zu installieren, empfehlen wir Websites wie Git Downloads.

  > **Note**
  >
  > Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter anFehlerbehebung.

- Für Windows:

  Um mit Dateien, Commits und anderen Informationen in CodeCommit Repositorys zu arbeiten, müssen Sie Git auf Ihrem lokalen Computer installieren. CodeCommit unterstützt Git-Versionen 1.7.9 und höher. Git Version 2.28 unterstützt die Konfiguration des Branchnamens für anfängliche Commits. Wir empfehlen die Verwendung einer aktuellen Version von Git.

  Um Git zu installieren, empfehlen wir Websites wie Git für Windows. Wenn du diesen Link verwendest, um Git zu installieren, kannst du alle Standardeinstellungen der Installation akzeptieren, mit Ausnahme der folgenden:

- Wenn Sie während des Schritts Anpassen Ihrer PATH-Umgebung dazu aufgefordert werden, wählen Sie in der Befehlszeile die Option, Git zu verwenden.
- (Optional) Wenn Sie beabsichtigen, HTTPS mit dem Credential Helper zu verwenden, der in AWS CLI statt der Konfiguration von Git-Anmeldeinformationen für enthalten ist CodeCommit, auf der Seite Zusätzliche Optionen konfigurieren, stellen Sie sicher, dass die Option Git Credential Manager aktivieren deaktiviert ist. Der Git Credential Manager ist nur kompatibel, CodeCommit wenn IAM-Benutzer Git-Anmeldeinformationen konfigurieren. Weitere Informationen finden Sie unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden und Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert (403).

> **Note**
>
> Git ist eine sich entwickelnde, regelmäßig aktualisierte Plattform. Gelegentlich kann sich eine Änderung der Funktionen auf die Art und Weise auswirken, mit der es funktioniert CodeCommit. Wenn du Probleme mit einer bestimmten Version von Git und hast CodeCommit, sieh dir die Informationen unter anFehlerbehebung.

CodeCommit unterstützt sowohl HTTPS- als auch SSH-Authentifizierung. Um die Installation abzuschließen, müssen Sie die Git-Anmeldeinformationen für CodeCommit (HTTPS, für die meisten Benutzer empfohlen), ein SSH-Schlüsselpaar (SSH) für den Zugriff git-remote-codecommit (empfohlen für Benutzer CodeCommit, die Verbundzugriff verwenden) oder den in der enthaltenen Credential Helper konfigurieren. AWS CLI

- Informationen zu Git-Anmeldeinformationen auf allen unterstützten Betriebssystemen finden Sie unter Schritt 3: Erstellen Sie Git-Anmeldeinformationen für HTTPS-Verbindungen zu CodeCommit.
- Informationen zu SSH unter Linux, macOS oder Unix finden Sie unterSSH und Linux, macOS oder Unix: Richte die öffentlichen und privaten Schlüssel für Git ein undCodeCommit.
- Informationen zu SSH unter Windows finden Sie unter Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein.
- Informationen für git-remote-codecommit finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit.
- Informationen zum Credential Helper unter Linux, macOS oder Unix finden Sie unter Credential Helper einrichten (Linux, macOS oder Unix).

- Informationen zum Hilfsprogramm für Anmeldeinformationen unter Windows finden Sie unter Einrichten des Hilfsprogramms für Anmeldeinformationen (Windows).

Schritt 1: Erstellen Sie ein Repository CodeCommit

In diesem Abschnitt verwenden Sie die CodeCommit Konsole, um das CodeCommit Repository zu erstellen, das Sie für den Rest dieses Tutorials verwenden. Informationen über die Verwendung der AWS CLI zum Erstellen des Repositorys finden Sie unter Erstellen Sie ein Repository ()AWS CLI.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl den AWS-Region Ort aus, an dem Sie das Repository erstellen möchten. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Wählen Sie auf der Seite Repositorys die Option Repository auswählen.
4. Geben Sie auf der Seite Create repository (Repository erstellen) im Feld Repository name (Repository-Name) einen Namen für das Repository ein.

   > **Note**
   >
   > Bei den Repository-Namen wird nach Groß- und Kleinschreibung unterschieden. Der Name muss AWS-Region für Ihr Amazon Web Services Services-Konto eindeutig sein.

5. (Optional) Geben Sie unter Description (Beschreibung) eine Beschreibung für das Repository ein. Dadurch können Sie und andere Benutzer den Zweck des Repositorys leichter erkennen.

   > **Note**
   >
   > Das Beschreibungsfeld zeigt Markdown in der Konsole an und akzeptiert alle HTML-Zeichen und gültigen Unicode-Zeichen. Wenn Sie ein Anwendungsentwickler sind, der die `BatchGetRepositories` APIs `GetRepository` oder verwendet und planen, das Repository-Beschreibungsfeld in einem Webbrowser anzuzeigen, finden Sie weitere Informationen in der CodeCommit API-Referenz.

6. (Optional) Wählen Sie Add tag (Tag hinzufügen) aus, um ein oder mehrere Repository-Tags (ein benutzerdefiniertes Attribut-Label, mit dem Sie Ihre AWS-Ressourcen organisieren

und verwalten), zu Ihrem Repository hinzuzufügen. Weitere Informationen finden Sie unter Kennzeichnen von Repositorys in AWS CodeCommit.

7. (Optional) Erweitern Sie Zusätzliche Konfiguration, um anzugeben, ob Sie den Standardschlüssel Von AWS verwalteter Schlüssel oder Ihren eigenen vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln von Daten in diesem Repository verwenden möchten. Wenn Sie Ihren eigenen, vom Kunden verwalteten Schlüssel verwenden möchten, müssen Sie sicherstellen, dass er dort verfügbar ist, AWS-Region wo Sie das Repository erstellen, und dass der Schlüssel aktiv ist. Weitere Informationen finden Sie unter AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorys.
8. (Optional) Wählen Sie Amazon CodeGuru Reviewer für Java und Python aktivieren, wenn dieses Repository Java- oder Python-Code enthält und Sie möchten, dass CodeGuru Reviewer ihn analysiert. CodeGuru Reviewer verwendet mehrere Modelle für maschinelles Lernen, um Codefehler zu finden und Verbesserungen und Korrekturen in Pull-Requests vorzuschlagen. Weitere Informationen finden Sie im Amazon CodeGuru Reviewer-Benutzerhandbuch.
9. Wählen Sie Erstellen aus.

Nachdem das Repository erstellt wurde, wird es in der Liste Repositories angezeigt. Wählen Sie in der URL-Spalte das Kopiersymbol aus und wählen Sie dann das Protokoll (HTTPS oder SSH) aus, das Sie zum Herstellen der Verbindung mit CodeCommit verwenden möchten. Kopieren Sie die URL.

Wenn Sie beispielsweise Ihrem Repository einen Namen *MyFirstRepo*gegeben haben und HTTPS verwenden, würde die URL wie folgt aussehen:

```
https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyFirstRepo
```

Sie benötigen diese URL später in Schritt 2: Migrieren Sie lokale Inhalte in das CodeCommit Repository.

Schritt 2: Migrieren Sie lokale Inhalte in das CodeCommit Repository

Da Sie nun ein CodeCommit Repository haben, können Sie ein Verzeichnis auf Ihrem lokalen Computer auswählen, das in ein lokales Git-Repository konvertiert werden soll. Der git init-Befehl kann verwendet werden, um entweder bestehende, nicht versionierte Inhalte in ein Git-Repository zu konvertieren oder, wenn Sie noch keine Dateien oder Inhalte haben, um ein neues, leeres Repository zu initialisieren.

1. Ändern Sie im Terminal- oder Befehlszeilenfenster des lokalen Computers Verzeichnisse in das Verzeichnis, das Sie als Quelle für Ihr Repository verwenden möchten.
2. Führen Sie den folgenden Befehl aus, um Git so zu konfigurieren, dass es einen Standardzweig mit dem Namen verwendet**`main`**:

   ```
   git config --local init.defaultBranch main
   ```

   Du kannst diesen Befehl auch ausführen, um deinen Standard-Branch-Namen **`main`** für alle neu erstellten Repositorys auf festzulegen:

   ```
   git config --global init.defaultBranch main
   ```
3. Führen Sie den git init-Befehl aus, um die Git-Versionskontrolle im Verzeichnis zu initialisieren. Dadurch wird ein Unterverzeichnis ".git" im Stamm des Verzeichnisses erstellt, das eine Nachverfolgung zur Versionskontrolle ermöglicht. Der Ordner ".git" enthält außerdem alle für das Repository erforderlichen Metadaten.

   ```
   git init
   ```
4. Überprüfen Sie den Status des initialisierten Verzeichnisses, indem Sie den folgenden Befehl ausführen:

   ```
   git status
   ```

   Fügen Sie die Dateien hinzu, die Sie der Versionskontrolle hinzufügen möchten. In diesem Tutorial führen Sie den `git add`-Befehl mit dem `.`-Spezifizierer aus, um alle Dateien in diesem Verzeichnis hinzuzufügen. Weitere Optionen finden Sie in der Git-Dokumentation.

   ```
   git add .
   ```
5. Erstellen Sie einen Commit für die hinzugefügten Dateien mit einer Commit-Nachricht.

   ```
   git commit -m "Initial commit"
   ```
6. Führen Sie den git push Befehl aus und geben Sie die URL und den Namen des CodeCommit Ziel-Repositorys sowie die `--all` Option an. (Dies ist die URL, die Sie in Schritt 1: Erstellen Sie ein Repository CodeCommit kopiert haben.)

Wenn Sie beispielsweise Ihr Repository benannt haben *MyFirstRepo*und für die Verwendung von HTTPS eingerichtet sind, würden Sie den folgenden Befehl ausführen:

```
git push https://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyFirstRepo --all
```

Schritt 3: Dateien anzeigen in CodeCommit

Nachdem Sie den Inhalt Ihres Verzeichnisses per Push übertragen haben, können Sie die CodeCommit Konsole verwenden, um schnell alle Dateien im Repository anzuzeigen.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie unter Repositories den Namen des Repositorys (z. B. *MyFirstRepository*) aus der Liste aus.
3. Zeigen Sie die Dateien im Repository an, um die Branches, die Klon-URLs, die Einstellungen und vieles mehr zu sehen.

Schritt 4: Teilen Sie das Repository CodeCommit

Wenn Sie ein Repository in erstellen CodeCommit, werden zwei Endpunkte generiert: einer für HTTPS-Verbindungen und einer für SSH-Verbindungen. Beide bieten sichere Verbindungen über ein Netzwerk. Ihre Benutzer können beide Protokolle verwenden. Beide Endpunkte bleiben aktiv, unabhängig davon, welches Protokoll Sie Ihren Benutzern empfehlen. Bevor Sie Ihr Repository mit anderen teilen können, müssen Sie IAM-Richtlinien erstellen, die anderen Benutzern Zugriff auf Ihr Repository gewähren. Stellen Sie Ihren Benutzern diese Zugriffsanweisungen zur Verfügung.

Erstellen einer kundenverwalteten Richtlinie für das Repository

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Dashboard die Optionen Policies und dann Create Policy aus.
3. Wählen Sie auf der Seite „Richtlinie erstellen“ die Option Verwaltete Richtlinie importieren aus.
4. Geben Sie auf der Seite Verwaltete Richtlinien importieren im Feld Richtlinien filtern den Text ein**`AWSCodeCommitPowerUser`**. Klicken Sie auf die Schaltfläche neben dem Richtliniennamen und wählen Sie dann Importieren aus.

5. Wählen Sie auf der Seite Create policy (Richtlinie erstellen) die Option JSON aus. Ersetzen Sie den Teil „*“ in der `Resource` Zeile für CodeCommit Aktionen durch den Amazon-Ressourcennamen (ARN) des CodeCommit Repositorys, wie hier gezeigt:

```
"Resource": [
 "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"
 ]
```

> **Tip**
>
> Um den ARN für das CodeCommit Repository zu finden, gehen Sie zur CodeCommit Konsole, wählen Sie den Repository-Namen aus der Liste aus und wählen Sie dann Einstellungen. Weitere Informationen finden Sie unter Repository-Details anzeigen.

Wenn Sie diese Richtlinie auf mehr als ein Repository anwenden möchten, fügen Sie jedes Repository als Ressource hinzu, indem Sie seinen ARN angeben. Fügen Sie zwischen den Ressourcenanweisungen ein Komma ein, wie hier dargestellt:

```
"Resource": [
 "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
 "arn:aws:codecommit:us-east-2:111111111111:MyOtherDemoRepo"
 ]
```

Wenn Sie mit der Bearbeitung fertig sind, wählen Sie „Richtlinie überprüfen“.

6. Geben Sie auf der Seite „Richtlinie überprüfen“ im Feld Name einen neuen Namen für die Richtlinie ein (z. B. *`AWSCodeCommitPowerUser- MyDemoRepo`*). Geben Sie optional eine Beschreibung für diese Richtlinie ein.
7. Wählen Sie Create Policy (Richtlinie erstellen) aus.

Um den Zugriff auf Ihr Repository zu verwalten, erstellen Sie eine IAM-Gruppe für deren Benutzer, fügen Sie dieser Gruppe IAM-Benutzer hinzu und fügen Sie dann die vom Kunden verwaltete Richtlinie hinzu, die Sie im vorherigen Schritt erstellt haben. Fügen Sie alle anderen für den Zugriff erforderlichen Richtlinien an, z. B. `IAMSelfManageServiceSpecificCredentials` oder `IAMUserSSHKeys`.

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die IAM-Konsole unter https://console.aws.amazon.com/iam/.
2. Wählen Sie im Navigationsbereich Dashboard die Option Groups und dann Create New Group aus.
3. Geben Sie auf der Seite Set Group Name (Name für Gruppe festlegen) im Feld Group Name (Gruppenname) einen Namen für die Gruppe ein (zum Beispiel *MyDemoRepoGroup*) und wählen Sie dann Next Step (Nächster Schritt) aus. Sie sollten den Repository-Namen als Teil des Gruppennamens einfügen.

   > **Note**
   >
   > Dieser Name muss in einem Amazon Web Services Services-Konto eindeutig sein.

4. Aktivieren Sie das Kontrollkästchen neben der kundenverwalteten Richtlinie, die Sie im vorherigen Abschnitt erstellt haben (z. B. AWSCodeCommitPowerUser-MyDemoRepo).
5. Klicken Sie auf der Seite Review auf Create Group. IAM erstellt diese Gruppe mit den angegebenen Richtlinien, die bereits angehängt sind. Die Gruppe erscheint in der Liste der Gruppen, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind.
6. Wählen Sie Ihre Gruppe in der Liste aus.
7. Wählen Sie auf der Gruppenübersichtsseite die Registerkarte Users und anschließend die Option Add Users to Groups aus. Markieren Sie in der Liste, in der alle mit Ihrem Amazon Web Services Services-Konto verknüpften Benutzer angezeigt werden, die Kästchen neben den Benutzern, denen Sie Zugriff auf das CodeCommit Repository gewähren möchten, und wählen Sie dann Benutzer hinzufügen.

   > **Tip**
   >
   > Sie können über das Suchfeld schnell nach Benutzern anhand ihres Namens suchen.

8. Wenn Sie Ihre Benutzer hinzugefügt haben, schließen Sie die IAM-Konsole.

Nachdem Sie einen IAM-Benutzer für den Zugriff CodeCommit mit der von Ihnen konfigurierten Policy-Gruppe und den Richtlinien erstellt haben, senden Sie diesem Benutzer die Informationen, die für die Verbindung mit dem Repository erforderlich sind.

1. Öffnen Sie die CodeCommit Konsole unter https://console.aws.amazon.com/codesuite/codecommit/home.
2. Wählen Sie in der Regionsauswahl den Ort aus, AWS-Region an dem das Repository erstellt wurde. Repositorys sind spezifisch für ein. AWS-Region Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
3. Wählen Sie auf der Seite Repositories (Repositorys) das Repository aus, das Sie freigeben möchten.
4. Wählen Sie unter Clone URL (Klon-URL) das Protokoll aus, das die Benutzer verwenden sollen. Dadurch wird die Klon-URL für das Verbindungsprotokoll kopiert.
5. Übermitteln Sie den Benutzern die Klon-URL und andere erforderliche Anweisungen, wie z. B. zum Installieren der AWS CLI, Konfigurieren eines Profils oder Installieren von Git. Stellen Sie sicher, dass Sie die Konfigurationsinformationen für das Verbindungsprotokoll (z. B. HTTPS) angeben.

Ein Repository inkrementell migrieren

Bei der Migration zu AWS CodeCommit sollten Sie eine inkrementelle oder stufenweise Push-Übertragung des Repositorys erwägen. Auf diese Weise sinkt das Risiko, dass aufgrund einer unterbrochenen oder schlechten Netzwerkverbindung die gesamte Push-Übertragung fehlschlägt. Indem Sie inkrementelle Push-Übertragungen mit einem Skript wie dem hier enthaltenen ausführen, können Sie die Migration erneut starten und nur die Commits, die im vorherigen Versuch nicht erfolgreich übermittelt werden konnten, erneut per Push übertragen.

Die Verfahren in diesem Thema veranschaulichen, wie Sie ein Skript erstellen und ausführen, mit dem eine inkrementelle Migration des Repositorys durchgeführt wird und nur die inkrementellen Schritte, die zuvor fehlgeschlagen sind, erneut ausgeführt werden, bis die Migration vollständig abgeschlossen ist.

Für diese Anweisungen wird davon ausgegangen, dass Sie bereits die Schritte unter Einrichtung und Erstellen eines -Repositorys ausgeführt haben.

Themen

- Schritt 0: Stellen Sie fest, ob eine inkrementelle Migration erfolgen soll
- Schritt 1: Installieren Sie die erforderlichen Komponenten und fügen Sie das CodeCommit Repository als Remote-Repository hinzu
- Schritt 2: Erstellen Sie das Skript, das für die inkrementelle Migration verwendet werden soll

- Schritt 3: Führen Sie das Skript aus und migrieren Sie schrittweise zu CodeCommit
- Anhang: Beispielskript incremental-repo-migration.py

Schritt 0: Stellen Sie fest, ob eine inkrementelle Migration erfolgen soll

Bei der Ermittlung der Repository-Gesamtgröße und der Bestimmung, ob eine inkrementelle Migration sinnvoll ist, sind zahlreiche Faktoren zu berücksichtigen. Am offensichtlichsten ist die Gesamtgröße der Artefakte im Repository. Aber auch andere Faktoren wie z. B. der gesamte Repository-Verlauf tragen zur Größe bei. Ein Repository mit jahrelangem Verlauf und Branches kann sehr umfangreich sein, auch wenn die einzelnen Komponenten nicht groß sind. Es gibt viele Strategien, die eine Migration dieser Repositorys einfacher und effizienter gestalten können. Sie können beispielsweise einen flachen Klon beim Klonen eines Repositorys mit einem umfangreichen Entwicklungsverlauf einsetzen oder die Deltakomprimierung bei großen Binärdateien deaktivieren. Sie können diese Optionen in der Git-Dokumentation prüfen oder Sie können inkrementelle Push-Übertragungen einrichten und konfigurieren, um Ihr Repository mit dem hier enthaltenen Skriptbeispiel `incremental-repo-migration.py` zu migrieren.

Sie sollten die Konfiguration inkrementeller Push-Übertragungen in Betracht ziehen, wenn eine der folgenden Bedingungen zutrifft:

- Das zu migrierende Repository weist einen Verlauf von mehr als fünf Jahren auf.
- Ihre Internetverbindung weist Probleme wie gelegentliche Ausfälle, nicht übertragene Pakete, langsame Geschwindigkeit oder andere Leistungsstörungen auf.
- Die Gesamtgröße des zu migrierenden Repositorys übersteigt 2 GB.
- Das Repository enthält große Artefakte oder Binärdateien, die sich nicht gut komprimieren lassen (z. B. große Bilddateien mit mehr als fünf verfolgten Versionen).
- Sie haben bereits versucht, zu migrieren, CodeCommit und die Meldung „Internal Service Error“ erhalten.

Auch wenn keine der oben genannten Bedingungen zutrifft, können Sie sich für eine inkrementelle Push-Übertragung entscheiden.

Schritt 1: Installieren Sie die erforderlichen Komponenten und fügen Sie das CodeCommit Repository als Remote-Repository hinzu

Sie können ein eigenes Skript mit eigenen Voraussetzungen erstellen. Wenn Sie das Beispiel in diesem Thema verwenden können, müssen Sie Folgendes beachten:

- Installieren Sie die Voraussetzungen.
- Klonen Sie das Repository auf Ihren lokalen Computer.
- Fügen Sie das CodeCommit Repository als Remote-Repository für das Repository hinzu, das Sie migrieren möchten.

Für die Ausführung von incremental-repo-migration .py einrichten

1. Installieren Sie Python 2.6 oder eine neuere Version auf Ihrem lokalen Computer. Weitere Informationen und die neuesten Versionen finden Sie auf der Python-Website.
2. Installieren Sie auf demselben Computer. Dabei handelt es sich um eine Python-Bibliothek GitPython, die für die Interaktion mit Git-Repositorys verwendet wird. Weitere Informationen finden Sie in der GitPython Dokumentation.
3. Verwenden Sie den Befehl git clone --mirror, um das Repository, das auf Ihren lokalen Computer migriert werden soll, zu klonen. Verwenden Sie im Terminal (Linux, macOS oder Unix) oder in der Befehlszeile (Windows) den git clone --mirror Befehl, um ein lokales Repository für das Repository zu erstellen, einschließlich des Verzeichnisses, in dem Sie das lokale Repository erstellen möchten. Um beispielsweise ein Git-Repository *MyMigrationRepo*mit der URL *https://example.com/my-repo/* in ein Verzeichnis namens *my-repo* zu klonen:

   ```
   git clone --mirror https://example.com/my-repo/MyMigrationRepo.git my-repo
   ```

   Die Ausgabe sieht folgendermaßen aus und gibt an, dass das Repository in ein leeres lokales Repository mit der Bezeichnung "my-repo" geklont wurde:

   ```
   Cloning into bare repository 'my-repo'...
   remote: Counting objects: 20, done.
   remote: Compressing objects: 100% (17/17), done.
   remote: Total 20 (delta 5), reused 15 (delta 3)
   Unpacking objects: 100% (20/20), done.
   Checking connectivity... done.
   ```

4. Ändern Sie die Verzeichnisverweise auf das lokale Repository entsprechend in das gerade geklonte Repository (z. B. *my-repo*). Fügen Sie für dieses Verzeichnis mit dem Befehl git remote add ***DefaultRemoteName RemoteRepositoryURL*** das CodeCommit -Repository als Remote-Repository für das lokale Repository hinzu.

   > **Note**
   >
   > Bei der Push-Übertragung von umfangreichen Repositorys sollten Sie anstelle von HTTPS die Verwendung von SSH in Betracht ziehen. Wird eine umfassende Änderung, eine große Anzahl an Änderungen oder ein großes Repository per Push übertragen, werden HTTPS-Langzeitverbindungen häufig aufgrund von Problemen mit der Netzwerkverbindung oder den Firewalleinstellungen vorzeitig beendet. Weitere Informationen zur Einrichtung von SSH CodeCommit finden Sie unter oder. Für SSH-Verbindungen unter Linux, macOS oder Unix Für SSH-Verbindungen unter Windows

   Verwenden Sie beispielsweise den folgenden Befehl, um den SSH-Endpunkt für ein CodeCommit Repository hinzuzufügen, das MyDestinationRepo als Remote-Repository für das Remote-Repository benannt ist: `codecommit`

   ```
   git remote add codecommit ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/
   MyDestinationRepo
   ```

   > **Tip**
   >
   > Da es sich um einen Klon handelt, ist der Standardname für das Remote-Repository (`origin`) bereits in Verwendung. Sie müssen daher einen anderen Namen festlegen. Obwohl im Beispiel `codecommit` verwendet wird, können Sie den Namen beliebig auswählen. Überprüfen Sie mit dem Befehl git remote show die Liste der Remote-Repositorys, die für Ihr lokales Repository eingerichtet sind.

5. Zeigen Sie mit dem Befehl git remote -v die Fetch- und Push-Einstellungen für das lokale Repository an und überprüfen Sie, ob die Einstellungen korrekt sind. Beispiele:

   ```
   codecommit  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDestinationRepo
    (fetch)
   ```

```
codecommit  ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDestinationRepo
 (push)
```

> **Tip**
>
> Falls noch Fetch- und Push-Einträge für ein anderes Remote-Repository angezeigt werden (z. B. Einträge für "origin"), entfernen Sie diese mit dem Befehl git remote set-url --delete.

Schritt 2: Erstellen Sie das Skript, das für die inkrementelle Migration verwendet werden soll

Bei diesen Schritten wird davon ausgegangen, dass Sie das Beispiel-Skript `incremental-repo-migration.py` verwenden.

1. Öffnen Sie einen Texteditor und fügen Sie den Inhalt aus dem Skriptbeispiel in ein leeres Dokument ein.
2. Speichern Sie das Dokument in einem Dokumentverzeichnis (nicht im Arbeitsverzeichnis des lokalen Repositorys) unter dem Namen `incremental-repo-migration.py`. Achten Sie darauf, dass das gewählte Verzeichnis für Ihre lokale Umgebung oder Pfadvariablen konfiguriert ist, damit Sie das Python-Skript über ein Befehlszeilen- oder Teminalfenster ausführen können.

Schritt 3: Führen Sie das Skript aus und migrieren Sie schrittweise zu CodeCommit

Nachdem Sie Ihr `incremental-repo-migration.py` Skript erstellt haben, können Sie es verwenden, um ein lokales Repo inkrementell in ein Repository zu migrieren. CodeCommit Standardmäßig überträgt das Skript Stapel von je 1 000 Commits. Zudem nutzt das Skript die Git-Einstellungen des Verzeichnisses, in dem es ausgeführt wird, als Einstellungen für das lokale und das Remote-Repository. Mit den in `incremental-repo-migration.py` enthaltenen Optionen können Sie bei Bedarf andere Einstellungen konfigurieren.

1. Ändern Sie über das Terminal- oder Befehlszeilenfenster die Verzeichnisverweise auf das zu migrierende lokale Repository.
2. Führen Sie aus dem Verzeichnis den folgenden Befehl aus:

```
python incremental-repo-migration.py
```

3. Das Skript wird ausgeführt, der Fortschritt wird im Terminal- oder Befehlszeilenfenster angezeigt. Bei sehr umfangreichen Repositorys ist nur ein langsamer Fortschritt sichtbar. Falls ein einzelner Push-Versuch dreimal fehlschlägt, wird das Skript angehalten. Sie können das Skript dann ab dem fehlgeschlagenen Stapel erneut ausführen. Das Skript kann so lange ausgeführt werden, bis alle Push-Übertragungen erfolgreich verlaufen sind und die Migration abgeschlossen ist.

Tip

Sie können `incremental-repo-migration.py` aus jedem Verzeichnis ausführen, sofern Sie die zu verwendenden Einstellungen für das lokale und das Remote-Repository mithilfe der Optionen `-l` und `-r` festgelegt haben. Um beispielsweise das Skript von einem beliebigen Verzeichnis aus zu verwenden, um das lokale Repository unter /tmp/*my-repo* in ein externes Repository namens *codecommit* zu migrieren, schreiben Sie:

```
python incremental-repo-migration.py -l "/tmp/my-repo" -r "codecommit"
```

Sie können auch die Option `-b` verwenden, um die Standard-Batch-Größe für den inkrementellen Push zu ändern. Wenn Sie beispielsweise regelmäßig ein Repository mit sehr umfangreichen Binärdateien, an denen häufig Änderungen vorgenommen werden, per Push über eine langsame Netzwerkverbindung übertragen müssen, können Sie die Stapelgröße mithilfe der Option `-b` von 1 000 auf 500 verringern. Beispiele:

```
python incremental-repo-migration.py -b 500
```

Auf diese Weise wird das lokale Repository inkrementell in Stapeln zu je 500 Commits per Push übertragen. Falls Sie die Stapelgröße bei der Repository-Migration erneut ändern möchten (z. B. um die Stapelgröße nach einem fehlgeschlagenen Versuch zu verringern), müssen Sie die Stapel-Tags erst mit der Option `-c` entfernen, bevor Sie die Stapelgröße mit `-b` ändern:

```
python incremental-repo-migration.py -c
python incremental-repo-migration.py -b 250
```

> **Important**
>
> Verwenden Sie nicht die Option `-c`, wenn Sie das Skript nach einem Fehlschlag erneut ausführen möchten. Die Option `-c` löscht die Tags, mit denen die Commits in Stapeln zusammengefasst werden. Nutzen Sie daher die Option `-c` nur, wenn Sie die Stapelgröße ändern und neu starten möchten oder wenn Sie entschieden haben, das Skript nicht weiter zu verwenden.

Anhang: Beispielskript **`incremental-repo-migration.py`**

Wir haben für Sie das Python-Skriptbeispiel `incremental-repo-migration.py` für eine einfache inkrementelle Push-Übertragung von Repositorys entwickelt. Dieses Skript ist ein Open Source-Codebeispiel und wird wie gesehen bereitgestellt.

```
# Copyright 2015 Amazon.com, Inc. or its affiliates. All Rights Reserved. Licensed
 under the Amazon Software License (the "License").
# You may not use this file except in compliance with the License. A copy of the
 License is located at
#    http://aws.amazon.com/asl/
# This file is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
 KIND, express or implied. See the License for
# the specific language governing permissions and limitations under the License.

#!/usr/bin/env python

import os
import sys
from optparse import OptionParser
from git import Repo, TagReference, RemoteProgress, GitCommandError


class PushProgressPrinter(RemoteProgress):
    def update(self, op_code, cur_count, max_count=None, message=""):
        op_id = op_code & self.OP_MASK
        stage_id = op_code & self.STAGE_MASK
        if op_id == self.WRITING and stage_id == self.BEGIN:
            print("\tObjects: %d" % max_count)


class RepositoryMigration:
```

```
    MAX_COMMITS_TOLERANCE_PERCENT = 0.05
    PUSH_RETRY_LIMIT = 3
    MIGRATION_TAG_PREFIX = "codecommit_migration_"

    def migrate_repository_in_parts(
        self, repo_dir, remote_name, commit_batch_size, clean
    ):
        self.next_tag_number = 0
        self.migration_tags = []
        self.walked_commits = set()
        self.local_repo = Repo(repo_dir)
        self.remote_name = remote_name
        self.max_commits_per_push = commit_batch_size
        self.max_commits_tolerance = (
            self.max_commits_per_push * self.MAX_COMMITS_TOLERANCE_PERCENT
        )

        try:
            self.remote_repo = self.local_repo.remote(remote_name)
            self.get_remote_migration_tags()
        except (ValueError, GitCommandError):
            print(
                "Could not contact the remote repository. The most common reasons for
this error are that the name of the remote repository is incorrect, or that you do not
have permissions to interact with that remote repository."
            )
            sys.exit(1)

        if clean:
            self.clean_up(clean_up_remote=True)
            return

        self.clean_up()

        print("Analyzing repository")
        head_commit = self.local_repo.head.commit
        sys.setrecursionlimit(max(sys.getrecursionlimit(), head_commit.count()))

        # tag commits on default branch
        leftover_commits = self.migrate_commit(head_commit)
        self.tag_commits([commit for (commit, commit_count) in leftover_commits])

        # tag commits on each branch
        for branch in self.local_repo.heads:
```

```
            leftover_commits = self.migrate_commit(branch.commit)
            self.tag_commits([commit for (commit, commit_count) in leftover_commits])

        # push the tags
        self.push_migration_tags()

        # push all branch references
        for branch in self.local_repo.heads:
            print("Pushing branch %s" % branch.name)
            self.do_push_with_retries(ref=branch.name)

        # push all tags
        print("Pushing tags")
        self.do_push_with_retries(push_tags=True)

        self.get_remote_migration_tags()
        self.clean_up(clean_up_remote=True)

        print("Migration to CodeCommit was successful")

    def migrate_commit(self, commit):
        if commit in self.walked_commits:
            return []

        pending_ancestor_pushes = []
        commit_count = 1

        if len(commit.parents) > 1:
            # This is a merge commit
            # Ensure that all parents are pushed first
            for parent_commit in commit.parents:
                pending_ancestor_pushes.extend(self.migrate_commit(parent_commit))
        elif len(commit.parents) == 1:
            # Split linear history into individual pushes
            next_ancestor, commits_to_next_ancestor = self.find_next_ancestor_for_push(
                commit.parents[0]
            )
            commit_count += commits_to_next_ancestor
            pending_ancestor_pushes.extend(self.migrate_commit(next_ancestor))

        self.walked_commits.add(commit)

        return self.stage_push(commit, commit_count, pending_ancestor_pushes)
```

```
    def find_next_ancestor_for_push(self, commit):
        commit_count = 0

        # Traverse linear history until we reach our commit limit, a merge commit, or
an initial commit
        while (
            len(commit.parents) == 1
            and commit_count < self.max_commits_per_push
            and commit not in self.walked_commits
        ):
            commit_count += 1
            self.walked_commits.add(commit)
            commit = commit.parents[0]

        return commit, commit_count

    def stage_push(self, commit, commit_count, pending_ancestor_pushes):
        # Determine whether we can roll up pending ancestor pushes into this push
        combined_commit_count = commit_count + sum(
            ancestor_commit_count
            for (ancestor, ancestor_commit_count) in pending_ancestor_pushes
        )

        if combined_commit_count < self.max_commits_per_push:
            # don't push anything, roll up all pending ancestor pushes into this
pending push
            return [(commit, combined_commit_count)]

        if combined_commit_count <= (
            self.max_commits_per_push + self.max_commits_tolerance
        ):
            # roll up everything into this commit and push
            self.tag_commits([commit])
            return []

        if commit_count >= self.max_commits_per_push:
            # need to push each pending ancestor and this commit
            self.tag_commits(
                [
                    ancestor
                    for (ancestor, ancestor_commit_count) in pending_ancestor_pushes
                ]
            )
            self.tag_commits([commit])
```

```
            return []

        # push each pending ancestor, but roll up this commit
        self.tag_commits(
            [ancestor for (ancestor, ancestor_commit_count) in pending_ancestor_pushes]
        )
        return [(commit, commit_count)]

    def tag_commits(self, commits):
        for commit in commits:
            self.next_tag_number += 1
            tag_name = self.MIGRATION_TAG_PREFIX + str(self.next_tag_number)

            if tag_name not in self.remote_migration_tags:
                tag = self.local_repo.create_tag(tag_name, ref=commit)
                self.migration_tags.append(tag)
            elif self.remote_migration_tags[tag_name] != str(commit):
                print(
                    "Migration tags on the remote do not match the local tags. Most
likely your batch size has changed since the last time you ran this script. Please run
this script with the --clean option, and try again."
                )
                sys.exit(1)

    def push_migration_tags(self):
        print("Will attempt to push %d tags" % len(self.migration_tags))
        self.migration_tags.sort(
            key=lambda tag: int(tag.name.replace(self.MIGRATION_TAG_PREFIX, ""))
        )
        for tag in self.migration_tags:
            print(
                "Pushing tag %s (out of %d tags), commit %s"
                % (tag.name, self.next_tag_number, str(tag.commit))
            )
            self.do_push_with_retries(ref=tag.name)

    def do_push_with_retries(self, ref=None, push_tags=False):
        for i in range(0, self.PUSH_RETRY_LIMIT):
            if i == 0:
                progress_printer = PushProgressPrinter()
            else:
                progress_printer = None

            try:
```

```
                if push_tags:
                    infos = self.remote_repo.push(tags=True, progress=progress_printer)
                elif ref is not None:
                    infos = self.remote_repo.push(
                        refspec=ref, progress=progress_printer
                    )
                else:
                    infos = self.remote_repo.push(progress=progress_printer)

                success = True
                if len(infos) == 0:
                    success = False
                else:
                    for info in infos:
                        if (
                            info.flags & info.UP_TO_DATE
                            or info.flags & info.NEW_TAG
                            or info.flags & info.NEW_HEAD
                        ):
                            continue
                        success = False
                        print(info.summary)

                if success:
                    return
            except GitCommandError as err:
                print(err)

        if push_tags:
            print("Pushing all tags failed after %d attempts" %
(self.PUSH_RETRY_LIMIT))
        elif ref is not None:
            print("Pushing %s failed after %d attempts" % (ref, self.PUSH_RETRY_LIMIT))
            print(
                "For more information about the cause of this error, run the following
command from the local repo: 'git push %s %s'"
                % (self.remote_name, ref)
            )
        else:
            print(
                "Pushing all branches failed after %d attempts"
                % (self.PUSH_RETRY_LIMIT)
            )
        sys.exit(1)
```

```
    def get_remote_migration_tags(self):
        remote_tags_output = self.local_repo.git.ls_remote(
            self.remote_name, tags=True
        ).split("\n")
        self.remote_migration_tags = dict(
            (tag.split()[1].replace("refs/tags/", ""), tag.split()[0])
            for tag in remote_tags_output
            if self.MIGRATION_TAG_PREFIX in tag
        )

    def clean_up(self, clean_up_remote=False):
        tags = [
            tag
            for tag in self.local_repo.tags
            if tag.name.startswith(self.MIGRATION_TAG_PREFIX)
        ]

        # delete the local tags
        TagReference.delete(self.local_repo, *tags)

        # delete the remote tags
        if clean_up_remote:
            tags_to_delete = [":" + tag_name for tag_name in
 self.remote_migration_tags]
            self.remote_repo.push(refspec=tags_to_delete)


parser = OptionParser()
parser.add_option(
    "-l",
    "--local",
    action="store",
    dest="localrepo",
    default=os.getcwd(),
    help="The path to the local repo. If this option is not specified, the script will
 attempt to use current directory by default. If it is not a local git repo, the script
 will fail.",
)
parser.add_option(
    "-r",
    "--remote",
    action="store",
    dest="remoterepo",
```

```
    default="codecommit",
    help="The name of the remote repository to be used as the push or migration
 destination. The remote must already be set in the local repo ('git remote add ...').
 If this option is not specified, the script will use 'codecommit' by default.",
)
parser.add_option(
    "-b",
    "--batch",
    action="store",
    dest="batchsize",
    default="1000",
    help="Specifies the commit batch size for pushes. If not explicitly set, the
 default is 1,000 commits.",
)
parser.add_option(
    "-c",
    "--clean",
    action="store_true",
    dest="clean",
    default=False,
    help="Remove the temporary tags created by migration from both the local repo
 and the remote repository. This option will not do any migration work, just cleanup.
 Cleanup is done automatically at the end of a successful migration, but not after a
 failure so that when you re-run the script, the tags from the prior run can be used to
 identify commit batches that were not pushed successfully.",
)

(options, args) = parser.parse_args()

migration = RepositoryMigration()
migration.migrate_repository_in_parts(
    options.localrepo, options.remoterepo, int(options.batchsize), options.clean
)
```

Sicherheit in AWS CodeCommit

Cloud-Sicherheit hat bei AWS höchste Priorität. Als AWS-Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die eingerichtet wurde, um die Anforderungen der anspruchsvollsten Organisationen in puncto Sicherheit zu erfüllen.

Sicherheit ist eine übergreifende Verantwortlichkeit zwischen AWS und Ihnen. Das Modell der geteilten Verantwortung beschreibt dies als Sicherheit der Cloud selbst und Sicherheit in der Cloud:

- Sicherheit der Cloud selbst – AWS ist dafür verantwortlich, die Infrastruktur zu schützen, mit der AWS-Services in der AWS Cloud ausgeführt werden. AWS stellt Ihnen außerdem Services bereit, die Sie sicher nutzen können. Auditoren von Drittanbietern testen und überprüfen die Effektivität unserer Sicherheitsmaßnahmen im Rahmen der AWS-Compliance-Programme regelmäßig. Weitere Informationen zu den Compliance-Programmen, die für geltenAWS CodeCommitfinden Sie unter,AWS-Dienstleistungen in Scope nach Compliance-Programmaus.
- Sicherheit in der Cloud – Ihr Verantwortungsumfang wird durch den AWS-Service bestimmt, den Sie verwenden. Sie sind auch für andere Faktoren verantwortlich, etwa für die Vertraulichkeit Ihrer Daten, für die Anforderungen Ihres Unternehmens und für die geltenden Gesetze und Vorschriften.

In dieser Dokumentation wird erläuterlich, wie das Modell der geteilten Verantwortung bei der Verwendung von CodeCommit zum Tragen kommt. Die folgenden Themen veranschaulichen, wie CodeCommit zur Erfüllung Ihrer Sicherheits- und Compliance-Ziele konfiguriert wird. Sie erfahren außerdem, wie man andere benutztAWS-Services, die Ihnen helfen, Ihre CodeCommit-Ressourcen zu überwachen und zu schützen.

Themen

- Datenschutz in AWS CodeCommit
- Identitäts- und Zugriffsverwaltung für AWS CodeCommit
- Ausfallsicherheit in AWS CodeCommit
- Sicherheit der Infrastruktur in AWS CodeCommit

Datenschutz in AWS CodeCommit

Als verwalteter Service ist durch die globalen Verfahren zur Gewährleistung der Netzwerksicherheit von AWS geschützt. Informationen zu AWS-Sicherheitsservices und wie AWS die Infrastruktur

schützt, finden Sie unter AWS-Cloud-Sicherheit. Informationen zum Entwerfen Ihrer AWS-Umgebung anhand der bewährten Methoden für die Infrastruktursicherheit finden Sie unter Infrastrukturschutz im Security Pillar AWS Well-Architected Framework.

Sie verwenden durch AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf zuzugreifen. Kunden müssen Folgendes unterstützen:

- Transport Layer Security (TLS). Wir benötigen TLS 1.2 und empfehlen TLS 1.3.
- Verschlüsselungs-Suiten mit Perfect Forward Secrecy (PFS) wie DHE (Ephemeral Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Die meisten modernen Systeme wie Java 7 und höher unterstützen diese Modi.

Außerdem müssen Anforderungen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der einem IAM-Prinzipal zugeordnet ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

CodeCommit Repositorys werden im Ruhezustand automatisch verschlüsselt. Es ist kein Eingreifen des Kunden erforderlich. CodeCommit verschlüsselt auch Repository-Daten während der Übertragung. Sie können entweder das HTTPS-Protokoll, das SSH-Protokoll oder beide mit CodeCommit Repositorys verwenden. Weitere Informationen finden Sie unter Einrichten für AWS CodeCommit. Sie können auch den kontoübergreifenden Zugriff auf Repositorys konfigurieren. CodeCommit

Themen

- AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorys
- Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen

AWS Key Management Service und Verschlüsselung für AWS CodeCommit Repositorys

Daten in CodeCommit Repositorys werden während der Übertragung und im Ruhezustand verschlüsselt. Wenn Daten in ein CodeCommit Repository übertragen werden (z. B. durch Aufruf von git push), CodeCommit verschlüsselt die empfangenen Daten, sobald sie im Repository gespeichert werden. Wenn Daten aus einem CodeCommit Repository abgerufen werden (z. B. durch Aufruf von git pull), CodeCommit entschlüsselt die Daten und sendet sie dann an den Aufrufer. Dies setzt voraus, dass der mit der Push- oder Pull-Anforderung verknüpfte IAM-Benutzer von authentifiziert

wurdeAWS. Gesendete oder empfangene Daten werden über die HTTPS- oder SSH-verschlüsselten Netzwerkprotokolle übertragen.

Sie können entweder einen Von AWS verwalteter Schlüssel oder einen vom Kunden verwalteten Schlüssel zum Verschlüsseln und Entschlüsseln der Daten in Ihrem Repository verwenden. Weitere Informationen zu den Unterschieden zwischen vom Kunden verwalteten Schlüsseln und finden Sie unter Vom Kunden Von AWS verwaltete Schlüsselverwaltete Schlüssel und . Von AWS verwaltete Schlüssel Wenn Sie keinen vom Kunden verwalteten Schlüssel angeben, CodeCommit verwendet einen Von AWS verwalteter Schlüssel zum Ver- und Entschlüsseln der Daten in Ihrem Repository. Dies Von AWS verwalteter Schlüssel wird automatisch für Sie in Ihrem erstelltAWS-Konto. Wenn Sie zum ersten Mal ein CodeCommit Repository in einem neuen AWS-Region in Ihrem Amazon-Web-Services-Konto erstellen und keinen vom Kunden verwalteten Schlüssel angeben, CodeCommit erstellt einen Von AWS verwalteter Schlüssel (den `aws/codecommit` Schlüssel) in derselben AWS-Region in AWS Key Management Service (AWS KMS). Dieser `aws/codecommit` Schlüssel wird nur von verwendet CodeCommit. Es wird in Ihrem Amazon Web Services-Konto gespeichert. Je nachdem, was Sie angeben, verwendet CodeCommit entweder den vom Kunden verwalteten Schlüssel oder die , Von AWS verwalteter Schlüssel um die Daten im Repository zu ver- und entschlüsseln.

> **Important**
>
> CodeCommit führt die folgenden AWS KMS Aktionen für den AWS KMS Schlüssel aus, der zum Verschlüsseln und Entschlüsseln von Daten in einem Repository verwendet wird. Wenn Sie ein verwendenVon AWS verwalteter Schlüssel, benötigt ein Benutzer keine expliziten Berechtigungen für diese Aktionen, aber er darf keine angefügten Richtlinien haben, die diese Aktionen für den `aws/codecommit` Schlüssel verweigern. Wenn Sie einen vom Kunden verwalteten Schlüssel verwenden, dessen AWS-Konto ID als Richtlinienprinzipal für diesen Schlüssel festgelegt ist, müssen diese Berechtigungen explizit auf festgelegt werden`allow`. Insbesondere wenn Sie Ihr erstes Repository erstellen und Schlüssel für Ihr Repository aktualisieren, dürfen Sie keine der folgenden Berechtigungen auf gesetzt haben, `deny` wenn Sie einen verwendenVon AWS verwalteter Schlüssel, und muss auf gesetzt sein, `allow` wenn Sie einen vom Kunden verwalteten Schlüssel mit einem Richtlinienprinzipal verwenden:
>
> - `"kms:Encrypt"`
> - `"kms:Decrypt"`
> - `"kms:ReEncrypt"` (Je nach Kontext könnte dies erfordern`kms:ReEncryptFrom`, `kms:ReEncryptTo`, oder `kms:ReEncrypt*` nicht auf Verweigern gesetzt)

- "kms:GenerateDataKey"
- "kms:GenerateDataKeyWithoutPlaintext"
- "kms:DescribeKey"

Wenn Sie Ihren eigenen kundenverwalteten Schlüssel verwenden möchten, muss der Schlüssel in der verfügbar seinAWS-Region, in der sich das Repository befindet. CodeCommit unterstützt die Verwendung von einzel- und multiregionalen kundenverwalteten Schlüsseln. Obwohl alle Ursprungstypen des Schlüsselmaterials unterstützt werden, empfehlen wir die Verwendung der Standard-KMS-Option. Bei Kunden, die die Option Externer Schlüsselspeicher verwenden, kann es zu Verzögerungen beim Anbieter ihres Geschäfts kommen. Darüber hinaus CodeCommit hat die folgenden Anforderungen für vom Kunden verwaltete Schlüssel:

- CodeCommit unterstützt nur die Verwendung von symmetrischen Schlüsseln.
- Der Schlüsselnutzungstyp muss auf Verschlüsseln und Entschlüsseln gesetzt sein.

Weitere Informationen zum Erstellen von kundenverwalteten Schlüsseln finden Sie unter Konzepte und Erstellen von Schlüsseln.

Gehen Sie wie folgt vor CodeCommit, um Informationen über das von Von AWS verwalteter Schlüsselgenerierte anzuzeigen:

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die AWS Key Management Service (AWS KMS)-Konsole unter https://console.aws.amazon.com/kms.
2. Wenn Sie die AWS-Region ändern möchten, verwenden Sie die Regionsauswahl in der oberen rechten Ecke der Seite.
3. Wählen Sie im Service-Navigationsbereich Von AWS verwaltete Schlüssel. Stellen Sie sicher, dass Sie bei der angemeldet sindAWS-Region, in der Sie die Schlüssel überprüfen möchten.
4. Wählen Sie in der Liste der Verschlüsselungsschlüssel die Von AWS verwalteter Schlüssel mit dem Alias aws/codecommit aus. Grundlegende Informationen über die AWS-eigener Schlüssel werden angezeigt.

Sie können diesen nicht ändern oder löschenVon AWS verwalteter Schlüssel.

Wie Verschlüsselungsalgorithmen zum Verschlüsseln von Repository-Daten verwendet werden

CodeCommit verwendet zwei verschiedene Ansätze für die Verschlüsselung von Daten. Einzelne Git-Objekte unter 6 MB werden mit AES-GCM-256 verschlüsselt, was eine Validierung der Datenintegrität ermöglicht. Objekte zwischen 6 MB und den maximalen 2 GB für einen einzelnen Blob werden mit AES-CBC-256 verschlüsselt. validiert CodeCommit immer den Verschlüsselungskontext.

Verschlüsselungskontext

Jeder in AWS KMS integrierte Service gibt einen Verschlüsselungskontext für Ver- und Entschlüsselungsvorgänge an. Der Verschlüsselungskontext enthält zusätzliche authentifizierte Informationen, anhand derer AWS KMS die Datenintegrität überprüft. Wenn dieser für den Verschlüsselungsvorgang angegeben ist, muss er auch für den Entschlüsselungsvorgang angegeben werden. Andernfalls schlägt die Entschlüsselung fehl. CodeCommit verwendet die CodeCommit Repository-ID für den Verschlüsselungskontext. Sie können den get-repository Befehl oder die CodeCommit Konsole verwenden, um die Repository-ID zu finden. Suchen Sie in -AWS CloudTrailProtokollen nach der CodeCommit Repository-ID, um zu verstehen, welche Verschlüsselungsvorgänge für welchen Schlüssel in durchgeführt wurdenAWS KMS, um Daten im CodeCommit Repository zu verschlüsseln oder zu entschlüsseln.

Weitere Informationen über AWS KMS finden Sie im AWS Key Management Service-Entwicklerleitfaden.

Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen

Sie können Benutzern Zugriff auf IhreAWS CodeCommit Repositorys gewähren, ohne IAM-Benutzer für sie zu konfigurieren oder einen Zugriffsschlüssel und einen geheimen Schlüssel zu verwenden. Um Berechtigungen Berechtigungen erstellen Sie eine Rolle und definieren Berechtigungen. Wenn eine Verbundidentität authentifiziert wird, wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center-Benutzerhandbuch. Sie können auch den rollenbasierten Zugriff für IAM-Benutzer konfigurieren, um aufCodeCommit Repositorys in separaten

Amazon Web Services Services-Konten zuzugreifen (eine Technik, die als kontoübergreifender Zugriff bezeichnet wird). Eine schrittweise Anleitung zur Konfiguration des kontoübergreifenden Zugriffs auf ein Repository finden Sie unter Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen.

Sie können den Zugriff für Benutzer konfigurieren, die sich über folgende Methoden authentifizieren möchten oder müssen:

- Security Assertion Markup Language (SAML)
- Multifaktor-Authentifizierung (MFA)
- Verbund
- Login with Amazon
- Amazon Cognito
- Facebook
- Google
- OpenID Connect-kompatibler (OIDC) Identitätsanbieter

> **Note**
>
> Die folgenden Informationen sind nur relevant, wenn die Verbindung mit CodeCommit-Repositorys über git-remote-codecommit oder das AWS CLI-Hilfsprogramm für Anmeldeinformationen hergestellt wird. Da der empfohlene Ansatz für den temporären oder Verbundzugriff auf CodeCommit das Einrichten von git-remote-codecommit ist, enthält dieses Thema Beispiele für dieses Dienstprogramm. Weitere Informationen finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit.
> Sie können keine SSH- oder Git-Anmeldeinformationen und kein HTTPS verwenden, um mit rotierenden oder temporären Anmeldeinformationen eine Verbindung zu CodeCommit-Repositorys herzustellen.

Sie müssen die folgenden Schritte nicht ausführen, wenn alle der folgenden Bedingungen zutreffen:

- Sie sind bei einer Amazon-EC2-Instance angemeldet.
- Sie verwenden Git und HTTPS mit demAWS CLI Credential-Helper, um eine Verbindung von der Amazon EC2 EC2-Instance zuCodeCommit Repositorys herzustellen.

- Die Amazon EC2 EC2-Instance hat ein angehängtes IAM-Instance-Profil, das die inFür HTTPS-Verbindungen unter Linux, MacOS oder Unix mit demAWS CLICredential-Helper oder beschriebenen Zugriffsberechtigungen enthältFür HTTPS-Verbindungen unter Windows mit demAWS CLICredential-Helper.
- Sie haben den Git Credential Helper auf der Amazon EC2 EC2-Instance installiert und konfiguriert, wie inFür HTTPS-Verbindungen unter Linux, MacOS oder Unix mit demAWS CLICredential-Helper oder beschriebenFür HTTPS-Verbindungen unter Windows mit demAWS CLICredential-Helper.

Amazon EC2 EC2-Instances, die die oben genannten Anforderungen erfüllen, sind bereits so eingerichtet, dass sie inCodeCommit Ihrem Namen temporäre Zugangsdaten übermitteln.

> **Note**
>
> Sie könnengit-remote-codecommit Amazon EC2 EC2-Instances konfigurieren und verwenden.

Führen Sie die folgenden Schritte aus, um Benutzern temporär Zugriff auf Ihre CodeCommit-Repositorys zu gewähren.

Schritt 1: Erfüllen der Voraussetzungen

Führen Sie die Einrichtungsschritte aus, um einem Benutzer mit rotierenden Anmeldeinformationen Zugriff auf Ihre CodeCommit-Repositorys zu gewähren:

- Informationen zum kontoübergreifenden Zugriff finden Sie unter Exemplarische Vorgehensweise: Delegieren des Zugriffs über Amazon Web Services Services-Konten mithilfe von IAM-Rollen undKonfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen.
- Weitere Informationen über SAML und einen Verbund finden Sie unter Verwendung des Authentifizierungssystems Ihres Unternehmens für die Erteilung von Zugriff auf AWS-Ressourcen und Informationen über den auf AWS STS SAML 2.0-basierten Verbund.
- Informationen zu MFA finden Sie unter Verwenden von MFA-Geräten (Multi-Factor Authentication) mitAWS und Erstellen temporärer Sicherheitsanmeldeinformationen zur Aktivierung des Zugriffs für IAM-Benutzer.
- Informationen zur Login with Amazon, Amazon Cognito, oder einen beliebigen OIDC-kompatiblen Identitätsanbieter finden Sie unter Abwechseln derAWS STS Web Identity Federation.

Unter Authentifizierung und Zugriffskontrolle für AWS CodeCommit erfahren Sie, mit welchen CodeCommit-Berechtigungen Sie dem Benutzer Zugriff erteilen können.

Schritt 2: Abrufen des Rolle oder Anmeldeinformationen

Wenn Sie möchten, dass Ihre Benutzer auf Repositorys zugreifen, indem sie eine Rolle übernehmen, geben Sie Ihren Benutzern den Amazon-Ressourcennamen (ARN) dieser Rolle an. Andernfalls kann der Benutzer je nach Einrichtung des Zugriffs rotierende Anmeldeinformationen auf eine der folgenden Arten abrufen:

- Rufen Sie für den kontoübergreifenden Zugriff den BefehlAWS CLI assume-role oder dieAWS STS AssumeRoleAPI auf.
- Rufen Sie für SAML denAWS CLI assume-role-with-samlBefehl oder dieAWS STS AssumeRoleWithSAML-API auf.
- Rufen Sie für den Verbund die get-federation-tokenBefehleAWS CLI assume-roleAWS STSAssumeRoleoder die GetFederationTokenOR-APIs auf.
- Rufen Sie für MFA denAWS CLI get-session-tokenBefehl oder dieAWS STS GetSessionTokenAPI auf.
- Rufen Sie für die Anmeldung mit Amazon, Amazon Cognito, Facebook, Google oder einem anderen OIDC-kompatiblen Identitätsanbieter den BefehlAWS CLI assume-role-with-web-identity oder dieAWS STS AssumeRoleWithWebIdentityAPI auf.

Schritt 3: Installierengit-remote-codecommit und konfigurieren Sie denAWS CLI

Sie müssen Ihren lokalen Computer so konfigurieren, dass die Anmeldeinformationen verwendet werden, indem Sie git-remote-codecommit installieren und ein Profil in der AWS CLI konfigurieren.

1. Folgen Sie den Anweisungen in Einrichtung , um die AWS CLI einzurichten. Verwenden Sie den aws configure-Befehl, um ein oder mehrere Profile zu konfigurieren. Erwägen Sie, ein benanntes Profil zu erstellen, das verwendet werden soll, wenn Sie eine Verbindung mit CodeCommit-Repositorys mithilfe rotierender Anmeldeinformationen herstellen.
2. Sie können die Anmeldeinformationen nach einem der folgenden Verfahren mit dem benannten AWS CLI-Profil des Benutzers verknüpfen.

   - Wenn Sie eine Rolle für den Zugriff auf CodeCommit übernehmen, konfigurieren Sie ein benanntes Profil mit den Informationen, die für die Übernahme dieser Rolle erforderlich sind. Wenn Sie beispielsweise eine Rolle annehmen möchten, die `CodeCommitAccess`im Amazon

Web Services Services-Konto 1111111111 benannt ist, können Sie ein Standardprofil für die Arbeit mit anderenAWS Ressourcen und ein benanntes Profil konfigurieren, das verwendet wird, wenn Sie diese Rolle übernehmen. Mit den folgenden Befehlen wird ein benanntes Profil erstellt *CodeAccess*, das eine benannte Rolle annimmt *CodeCommitAccess*. Der Benutzername *Maria_Garcia* ist mit der Sitzung verknüpft und das Standardprofil wird als Quelle derAWS Anmeldeinformationen festgelegt:

```
aws configure set role_arn arn:aws:iam::111111111111:role/CodeCommitAccess --
profile CodeAccess
aws configure set source_profile default --profile CodeAccess
aws configure set role_session_name "Maria_Garcia" --profile CodeAccess
```

Wenn Sie die Änderungen überprüfen möchten, können Sie die `~/.aws/config`-Datei (für Linux) oder die `%UserProfile%.aws\config`-Datei (für Windows) manuell anzeigen oder bearbeiten und die Informationen unter dem benannten Profil überprüfen. Zum Beispiel ähnelt Ihre Datei möglicherweise dem Folgenden:

```
[default]
region = us-east-1
output = json

[profile CodeAccess]
source_profile = default
role_session_name = Maria_Garcia
role_arn = arn:aws:iam::111111111111:role/CodeCommitAccess
```

Nachdem Sie Ihr benanntes Profil konfiguriert haben, können Sie CodeCommit-Repositorys mit dem git-remote-codecommit-Dienstprogramm mithilfe des benannten Profils klonen. Beispielsweise löschen Sie das Repository namens *MyDemoRepo* wie folgt:

```
git clone codecommit://CodeAccess@MyDemoRepo
```

- Wenn Sie Web-Identitätsverbund und OpenID Connect (OIDC) verwenden, konfigurieren Sie ein benanntes Profil, das die AWS Security Token Service (AWS STS) `AssumeRoleWithWebIdentity`-API in Ihrem Namen aufruft, um temporäre Anmeldeinformationen zu aktualisieren. Verwenden Sie den aws configure set-Befehl oder bearbeiten Sie die `~/.aws/credentials`-Datei (für Linux) oder die `%UserProfile%.aws\credentials`-Datei (für Windows) manuell, um ein benanntes AWS CLI-Profil mit den erforderlichen Einstellungswerten hinzuzufügen. Um beispielsweise ein Profil zu erstellen,

das die *CodeCommitAccess*Rolle übernimmt und eine Web-Identitätstoken-Datei ~/ *my-credentials/*verwendetmy-token-file:

```
[CodeCommitWebIdentity]
role_arn = arn:aws:iam::111111111111:role/CodeCommitAccess
web_identity_token_file=~/my-credentials/my-token-file
role_session_name = Maria_Garcia
```

Weitere Informationen finden Sie unter KonfigurationAWS Command Line Interface und Verwendung einer IAM-RolleAWS CLI imAWS Command Line Interface Benutzerhandbuch.

Schritt 4: Zugriff auf dieCodeCommit Repositorys

Nachdem der Benutzer die Anweisungen unter Herstellen einer Verbindung mit einem Repository ausgeführt hat, um eine Verbindung mit den CodeCommit-Repositorys herzustellen, verwendet er die von git-remote-codecommit und Git bereitgestellte erweiterte Funktionalität, um git clone, git push und git pull aufzurufen, um die CodeCommit-Repositorys, auf die er temporären Zugriff hat, zu klonen oder Push- und Pull-Übertragungen durchzuführen. Beispiel zum Klonen eines Repositorys:

```
git clone codecommit://CodeAccess@MyDemoRepo
```

Git-Commit-, Push- und Pull-Befehle verwenden reguläre Git-Syntax.

Wenn der Benutzer in der AWS CLI das benannte AWS CLI-Profil angibt, das den rotierenden Anmeldeinformationen zugewiesen ist, werden die Ergebnisse zurückgegeben, die für dieses Profil gültig sind.

Identitäts- und Zugriffsverwaltung für AWS CodeCommit

AWS Identity and Access Management (IAM) ist ein AWS-Service, mit dem Administratoren den Zugriff auf AWS-Ressourcen sicher steuern können. IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um Ressourcen zu verwenden CodeCommit . IAM ist ein AWS-Service, den Sie ohne zusätzliche Kosten verwenden können.

Themen

- Zielgruppe
- Authentifizierung mit Identitäten
- Verwalten des Zugriffs mit Richtlinien
- Authentifizierung und Zugriffskontrolle für AWS CodeCommit
- Featuresweise von AWS CodeCommit mit IAM
- Ressourcenbasierte CodeCommit-Richtlinien
- Autorisierung auf der Grundlage von Tags CodeCommit
- CodeCommit IAM-Rollen
- AWS CodeCommitBeispiele für identitätsbasierte -Richtlinien
- Fehlerbehebung für AWS CodeCommit-Identität und -Zugriff

Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, unterscheidet sich je nach Ihrer Arbeit in CodeCommit.

Dienstbenutzer — Wenn Sie den CodeCommit Dienst für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die erforderlichen Anmeldeinformationen und Berechtigungen zur Verfügung. Wenn Sie für Ihre Arbeit mehr CodeCommit Funktionen verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Featuresweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anzufordern müssen. Unter Fehlerbehebung für AWS CodeCommit-Identität und -Zugriff finden Sie nützliche Informationen für den Fall, dass Sie keinen Zugriff auf eine Funktion in CodeCommit haben.

Serviceadministrator — Wenn Sie in Ihrem Unternehmen für die CodeCommit Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf CodeCommit. Es ist Ihre Aufgabe, zu bestimmen, auf welche CodeCommit Funktionen und Ressourcen Ihre Servicebenutzer zugreifen sollen. Sie müssen dann Anträge an Ihren IAM-Administrator stellen, um die Berechtigungen Ihrer Servicenutzer zu ändern. Lesen Sie die Informationen auf dieser Seite, um die Grundkonzepte von IAM nachzuvollziehen. Weitere Informationen darüber, wie Ihr Unternehmen IAM nutzen kann CodeCommit, finden Sie unterFeaturesweise von AWS CodeCommit mit IAM.

IAM-Administrator – Wenn Sie als IAM-Administrator fungieren, sollten Sie Einzelheiten dazu kennen, wie Sie Richtlinien zur Verwaltung des Zugriffs auf CodeCommit verfassen können. Beispiele für CodeCommit identitätsbasierte Richtlinien, die Sie in IAM verwenden können, finden Sie unter. AWS CodeCommitBeispiele für identitätsbasierte -Richtlinien

Authentifizierung mit Identitäten

Authentifizierung ist die Art, wie Sie sich mit Ihren Anmeldeinformationen bei AWS anmelden. Die Authentifizierung (Anmeldung bei AWS) muss als Root-Benutzer des AWS-Kontos, als IAM-Benutzer oder durch Übernahme einer IAM-Rolle erfolgen.

Sie können sich bei AWS als Verbundidentität mit Anmeldeinformationen anmelden, die über eine Identitätsquelle bereitgestellt werden. Benutzer von AWS IAM Identity Center. (IAM Identity Center), die Single-Sign-on-Authentifizierung Ihres Unternehmens und Anmeldeinformationen für Google oder Facebook sind Beispiele für Verbundidentitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie auf AWS mithilfe des Verbunds zugreifen, übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich bei der AWS Management Console oder beim AWS-Zugriffsportal anmelden. Weitere Informationen zum Anmelden bei AWS finden Sie unter So melden Sie sich bei Ihrem AWS-Konto an im Benutzerhandbuch von AWS-Anmeldung.

Bei programmgesteuerten Zugriff auf AWS bietet AWS ein Software Development Kit (SDK) und eine Command Line Interface (CLI, Befehlszeilenschnittstelle) zum kryptographischen Signieren Ihrer Anforderungen mit Ihren Anmeldeinformationen. Wenn Sie keine AWS-Tools verwenden, müssen Sie Anforderungen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode zum eigenen Signieren von Anforderungen finden Sie unter Signieren von AWS-API-Anforderungen im IAM-Benutzerhandbuch.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. AWS empfiehlt beispielsweise die Verwendung von Multi-Faktor Authentifizierung (MFA), um die Sicherheit Ihres Kontos zu verbessern. Weitere Informationen finden Sie unter Multi-Faktor-Authentifizierung im AWS IAM Identity Center-Benutzerhandbuch und Verwenden der Multi-Faktor-Authentifizierung (MFA) in AWS im IAM-Benutzerhandbuch.

AWS-Konto-Root-Benutzer

Wenn Sie ein AWS-Konto neu erstellen, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services und Ressourcen des Kontos hat. Diese Identität wird als AWS-Konto-Root-Benutzer bezeichnet. Für den Zugriff auf den Root-Benutzer müssen Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, die zur Erstellung des Kontos verwendet wurden. Wir raten ausdrücklich davon ab, den Root-Benutzer für Alltagsaufgaben zu verwenden. Schützen Sie Ihre Root-Benutzer-Anmeldeinformationen und verwenden Sie diese, um die Aufgaben

auszuführen, die nur der Root-Benutzer ausführen kann. Eine vollständige Liste der Aufgaben, für die Sie sich als Root-Benutzer anmelden müssen, finden Sie unter Aufgaben, die Root-Benutzer-Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

IAM-Benutzer und -Gruppen

Ein IAM-Benutzer ist eine Identität in Ihrem AWS-Konto mit bestimmten Berechtigungen für eine einzelne Person oder eine einzelne Anwendung. Wenn möglich, empfehlen wir, temporäre Anmeldeinformationen zu verwenden, anstatt IAM-Benutzer zu erstellen, die langfristige Anmeldeinformationen wie Passwörter und Zugriffsschlüssel haben. Bei speziellen Anwendungsfällen, die langfristige Anmeldeinformationen mit IAM-Benutzern erfordern, empfehlen wir jedoch, die Zugriffsschlüssel zu rotieren. Weitere Informationen finden Sie unter Regelmäßiges Rotieren von Zugriffsschlüsseln für Anwendungsfälle, die langfristige Anmeldeinformationen erfordern im IAM-Benutzerhandbuch.

Eine IAM-Gruppe ist eine Identität, die eine Sammlung von IAM-Benutzern angibt. Sie können sich nicht als Gruppe anmelden. Mithilfe von Gruppen können Sie Berechtigungen für mehrere Benutzer gleichzeitig angeben. Gruppen vereinfachen die Verwaltung von Berechtigungen, wenn es zahlreiche Benutzer gibt. Sie könnten beispielsweise einer Gruppe mit dem Namen IAMAdmins Berechtigungen zum Verwalten von IAM-Ressourcen erteilen.

Benutzer unterscheiden sich von Rollen. Ein Benutzer ist einer einzigen Person oder Anwendung eindeutig zugeordnet. Eine Rolle kann von allen Personen angenommen werden, die sie benötigen. Benutzer besitzen dauerhafte Anmeldeinformationen. Rollen stellen temporäre Anmeldeinformationen bereit. Weitere Informationen finden Sie unter Erstellen eines IAM-Benutzers (anstatt einer Rolle) im IAM-Benutzerhandbuch.

IAM-Rollen

Eine IAM-Rolle ist eine Identität in Ihrem AWS-Konto mit spezifischen Berechtigungen. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der AWS Management Console übernehmen, indem Sie Rollen wechseln. Sie können eine Rolle annehmen, indem Sie eine AWS CLI oder AWS-API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Verwenden von IAM-Rollen im IAM-Benutzerhandbuch.

IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

- Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert,

so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center-Benutzerhandbuch.

- Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.
- Kontoübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. In einigen AWS-Services können Sie jedoch eine Richtlinie direkt an eine Ressource anfügen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.
- Serviceübergreifender Zugriff – Einige AWS-Services verwenden Features in anderen AWS-Services. Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon EC2 aus oder speichert Objekte in Amazon S3. Ein Dienst kann dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.
  - Forward access sessions (FAS) – Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle zum Ausführen von Aktionen in AWS verwenden, gelten Sie als Prinzipal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen AWS-Service aufruft, in Kombination mit der Anforderung an den AWS-Service, Anforderungen an nachgelagerte Services zu stellen. FAS-Anforderungen werden nur dann gestellt, wenn ein Dienst eine Anforderung erhält, die Interaktionen mit anderen AWS-Services oder Ressourcen erfordert, um abgeschlossen werden zu können. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.
  - Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM

erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.

- Serviceverknüpfte Rolle – Eine serviceverknüpfte Rolle ist ein Typ von Servicerolle, die mit einem AWS-Service verknüpft ist. Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Serviceverknüpfte Rollen werden in Ihrem AWS-Konto angezeigt und gehören zum Service. Ein IAM-Administrator kann die Berechtigungen für serviceverbundene Rollen anzeigen, aber nicht bearbeiten.

- Anwendungen in Amazon EC2 – Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und AWS CLI- oder AWS-API-Anforderungen durchführen. Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Erstellen Sie ein Instance-Profil, das an die Instance angefügt ist, um eine AWS-Rolle einer EC2-Instance zuzuweisen und die Rolle für sämtliche Anwendungen der Instance bereitzustellen. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter Erstellen einer IAM-Rolle (anstatt eines Benutzers) im IAM-Benutzerhandbuch.

Verwalten des Zugriffs mit Richtlinien

Für die Zugriffssteuerung in AWS erstellen Sie Richtlinien und weisen diese den AWS-Identitäten oder -Ressourcen zu. Eine Richtlinie ist ein Objekt in AWS, das, wenn es einer Identität oder Ressource zugeordnet wird, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anforderung stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden in AWS als JSON-Dokumente gespeichert. Weitere Informationen zu Struktur und Inhalten von JSON-Richtliniendokumenten finden Sie unter Übersicht über JSON-Richtlinien im IAM-Benutzerhandbuch.

Administratoren können mithilfe von AWS-JSON-Richtlinien festlegen, wer zum Zugriff auf was berechtigt ist. Das bedeutet, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Standardmäßig haben Benutzer, Gruppen und Rollen keine Berechtigungen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern die Berechtigung erteilen, Aktionen für die Ressourcen auszuführen, die sie benötigen. Der Administrator kann dann die IAM-Richtlinien zu Rollen hinzufügen, und Benutzer können die Rollen annehmen.

IAM-Richtlinien definieren Berechtigungen für eine Aktion unabhängig von der Methode, die Sie zur Ausführung der Aktion verwenden. Angenommen, es gibt eine Richtlinie, die Berechtigungen für die `iam:GetRole`-Aktion erteilt. Ein Benutzer mit dieser Richtlinie kann Benutzerinformationen über die AWS Management Console, die AWS CLI oder die AWS -API abrufen.

Identitätsbasierte Richtlinien

Identitätsbasierte Richtlinien sind JSON-Berechtigungsrichtliniendokumente, die Sie einer Identität anfügen können, wie z. B. IAM-Benutzern, -Benutzergruppen oder -Rollen. Diese Richtlinien steuern, welche Aktionen die Benutzer und Rollen für welche Ressourcen und unter welchen Bedingungen ausführen können. Informationen zum Erstellen identitätsbasierter Richtlinien finden Sie unter Erstellen von IAM-Richtlinien im IAM-Benutzerhandbuch.

Identitätsbasierte Richtlinien können weiter als Inline-Richtlinien oder verwaltete Richtlinien kategorisiert werden. Inline-Richtlinien sind direkt in einen einzelnen Benutzer, eine einzelne Gruppe oder eine einzelne Rolle eingebettet. Verwaltete Richtlinien sind eigenständige Richtlinien, die Sie mehreren Benutzern, Gruppen und Rollen in Ihrem AWS-Konto anfügen können. Verwaltete Richtlinien umfassen von AWS verwaltete und von Kunden verwaltete Richtlinien. Informationen dazu, wie Sie zwischen einer verwalteten Richtlinie und einer eingebundenen Richtlinie wählen, finden Sie unter Auswahl zwischen verwalteten und eingebundenen Richtlinien im IAM-Benutzerhandbuch.

Ressourcenbasierte Richtlinien

Ressourcenbasierte Richtlinien sind JSON-Richtliniendokumente, die Sie an eine Ressource anfügen. Beispiele für ressourcenbasierte Richtlinien sind IAM-Rollen-Vertrauensrichtlinien und Amazon-S3-Bucket-Richtlinien. In Services, die ressourcenbasierte Richtlinien unterstützen, können Service-Administratoren sie verwenden, um den Zugriff auf eine bestimmte Ressource zu steuern. Für die Ressource, an welche die Richtlinie angehängt ist, legt die Richtlinie fest, welche Aktionen ein bestimmter Prinzipal unter welchen Bedingungen für diese Ressource ausführen kann. Sie müssen in einer ressourcenbasierten Richtlinie einen Prinzipal angeben. Prinzipale können Konten, Benutzer, Rollen, Verbundbenutzer oder AWS-Services umfassen.

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können verwaltete AWS-Richtlinien von IAM nicht in einer ressourcenbasierten Richtlinie verwenden.

Zugriffssteuerungslisten (ACLs)

Zugriffssteuerungslisten (ACLs) steuern, welche Prinzipale (Kontomitglieder, Benutzer oder Rollen) auf eine Ressource zugreifen können. ACLs sind ähnlich wie ressourcenbasierte Richtlinien, verwenden jedoch nicht das JSON-Richtliniendokumentformat.

Amazon S3, AWS WAF und Amazon VPC sind Beispiele für Dienste, die ACLs unterstützen. Weitere Informationen“ zu ACLs finden Sie unter Zugriffskontrollliste (ACL) – Übersicht (Access Control List) im Amazon-Simple-Storage-Service-Entwicklerhandbuch.

Weitere Richtlinientypen

AWS unterstützt zusätzliche, weniger häufig verwendete Richtlinientypen. Diese Richtlinientypen können die maximalen Berechtigungen festlegen, die Ihnen von den häufiger verwendeten Richtlinientypen erteilt werden können.

- Berechtigungsgrenzen – Eine Berechtigungsgrenze ist ein erweitertes Feature, mit der Sie die maximalen Berechtigungen festlegen können, die eine identitätsbasierte Richtlinie einer IAM-Entität (IAM-Benutzer oder -Rolle) erteilen kann. Sie können eine Berechtigungsgrenze für eine Entität festlegen. Die daraus resultierenden Berechtigungen sind der Schnittpunkt der identitätsbasierten Richtlinien einer Entität und ihrer Berechtigungsgrenzen. Ressourcenbasierte Richtlinien, die den Benutzer oder die Rolle im Feld `Principal` angeben, werden nicht durch Berechtigungsgrenzen eingeschränkt. Ein ausdrückliches Ablehnen in einer dieser Richtlinien setzt das Zulassen außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter Berechtigungsgrenzen für IAM-Entitäten im IAM-Benutzerhandbuch.
- Service-Kontrollrichtlinien (SCPs) – SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OE) in AWS Organizations angeben. AWS Organizations ist ein Dienst für die Gruppierung und zentrale Verwaltung mehrerer AWS-Konten Ihres Unternehmens. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. SCPs schränken Berechtigungen für Entitäten in Mitgliedskonten einschließlich des jeweiligen Root-Benutzer des AWS-Kontos ein. Weitere Informationen zu Organisationen und SCPs finden Sie unter Funktionsweise von SCPs im AWS Organizations-Benutzerhandbuch.
- Sitzungsrichtlinien – Sitzungsrichtlinien sind erweiterte Richtlinien, die Sie als Parameter übergeben, wenn Sie eine temporäre Sitzung für eine Rolle oder einen verbundenen

Benutzer programmgesteuert erstellen. Die resultierenden Sitzungsberechtigungen sind eine Schnittmenge der auf der Identität des Benutzers oder der Rolle basierenden Richtlinien und der Sitzungsrichtlinien. Berechtigungen können auch aus einer ressourcenbasierten Richtlinie stammen. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen finden Sie unter Sitzungsrichtlinien im IAM-Benutzerhandbuch.

Mehrere Richtlinientypen

Wenn mehrere auf eine Anforderung mehrere Richtlinientypen angewendet werden können, sind die entsprechenden Berechtigungen komplizierter. Informationen dazu, wie AWS die Zulässigkeit einer Anforderung ermittelt, wenn mehrere Richtlinientypen beteiligt sind, finden Sie unter Logik für die Richtlinienauswertung im IAM-Benutzerhandbuch.

Authentifizierung und Zugriffskontrolle für AWS CodeCommit

Für den Zugriff auf AWS CodeCommit sind Anmeldeinformationen erforderlich. Diese Anmeldeinformationen müssen über Zugriffsberechtigungen für AWS Ressourcen wie CodeCommit Repositorys und Ihren IAM-Benutzer verfügen, mit dem Sie Ihre Git-Anmeldeinformationen oder den öffentlichen SSH-Schlüssel verwalten, den Sie für die Herstellung von Git-Verbindungen verwenden. In den folgenden Abschnitten erfahren Sie, wie Sie AWS Identity and Access Management(IAM) verwenden und CodeCommit wie Sie den Zugriff auf Ihre Ressourcen sichern können:

- Authentifizierung
- Zugriffskontrolle

Authentifizierung

Da CodeCommit Repositorys Git-basiert sind und die grundlegenden Funktionen von Git unterstützen, einschließlich Git-Anmeldeinformationen, empfehlen wir, dass Sie bei der Arbeit mit einem IAM-Benutzer arbeiten. CodeCommit Du kannst CodeCommit mit anderen Identitätstypen darauf zugreifen, aber die anderen Identitätstypen unterliegen Einschränkungen, wie unten beschrieben.

Identitätstypen:

- IAM-Benutzer — Ein IAM-Benutzer ist eine Identität innerhalb Ihres Amazon Web Services Services-Kontos, die über spezifische benutzerdefinierte Berechtigungen verfügt. Ein IAM-

Benutzer kann beispielsweise über Berechtigungen zum Erstellen und Verwalten von Git-Anmeldeinformationen für den Zugriff auf CodeCommit Repositorys verfügen. Dies ist der empfohlene Benutzertyp für die Arbeit mit. CodeCommit Sie können einen IAM-Benutzernamen und ein Passwort für die Anmeldung bei sicheren AWS-Webseiten verwenden. Dazu zählen beispielsweise die AWS Management Console, AWS -Diskussionsforen und das AWS Support Center.

Sie können Git-Anmeldeinformationen generieren oder öffentliche SSH-Schlüssel mit Ihrem IAM-Benutzer verknüpfen, oder Sie können sie installieren und konfigurieren. git-remote-codecommit Dies sind die einfachsten Möglichkeiten, Git so einzurichten, dass es mit deinen CodeCommit Repositorys funktioniert. Mit Git-Anmeldeinformationen generieren Sie einen statischen Benutzernamen und ein statisches Passwort in IAM. Sie können dieselben Anmeldeinformationen für HTTPS-Verbindungen dann auch mit Git und jedem Drittanbieter-Tool verwenden, das die Authentifizierung mit Git-Benutzername und -Passwort unterstützt. Mit SSH-Verbindungen erstellen Sie öffentliche und private Schlüsseldateien auf Ihrem lokalen Computer, die Git CodeCommit verwenden und für die SSH-Authentifizierung verwenden. Sie verknüpfen den öffentlichen Schlüssel mit Ihrem IAM-Benutzer und speichern den privaten Schlüssel auf Ihrem lokalen Computer. git-remote-codecommiterweitert Git selbst und erfordert keine Einrichtung von Git-Anmeldeinformationen für den Benutzer.

Darüber hinaus können Sie Zugriffsschlüssel für jeden Benutzer erstellen. Verwenden Sie Zugriffsschlüssel, wenn Sie über AWSeines der verschiedenen -SDKsAWS oder über die (AWS Command Line Interface)AWS CLI programmgesteuert auf -Services zugreifen. Das SDK und die CLI-Tools verwenden die Zugriffsschlüssel, um Ihre Anfragen verschlüsselt zu signieren. Wenn Sie die AWS-Tools nicht nutzen, müssen Sie die Anfragen selbst signieren. CodeCommit unterstützt Signature Version 4, ein Protokoll zur Authentifizierung eingehender API-Anfragen. Weitere Informationen zur Authentifizierung von Anfragen finden Sie unter Signature Version 4-Signaturprozess im Allgemeine AWS-Referenz.

- Root-Benutzer für das Amazon Web Services Services-Konto — Wenn Sie sich registrierenAWS, geben Sie eine E-Mail-Adresse und ein Passwort an, die mit Ihrem Amazon Web Services Services-Konto verknüpft sind. Dies sind Ihre Root-Anmeldeinformationen. Sie bieten vollständigen Zugriff auf alle Ihre AWS-Ressourcen. Einige CodeCommit Funktionen sind für Benutzer mit Root-Konten nicht verfügbar. Darüber hinaus besteht die einzige Möglichkeit, Git mit Ihrem Root-Konto zu verwenden, darin, entweder git-remote-codecommit zu installieren und zu konfigurieren (empfohlen) oder das Hilfsprogramm für Anmeldeinformationen von AWS zu konfigurieren, das in der AWS CLI enthalten ist. Sie können mit Ihrem Stammkonto-Benutzer keine GIT-Anmeldeinformationen oder öffentliche/private SSH-Schlüsselpaare verwenden. Aus diesen

Gründen empfehlen wir nicht, bei der Interaktion mit Ihrem Root-Konto den Benutzer zu verwenden CodeCommit.

> **Important**
>
> Aus Sicherheitsgründen empfehlen wir, die Root-Anmeldeinformationen nur zum Erstellen eines Administrator-Benutzers zu verwenden. Hierbei handelt es sich um einen IAM-Benutzer mit vollständigen Berechtigungen für Ihr AWS-Konto. Anschließend können Sie mit diesem Administrator-Benutzer andere IAM-Benutzer und -Rollen mit eingeschränkten Berechtigungen erstellen. Weitere Informationen finden Sie unter Bewährte Methoden für IAM und Erstellen eines Administratorbenutzers und einer Gruppe im IAM-Benutzerhandbuch.

- IAM Identity Center und Benutzer in IAM Identity Center — AWS IAM Identity Center erweitert die Funktionen von und bietet einen zentralen OrtAWS Identity and Access Management, an dem die Verwaltung von Benutzern und deren Zugriff auf Cloud-Anwendungen sowie deren Zugriff auf Cloud-Anwendungen AWS-Konten zusammengeführt werden. Obwohl IAM Identity Center für die meisten Benutzer, die damit arbeitenAWS, als bewährte Methode empfohlen wird, bietet es derzeit keine Mechanismen für Git-Anmeldeinformationen oder SSH-Schlüsselpaare. Diese Benutzer können CodeCommit Repositorys so installieren und konfigurieren, git-remote-codecommit dass sie lokal geklont werden, aber nicht alle integrierten Entwicklungsumgebungen (IDEs) unterstützen das Klonen, Pushen oder Ziehen mit. git-remote-codecommit

  Als bewährte Methode empfiehlt es sich, menschliche Benutzer, einschließlich Benutzer, die Administratorzugriff benötigen, aufzufordern, den Verbund mit einem Identitätsanbieter zu verwenden, um auf AWS-Services mit temporären Anmeldeinformationen zuzugreifen.

  Eine Verbundidentität ist ein Benutzer aus dem Benutzerverzeichnis Ihres Unternehmens, ein Web Identity Provider, AWS Directory Service, das Identity-Center-Verzeichnis oder jeder Benutzer, der mit Anmeldeinformationen, die über eine Identitätsquelle bereitgestellt werden, auf AWS-Services zugreift. Wenn Verbundidentitäten auf AWS-Konten zugreifen, übernehmen sie Rollen und die Rollen stellen temporäre Anmeldeinformationen bereit.

  Für die zentrale Zugriffsverwaltung empfehlen wir Ihnen, AWS IAM Identity Center zu verwenden. Sie können Benutzer und Gruppen im IAM Identity Center erstellen oder Sie können eine Verbindung mit einer Gruppe von Benutzern und Gruppen in Ihrer eigenen Identitätsquelle herstellen und synchronisieren, um sie in allen AWS-Konten und Anwendungen zu verwenden.

Informationen zu IAM Identity Center finden Sie unter Was ist IAM Identity Center? im AWS IAM Identity Center-Benutzerhandbuch.

- IAM-Rolle — Wie ein IAM-Benutzer ist auch eine IAM-Rolle eine IAM-Identität, die Sie in Ihrem Konto erstellen können, um bestimmte Berechtigungen zu gewähren.

  Eine IAM-Rolle ist eine Identität in Ihrem AWS-Konto mit spezifischen Berechtigungen. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der AWS Management Console übernehmen, indem Sie Rollen wechseln. Sie können eine Rolle annehmen, indem Sie eine AWS CLI oder AWS-API-Operation aufrufen oder eine benutzerdefinierte URL verwenden. Weitere Informationen zu Methoden für die Verwendung von Rollen finden Sie unter Verwenden von IAM-Rollen im IAM-Benutzerhandbuch.

  IAM-Rollen mit temporären Anmeldeinformationen sind in folgenden Situationen hilfreich:

  - Verbundbenutzerzugriff – Um einer Verbundidentität Berechtigungen zuzuweisen, erstellen Sie eine Rolle und definieren Berechtigungen für die Rolle. Wird eine Verbundidentität authentifiziert, so wird die Identität der Rolle zugeordnet und erhält die von der Rolle definierten Berechtigungen. Informationen zu Rollen für den Verbund finden Sie unter Erstellen von Rollen für externe Identitätsanbieter im IAM-Benutzerhandbuch. Wenn Sie IAM Identity Center verwenden, konfigurieren Sie einen Berechtigungssatz. Wenn Sie steuern möchten, worauf Ihre Identitäten nach der Authentifizierung zugreifen können, korreliert IAM Identity Center den Berechtigungssatz mit einer Rolle in IAM. Informationen zu Berechtigungssätzen finden Sie unter Berechtigungssätze im AWS IAM Identity Center-Benutzerhandbuch.
  - Temporäre IAM-Benutzerberechtigungen – Ein IAM-Benutzer oder eine -Rolle kann eine IAM-Rolle übernehmen, um vorübergehend andere Berechtigungen für eine bestimmte Aufgabe zu erhalten.
  - Kontoübergreifender Zugriff – Sie können eine IAM-Rolle verwenden, um einem vertrauenswürdigen Prinzipal in einem anderen Konto den Zugriff auf Ressourcen in Ihrem Konto zu ermöglichen. Rollen stellen die primäre Möglichkeit dar, um kontoübergreifendem Zugriff zu gewähren. In einigen AWS-Services können Sie jedoch eine Richtlinie direkt an eine Ressource anfügen (anstatt eine Rolle als Proxy zu verwenden). Informationen zu den Unterschieden zwischen Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien im IAM-Benutzerhandbuch.
  - Serviceübergreifender Zugriff – Einige AWS-Services verwenden Features in anderen AWS-Services. Wenn Sie beispielsweise einen Aufruf in einem Service tätigen, führt dieser Service häufig Anwendungen in Amazon EC2 aus oder speichert Objekte in Amazon S3. Ein Dienst kann

dies mit den Berechtigungen des aufrufenden Prinzipals mit einer Servicerolle oder mit einer serviceverknüpften Rolle tun.

- Forward access sessions (FAS) – Wenn Sie einen IAM-Benutzer oder eine IAM-Rolle zum Ausführen von Aktionen in AWS verwenden, gelten Sie als Prinzipal. Bei einigen Services könnte es Aktionen geben, die dann eine andere Aktion in einem anderen Service initiieren. FAS verwendet die Berechtigungen des Prinzipals, der einen AWS-Service aufruft, in Kombination mit der Anforderung an den AWS-Service, Anforderungen an nachgelagerte Services zu stellen. FAS-Anforderungen werden nur dann gestellt, wenn ein Dienst eine Anforderung erhält, die Interaktionen mit anderen AWS-Services oder Ressourcen erfordert, um abgeschlossen werden zu können. In diesem Fall müssen Sie über Berechtigungen zum Ausführen beider Aktionen verfügen. Einzelheiten zu den Richtlinien für FAS-Anfragen finden Sie unter Zugriffssitzungen weiterleiten.
- Servicerolle – Eine Servicerolle ist eine IAM-Rolle, die ein Service übernimmt, um Aktionen in Ihrem Namen auszuführen. Ein IAM-Administrator kann eine Servicerolle innerhalb von IAM erstellen, ändern und löschen. Weitere Informationen finden Sie unter Erstellen einer Rolle zum Delegieren von Berechtigungen an einen AWS-Service im IAM-Benutzerhandbuch.
- Serviceverknüpfte Rolle – Eine serviceverknüpfte Rolle ist ein Typ von Servicerolle, die mit einem AWS-Service verknüpft ist. Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Serviceverknüpfte Rollen werden in Ihrem AWS-Konto angezeigt und gehören zum Service. Ein IAM-Administrator kann die Berechtigungen für serviceverbundene Rollen anzeigen, aber nicht bearbeiten.

- Anwendungen in Amazon EC2 – Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und AWS CLI- oder AWS-API-Anforderungen durchführen. Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Erstellen Sie ein Instance-Profil, das an die Instance angefügt ist, um eine AWS-Rolle einer EC2-Instance zuzuweisen und die Rolle für sämtliche Anwendungen der Instance bereitzustellen. Ein Instance-Profil enthält die Rolle und ermöglicht, dass Programme, die in der EC2-Instance ausgeführt werden, temporäre Anmeldeinformationen erhalten. Weitere Informationen finden Sie unter Verwenden einer IAM-Rolle zum Erteilen von Berechtigungen für Anwendungen, die auf Amazon EC2-Instances ausgeführt werden im IAM-Benutzerhandbuch.

Informationen dazu, wann Sie IAM-Rollen oder IAM-Benutzer verwenden sollten, finden Sie unter Erstellen einer IAM-Rolle (anstatt eines Benutzers) im IAM-Benutzerhandbuch.

> **Note**
>
> Sie können mit verbundenen Benutzern keine GIT-Anmeldeinformationen oder öffentliche/private SSH-Schlüsselpaare verwenden. Darüber hinaus stehen für verbundene Benutzer keine Benutzereinstellungen zur Verfügung. Informationen zum Einrichten von Verbindungen mithilfe des Verbundzugriffs finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit.

Zugriffskontrolle

Sie können über gültige Anmeldeinformationen verfügen, um Ihre Anfragen zu authentifizieren, aber ohne die entsprechenden Berechtigungen können Sie keine Ressourcen erstellen oder darauf zugreifen. CodeCommit Beispielsweise müssen Sie über Berechtigungen zum Anzeigen von Repositorys, zum Senden von Code, zum Erstellen und Verwalten von Git-Anmeldeinformationen usw. verfügen.

In den folgenden Abschnitten wird beschrieben, wie Sie Berechtigungen für CodeCommit verwalten. Wir empfehlen Ihnen, zunächst die Übersicht zu lesen.

- Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre CodeCommit Ressourcen
- Verwendung identitätsbasierter Richtlinien (IAM-Richtlinien) für CodeCommit
- Referenz für CodeCommit-Berechtigungen

Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre CodeCommit Ressourcen

Jede AWS Ressource gehört einem Amazon Web Services Services-Konto. Die Berechtigungen zum Erstellen einer Ressource oder für den Zugriff darauf werden durch Berechtigungsrichtlinien geregelt. Ein Kontoadministrator kann Berechtigungsrichtlinien an IAM-Identitäten (Benutzer, Gruppen und Rollen) anfügen. Einige Services, wie z. B. AWS Lambda, unterstützen auch das Zuordnen von Berechtigungsrichtlinien zu Ressourcen.

> **Note**
>
> Ein Kontoadministrator (oder Administratorbenutzer) ist ein Benutzer mit Administratorrechten. Weitere Informationen finden Sie unter Bewährte Methoden für IAM im IAM-Benutzerhandbuch.

Beim Erteilen von Berechtigungen entscheiden Sie, wer die Berechtigungen erhält, für welche Ressourcen die Berechtigungen gelten und welche Aktionen an diesen Ressourcen gestattet werden sollen.

Themen

- CodeCommit Ressourcen und Abläufe
- Grundlegendes zum Eigentum an Ressourcen
- Verwalten des Zugriffs auf Ressourcen
- Eingrenzung des Ressourcenbereichs CodeCommit
- Angeben der Richtlinienelemente: Ressourcen, Aktionen, Effekte und Prinzipale
- Angeben von Bedingungen in einer Richtlinie

CodeCommit Ressourcen und Abläufe

CodeCommitIn ist die primäre Ressource ein Repository. Jeder dieser Ressourcen ist ein eindeutiger Amazon-Ressourcenname (ARN) zugeordnet. In einer Richtlinie identifizieren Sie die Ressource, für welche die Richtlinie gilt, mithilfe eines Amazon-Ressourcennamens (ARN). Weitere Informationen über ARNs finden Sie unter Amazon-Ressourcennamen (ARN) und AWS-Service-Namespaces im Allgemeine Amazon Web Services-Referenz. CodeCommit unterstützt derzeit keine anderen Ressourcentypen, die als Unterressourcen bezeichnet werden.

In der folgenden Tabelle wird beschrieben, wie Ressourcen angegeben CodeCommit werden.

Ressourcentyp	ARN-Format
Repository	arn:aws:codecommit:*region*:*account-id* :*repository-name*
Alle CodeCommit Repositorys	arn:aws:codecommit:*

Ressourcentyp	ARN-Format
Alle CodeCommit Repositorys, die dem angegebenen Konto im angegebenen Konto gehören AWS-Region	arn:aws:codecommit:*region*:*account-id* :*

> **Note**
>
> Die meisten AWS-Services behandeln einen Doppelpunkt (:) oder einen Schrägstrich (/) in ARNs als genau dieses Zeichen. CodeCommit Erfordert jedoch eine exakte Übereinstimmung der Ressourcenmuster und Regeln. Verwenden Sie also die richtigen ARN-Zeichen zum Erstellen von Ereignismustern, sodass sie mit der ARN-Syntax in der Ressource übereinstimmen.

Sie können beispielsweise folgendermaßen ein bestimmtes Repository (*MyDemoRepo*) mit dem ARN in der Anweisung angeben:

```
"Resource": "arn:aws:codecommit:us-west-2:111111111111:MyDemoRepo"
```

Um alle Repositorys anzugeben, die zu einem bestimmten Konto gehören, verwenden Sie das Platzhalterzeichen (*) wie folgt:

```
"Resource": "arn:aws:codecommit:us-west-2:111111111111:*"
```

Wenn Sie alle Ressourcen angeben möchten oder wenn eine bestimmte API-Aktion keine ARNs unterstützt, verwenden Sie das Platzhalterzeichen (*) wie folgt im `Resource`-Element:

```
"Resource": "*"
```

Sie können auch das Platzhalterzeichen (*) verwenden, um alle Ressourcen anzugeben, die mit einem Teil eines Repository-Namen übereinstimmen. Der folgende ARN gibt beispielsweise jedes CodeCommit Repository an, das mit dem Namen beginnt `MyDemo` und das für das Amazon Web Services Services-Konto registriert ist `111111111111` in `us-east-2`AWS-Region:

```
arn:aws:codecommit:us-east-2:111111111111:MyDemo*
```

Eine Liste der verfügbaren Operationen, die mit den CodeCommit Ressourcen arbeiten, finden Sie unterReferenz für CodeCommit-Berechtigungen.

Grundlegendes zum Eigentum an Ressourcen

Das Amazon Web Services Services-Konto besitzt die Ressourcen, die im Konto erstellt wurden, unabhängig davon, wer sie erstellt hat. Insbesondere ist der Ressourcenbesitzer das Amazon Web Services Services-Konto der Prinzipalentität (d. h. das Root-Konto, ein IAM-Benutzer oder eine IAM-Rolle), das die Anfrage zur Ressourcenerstellung authentifiziert. Die Funktionsweise wird anhand der folgenden Beispiele deutlich:

- Wenn Sie in Ihrem Amazon Web Services Services-Konto einen IAM-Benutzer erstellen und diesem Benutzer Berechtigungen zum Erstellen von CodeCommit Ressourcen gewähren, kann der Benutzer CodeCommit Ressourcen erstellen. Ihr Amazon Web Services Services-Konto, zu dem der Benutzer gehört, besitzt jedoch die CodeCommit Ressourcen.
- Wenn Sie die Root-Kontoanmeldeinformationen Ihres Amazon Web Services Services-Kontos verwenden, um eine Regel zu erstellen, ist Ihr Amazon Web Services Services-Konto der Eigentümer der CodeCommit Ressource.
- Wenn Sie in Ihrem Amazon Web Services Services-Konto eine IAM-Rolle mit Berechtigungen zum Erstellen von CodeCommit Ressourcen erstellen, kann jeder, der die Rolle übernehmen kann, CodeCommit Ressourcen erstellen. Ihr Amazon Web Services Services-Konto, zu dem die Rolle gehört, besitzt die CodeCommit Ressourcen.

Verwalten des Zugriffs auf Ressourcen

Zum Verwalten des Zugriffs auf AWS-Ressourcen verwenden Sie Berechtigungsrichtlinien. Eine Berechtigungsrichtlinie beschreibt, wer Zugriff auf welche Objekte hat. Im folgenden Abschnitt werden die Optionen zum Erstellen von Berechtigungsrichtlinien erläutert.

> **Note**
>
> In diesem Abschnitt wird die Verwendung von IAM im Kontext von CodeCommit beschrieben. Er enthält keine detaillierten Informationen über den IAM-Service. Weitere Informationen zu IAM finden Sie unter Was ist IAM? im IAM-Benutzerhandbuch. Für Informationen über die

Syntax und Beschreibungen von -IAM-Richtlinien lesen Sie die IAM-Richtlinienreferenz im IAM-Benutzerhandbuch.

Berechtigungsrichtlinien, die mit einer IAM-Identität verknüpft sind, werden als identitätsbasierte Richtlinien (IAM-Richtlinien) bezeichnet. An Ressourcen angehängte Berechtigungsrichtlinien werden als ressourcenbasierte Richtlinien bezeichnet. CodeCommitUnterstützt derzeit nur identitätsbasierte Richtlinien (IAM-Richtlinien).

Themen

- Identitätsbasierte Richtlinien (IAM-Richtlinien)
- Ressourcenbasierte Richtlinien

Identitätsbasierte Richtlinien (IAM-Richtlinien)

Um den Zugriff auf AWS Ressourcen zu verwalten, fügen Sie Berechtigungsrichtlinien an IAM-Identitäten an. In verwenden Sie identitätsbasierte Richtlinien CodeCommit, um den Zugriff auf Repositorys zu steuern. Sie können z. B. Folgendes tun:

- Ordnen Sie einem Benutzer oder einer Gruppe in Ihrem Konto eine Berechtigungsrichtlinie zu — Um einem Benutzer Berechtigungen zum Anzeigen von CodeCommit Ressourcen in der CodeCommit Konsole zu gewähren, fügen Sie einem Benutzer oder einer Gruppe, zu der der Benutzer gehört, eine identitätsbasierte Berechtigungsrichtlinie hinzu.
- Einer Rolle eine Berechtigungsrichtlinie zuordnen (um kontoübergreifende Berechtigungen zu gewähren) — Delegierung, z. B. wenn Sie kontenübergreifenden Zugriff gewähren möchten, beinhaltet die Einrichtung einer Vertrauensstellung zwischen dem Konto, dem die Ressource gehört (das vertrauenswürdige Konto), und dem Konto, das die Benutzer enthält, die auf die Ressource zugreifen müssen (dem vertrauenswürdigen Konto). Eine Berechtigungsrichtlinie erteilt dem Benutzer einer Rolle die erforderlichen Berechtigungen zum Ausführen der vorgesehenen Aufgaben auf der Ressource. Eine Vertrauensrichtlinie gibt an, welche vertrauenswürdigen Konten ihren Benutzern Berechtigungen zum Übernehmen der Rolle erteilen dürfen. Weitere Informationen finden Sie unter IAM-Begriffe und Konzepte.

  Um kontoübergreifende Berechtigungen zu gewähren, fügen Sie einer IAM-Rolle eine identitätsbasierte Berechtigungsrichtlinie hinzu. Beispielsweise kann der Administrator in Konto A wie folgt eine Rolle erstellen, um einem anderen Amazon Web Services Services-Konto (z. B. Konto B) oder einem AWS Service kontoübergreifende Berechtigungen zu gewähren:

1. Der Administrator von Konto A erstellt eine IAM-Rolle und fügt ihr eine Berechtigungsrichtlinie an, die Berechtigungen für Ressourcen in Konto A erteilt.
2. Der Administrator von Konto A weist der Rolle eine Vertrauensrichtlinie zu, die Konto B als den Prinzipal identifiziert, der die Rolle übernehmen kann.
3. Der Administrator von Konto B kann nun Berechtigungen zur Übernahme der Rolle an alle Benutzer in Konto B delegieren. Daraufhin können die Benutzer in Konto B auf Ressourcen von Konto A zugreifen oder auch Ressourcen erstellen. Wenn Sie eine AWS-Serviceberechtigung für die Übernahme der Rolle erteilen wollen, kann der Prinzipal in der Vertrauensrichtlinie auch ein AWS-Service-Prinzipal sein. Weitere Informationen finden Sie unter Delegierung in den Nutzungsbedingungen und Konzepten von IAM.

Weitere Informationen zum Delegieren von Berechtigungen mithilfe von IAM finden Sie unter Zugriffsverwaltung im IAM-Benutzerhandbuch.

Die folgende Beispielrichtlinie ermöglicht einem Benutzer das Erstellen eines Branch in einem Repository mit dem Namen *MyDemoRepo*:

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codecommit:CreateBranch"
      ],
      "Resource" : "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"
    }
  ]
}
```

Um die Aufrufe und Ressourcen einzuschränken, auf die Benutzer in Ihrem Konto Zugriff haben, erstellen Sie spezifische IAM-Richtlinien und fügen Sie diese Richtlinien dann IAM-Benutzern hinzu. Weitere Informationen zum Erstellen von IAM-Rollen und Beispiele für IAM-Richtlinienerklärungen finden Sie unter. CodeCommit Beispiele für vom Kunden verwaltete Identitätsrichtlinien

Ressourcenbasierte Richtlinien

Einige Dienste, wie Amazon S3, unterstützen auch ressourcenbasierte Berechtigungsrichtlinien. Sie können beispielsweise eine ressourcenbasierte Richtlinie an einen S3-Bucket anhängen, um die Zugriffsberechtigungen für diesen Bucket zu verwalten. CodeCommit unterstützt keine ressourcenbasierten Richtlinien, aber Sie können Tags verwenden, um Ressourcen zu identifizieren, die Sie dann in IAM-Richtlinien verwenden können. Ein Beispiel für eine Tag-basierte Richtlinien finden Sie unter Identitätsbasierte Richtlinien (IAM-Richtlinien).

Eingrenzung des Ressourcenbereichs CodeCommit

In CodeCommit können Sie identitätsbasierte Richtlinien und Berechtigungen auf Ressourcen beschränken, wie unter beschrieben. CodeCommit Ressourcen und Abläufe Sie können jedoch nicht die `ListRepositories`- Berechtigung auf eine Ressource anwenden. Stattdessen müssen Sie sie auf alle Ressourcen anwenden (mit dem Platzhalter *). Andernfalls schlägt die Aktion fehl.

Alle anderen CodeCommit Berechtigungen können auf Ressourcen beschränkt werden.

Angeben der Richtlinienelemente: Ressourcen, Aktionen, Effekte und Prinzipale

Sie können Richtlinien erstellen, um Benutzern den Zugriff auf Ressourcen zu gewähren oder zu verweigern oder Benutzern zu erlauben oder zu verweigern, bestimmte Aktionen mit diesen Ressourcen durchzuführen. CodeCommit definiert eine Reihe von öffentlichen API-Vorgängen, die definieren, wie Benutzer mit dem Dienst arbeiten, sei es über die CodeCommit Konsole, die SDKsAWS CLI, die oder durch den direkten Aufruf dieser APIs. CodeCommit Definiert eine Reihe von Aktionen, die Sie in einer Richtlinie angeben können, um Berechtigungen für diese API-Operationen zu gewähren.

Für einige API-Operationen können Berechtigungen für mehrere Aktionen erforderlich sein. Weitere Informationen zu Ressourcen und API-Operationen finden Sie unter CodeCommit Ressourcen und Abläufe und Referenz für CodeCommit-Berechtigungen.

Die Grundelemente einer Richtlinie sind:

- Ressource — Um die Ressource zu identifizieren, für die die Richtlinie gilt, verwenden Sie einen Amazon-Ressourcennamen (ARN). Weitere Informationen finden Sie unter CodeCommit Ressourcen und Abläufe.
- Aktion — Um Ressourcenoperationen zu identifizieren, die Sie zulassen oder verweigern möchten, verwenden Sie Aktionsschlüsselwörter. Je nach Angabe `Effect` erlaubt oder verweigert die `codecommit:GetBranch` Berechtigung dem Benutzer beispielsweise die Ausführung des

`GetBranch` Vorgangs, mit dem Details zu einem Branch in einem CodeCommit Repository abgerufen werden.

- Wirkung — Sie geben den Effekt an, der eintritt, wenn der Benutzer die bestimmte Aktion anfordert, entweder zulassen oder verweigern. Wenn Sie den Zugriff auf eine Ressource nicht ausdrücklich gestatten („Allow“), wird er automatisch verweigert. Sie können den Zugriff auf eine Ressource auch explizit verweigern. So können Sie sicherstellen, dass Benutzer nicht darauf zugreifen können, auch wenn der Zugriff durch eine andere Richtlinie gestattet wird.
- Prinzipal — In identitätsbasierten Richtlinien (IAM-Richtlinien) CodeCommit unterstützt der einzige Richtlinientyp den Benutzer, dem die Richtlinie zugeordnet ist, der implizite Prinzipal.

Weitere Informationen zur IAM-Richtliniensyntax finden Sie unter IAM-Richtlinienreferenz im IAM-Benutzerhandbuch.

Eine Tabelle mit allen CodeCommit API-Aktionen und den Ressourcen, für die sie gelten, finden Sie unter. Referenz für CodeCommit-Berechtigungen

Angeben von Bedingungen in einer Richtlinie

Wenn Sie Berechtigungen gewähren, verwenden Sie die Sprache der Zugriffsrichtlinie für IAM, um die Bedingungen anzugeben, unter denen eine Richtlinie wirksam werden soll. Beispielsweise kann festgelegt werden, dass eine Richtlinie erst ab einem bestimmten Datum gilt. Weitere Informationen zur Angabe von Bedingungen in einer Richtliniensprache finden Sie unter Bedingungen und Richtliniengrammatik im IAM-Benutzerhandbuch.

Bedingungen werden mithilfe vordefinierter Bedingungsschlüssel formuliert. Für CodeCommit gibt es keine speziellen Bedingungsschlüssel. Stattdessen können Sie nach Bedarf die AWS-weiten Bedingungsschlüssel verwenden. Sie finden eine vollständige Liste der AWS-weiten Schlüssel unter Verfügbare Schlüssel für Bedingungen im IAM-Benutzerhandbuch enthalten.

Verwendung identitätsbasierter Richtlinien (IAM-Richtlinien) für CodeCommit

Die folgenden Beispiele für identitätsbasierte Richtlinien zeigen, wie ein Kontoadministrator Berechtigungsrichtlinien an IAM-Identitäten (Benutzer, Gruppen und Rollen) anhängen kann, um Berechtigungen zur Ausführung von Vorgängen mit Ressourcen zu erteilen. CodeCommit

> **Important**
>
> Wir empfehlen Ihnen, zunächst die einführenden Themen zu lesen, in denen die grundlegenden Konzepte und Optionen für die Verwaltung des Zugriffs auf Ihre Ressourcen

erläutert werden. CodeCommit Weitere Informationen finden Sie unter Überblick über die Verwaltung von Zugriffsberechtigungen für Ihre CodeCommit Ressourcen.

Themen

- Erforderliche Berechtigungen für die Verwendung der CodeCommit-Konsole
- Anzeigen von Ressourcen in der Konsole
- Von AWS verwaltete Richtlinien für CodeCommit
- Beispiele für vom Kunden verwaltete Richtlinien

Nachstehend finden Sie ein Beispiel für eine identitätsbasierte Berechtigungsrichtlinie:

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codecommit:BatchGetRepositories"
      ],
      "Resource" : [
        "arn:aws:codecommit:us-east-2:111111111111:MyDestinationRepo",
        "arn:aws:codecommit:us-east-2:111111111111:MyDemo*"
      ]
    }
  ]
}
```

Diese Richtlinie enthält eine Erklärung, die es einem Benutzer ermöglicht, Informationen über das angegebene CodeCommit Repository `MyDestinationRepo` und alle CodeCommit Repositorys, die mit dem Namen beginnen, `MyDemo` in der **`us-east-2`** Region abzurufen.

Erforderliche Berechtigungen für die Verwendung der CodeCommit-Konsole

Informationen zu den erforderlichen Berechtigungen für jeden CodeCommit API-Vorgang sowie weitere Informationen zu CodeCommit Vorgängen finden Sie unterReferenz für CodeCommit-Berechtigungen.

Damit Benutzer die CodeCommit Konsole verwenden können, muss ihnen der Administrator Berechtigungen für CodeCommit Aktionen gewähren. Sie könnten beispielsweise die AWSCodeCommitPowerUserverwaltete Richtlinie oder eine entsprechende Richtlinie einem Benutzer oder einer Gruppe zuordnen.

Zusätzlich zu den Berechtigungen, die Benutzern durch identitätsbasierte Richtlinien erteilt werden, benötigt CodeCommit Berechtigungen für AWS Key Management Service (AWS KMS)-Aktionen. Ein IAM-Benutzer benötigt für diese Aktionen keine ausdrücklichen `Allow` Berechtigungen, dem Benutzer dürfen jedoch keine Richtlinien angehängt sein, die die folgenden Berechtigungen festlegen auf`Deny`:

```
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyWithoutPlaintext",
        "kms:DescribeKey"
```

Weitere Informationen zur Verschlüsselung und finden Sie CodeCommit unterAWS KMS und Verschlüsselung.

Anzeigen von Ressourcen in der Konsole

Die CodeCommit Konsole benötigt die `ListRepositories` Erlaubnis, eine Liste der Repositorys für Ihr Amazon Web Services Services-Konto in dem Land anzuzeigen, in AWS-Region dem Sie angemeldet sind. Die Konsole umfasst auch eine Funktion Go to resource (Zu Ressource wechseln), um schnell ohne Berücksichtigung der Groß- und Kleinschreibung nach Ressourcen zu suchen. Diese Suche wird in Ihrem Amazon Web Services Services-Konto durchgeführt, in AWS-Region dem Sie angemeldet sind. Die folgenden Ressourcen werden für die folgenden Services angezeigt:

- AWS CodeBuild: Build-Projekte
- AWS CodeCommit: Repositorys
- AWS CodeDeploy: Anwendungen
- AWS CodePipeline: Pipelines

Für diese Suche unter den Ressourcen in allen Services müssen Sie über die folgenden Berechtigungen verfügen:

- CodeBuild: `ListProjects`

- CodeCommit: `ListRepositories`
- CodeDeploy: `ListApplications`
- CodePipeline: `ListPipelines`

Es werden keine Ergebnisse für die Ressourcen eines Service zurückgegeben, wenn Sie nicht über Berechtigungen für diesen Service verfügen. Auch wenn Sie über Berechtigungen zum Anzeigen von Ressourcen verfügen, werden bestimmte Ressourcen nicht zurückgegeben, wenn die Anzeige dieser Ressourcen ausdrücklich verweigert wird (`Deny`).

Von AWS verwaltete Richtlinien für CodeCommit

Um Benutzern, Gruppen und Rollen Berechtigungen hinzuzufügen, ist es einfacher, von AWS verwaltete Richtlinien zu verwenden, als selbst Richtlinien zu schreiben. Es erfordert Zeit und Fachwissen, um von Kunden verwaltete IAM-Richtlinien zu erstellen, die Ihrem Team nur die benötigten Berechtigungen bieten. Um schnell loszulegen, können Sie unsere von AWS verwalteten Richtlinien verwenden. Diese Richtlinien decken allgemeine Anwendungsfälle ab und sind in Ihrem AWS-Konto verfügbar. Weitere Informationen zu verwalteten AWS-Richtlinien finden Sie unter Verwaltete AWS-Richtlinien im IAM-Leitfaden.

AWS-Services pflegen und Aktualisieren von verwalteten AWS-Richtlinien. Die Berechtigungen in von AWS verwalteten Richtlinien können nicht geändert werden. Services fügen einer von AWS verwalteten Richtlinien gelegentlich zusätzliche Berechtigungen hinzu, um neue Features zu unterstützen. Diese Art von Update betrifft alle Identitäten (Benutzer, Gruppen und Rollen), an welche die Richtlinie angehängt ist. Services aktualisieren eine von AWS verwaltete Richtlinie am ehesten, ein neues Feature gestartet wird oder neue Vorgänge verfügbar werden. Services entfernen keine Berechtigungen aus einer von AWS verwalteten Richtlinie, so dass Richtlinien-Aktualisierungen Ihre vorhandenen Berechtigungen nicht beeinträchtigen.

Darüber hinaus unterstützt AWS verwaltete Richtlinien für Auftragsfunktionen, die mehrere Services umfassen. Die ReadOnlyAccess AWS verwaltete Richtlinie bietet beispielsweise schreibgeschützten Zugriff auf alle AWS Services und -Ressourcen. Wenn ein Service ein neues Feature startet, fügt AWS schreibgeschützte Berechtigungen für neue Vorgänge und Ressourcen hinzu. Eine Liste und Beschreibungen der Richtlinien für Auftragsfunktionen finden Sie in Verwaltete AWS-Richtlinien für Auftragsfunktionen im IAM-Leitfaden.

AWS Durch die Bereitstellung von eigenständigen IAM-Richtlinien, die von erstellt und verwaltet werden, deckt viele häufige Anwendungsfälle ab AWS. Diese von AWS verwalteten Richtlinien erteilen die erforderlichen Berechtigungen für häufige Anwendungsfälle. Die verwalteten Richtlinien

für gewähren CodeCommit auch Berechtigungen zur Ausführung von Vorgängen in anderen Diensten wie IAM, Amazon SNS und Amazon CloudWatch Events, je nach den Zuständigkeiten der Benutzer, denen die betreffende Richtlinie erteilt wurde. Bei der AWSCodeCommitFullAccess Richtlinie handelt es sich beispielsweise um eine Benutzerrichtlinie auf Administrationsebene, die es Benutzern mit dieser Richtlinie ermöglicht, CloudWatch Ereignisregeln für Repositorys (Regeln, deren Namen ein Präfix haben`codecommit`) und Amazon SNS SNS-Themen für Benachrichtigungen über repository-bezogene Ereignisse (Themen, deren Namen ein Präfix haben) zu erstellen und zu verwalten sowie Repositorys in zu verwalten. `codecommit` CodeCommit

Die folgenden AWS-verwalteten Richtlinien, die Sie Benutzern in Ihrem Konto anfügen können, gelten speziell für CodeCommit.

Themen

- AWSverwaltete Richtlinie: AWSCodeCommitFullAccess
- AWSverwaltete Richtlinie: AWSCodeCommitPowerUser
- AWSverwaltete Richtlinie: AWSCodeCommitReadOnly
- CodeCommit verwaltete Richtlinien und Benachrichtigungen
- AWS CodeCommitverwaltete Richtlinien und Amazon CodeGuru Reviewer
- CodeCommit Aktualisierungen der AWS verwalteten Richtlinien

AWSverwaltete Richtlinie: AWSCodeCommitFullAccess

Sie können die `AWSCodeCommitFullAccess`-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie gewährt vollen Zugriff auf CodeCommit. Wenden Sie diese Richtlinie nur auf Benutzer auf Administratorebene an, denen Sie die volle Kontrolle über CodeCommit Repositorys und zugehörige Ressourcen in Ihrem Amazon Web Services Services-Konto gewähren möchten, einschließlich der Möglichkeit, Repositorys zu löschen.

Die AWSCodeCommitFullAccess Richtlinie enthält die folgende Grundsatzerklärung:

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

```
      "Resource": "*"
    },
    {
      "Sid": "CloudWatchEventsCodeCommitRulesAccess",
      "Effect": "Allow",
      "Action": [
        "events:DeleteRule",
        "events:DescribeRule",
        "events:DisableRule",
        "events:EnableRule",
        "events:PutRule",
        "events:PutTargets",
        "events:RemoveTargets",
        "events:ListTargetsByRule"
      ],
      "Resource": "arn:aws:events:*:*:rule/codecommit*"
    },
    {
      "Sid": "SNSTopicAndSubscriptionAccess",
      "Effect": "Allow",
      "Action": [
        "sns:CreateTopic",
        "sns:DeleteTopic",
        "sns:Subscribe",
        "sns:Unsubscribe",
        "sns:SetTopicAttributes"
      ],
      "Resource": "arn:aws:sns:*:*:codecommit*"
    },
    {
      "Sid": "SNSTopicAndSubscriptionReadAccess",
      "Effect": "Allow",
      "Action": [
        "sns:ListTopics",
        "sns:ListSubscriptionsByTopic",
        "sns:GetTopicAttributes"
      ],
      "Resource": "*"
    },
    {
      "Sid": "LambdaReadOnlyListAccess",
      "Effect": "Allow",
      "Action": [
        "lambda:ListFunctions"
```

```
        ],
        "Resource": "*"
      },
      {
        "Sid": "IAMReadOnlyListAccess",
        "Effect": "Allow",
        "Action": [
          "iam:ListUsers"
        ],
        "Resource": "*"
      },
      {
        "Sid": "IAMReadOnlyConsoleAccess",
        "Effect": "Allow",
        "Action": [
          "iam:ListAccessKeys",
          "iam:ListSSHPublicKeys",
          "iam:ListServiceSpecificCredentials"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
      },
      {
        "Sid": "IAMUserSSHKeys",
        "Effect": "Allow",
        "Action": [
          "iam:DeleteSSHPublicKey",
          "iam:GetSSHPublicKey",
          "iam:ListSSHPublicKeys",
          "iam:UpdateSSHPublicKey",
          "iam:UploadSSHPublicKey"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
      },
      {
        "Sid": "IAMSelfManageServiceSpecificCredentials",
        "Effect": "Allow",
        "Action": [
          "iam:CreateServiceSpecificCredential",
          "iam:UpdateServiceSpecificCredential",
          "iam:DeleteServiceSpecificCredential",
          "iam:ResetServiceSpecificCredential"
        ],
        "Resource": "arn:aws:iam::*:user/${aws:username}"
      },
```

```
        {
          "Sid": "CodeStarNotificationsReadWriteAccess",
          "Effect": "Allow",
          "Action": [
            "codestar-notifications:CreateNotificationRule",
            "codestar-notifications:DescribeNotificationRule",
            "codestar-notifications:UpdateNotificationRule",
            "codestar-notifications:DeleteNotificationRule",
            "codestar-notifications:Subscribe",
            "codestar-notifications:Unsubscribe"
          ],
          "Resource": "*",
          "Condition": {
            "StringLike": {
              "codestar-notifications:NotificationsForResource": "arn:aws:codecommit:*"
            }
          }
        },
        {
          "Sid": "CodeStarNotificationsListAccess",
          "Effect": "Allow",
          "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListTargets",
            "codestar-notifications:ListTagsforResource",
            "codestar-notifications:ListEventTypes"
          ],
          "Resource": "*"
        },
        {
          "Sid": "CodeStarNotificationsSNSTopicCreateAccess",
          "Effect": "Allow",
          "Action": [
            "sns:CreateTopic",
            "sns:SetTopicAttributes"
          ],
          "Resource": "arn:aws:sns:*:*:codestar-notifications*"
        },
        {
          "Sid": "AmazonCodeGuruReviewerFullAccess",
          "Effect": "Allow",
          "Action": [
            "codeguru-reviewer:AssociateRepository",
            "codeguru-reviewer:DescribeRepositoryAssociation",
```

```
            "codeguru-reviewer:ListRepositoryAssociations",
            "codeguru-reviewer:DisassociateRepository",
            "codeguru-reviewer:DescribeCodeReview",
            "codeguru-reviewer:ListCodeReviews"
          ],
          "Resource": "*"
        },
        {
          "Sid": "AmazonCodeGuruReviewerSLRCreation",
          "Action": "iam:CreateServiceLinkedRole",
          "Effect": "Allow",
          "Resource": "arn:aws:iam::*:role/aws-service-role/codeguru-
reviewer.amazonaws.com/AWSServiceRoleForAmazonCodeGuruReviewer",
          "Condition": {
            "StringLike": {
              "iam:AWSServiceName": "codeguru-reviewer.amazonaws.com"
            }
          }
        },
        {
          "Sid": "CloudWatchEventsManagedRules",
          "Effect": "Allow",
          "Action": [
            "events:PutRule",
            "events:PutTargets",
            "events:DeleteRule",
            "events:RemoveTargets"
          ],
          "Resource": "*",
          "Condition": {
            "StringEquals": {
              "events:ManagedBy": "codeguru-reviewer.amazonaws.com"
            }
          }
        },
        {
          "Sid": "CodeStarNotificationsChatbotAccess",
          "Effect": "Allow",
          "Action": [
            "chatbot:DescribeSlackChannelConfigurations",
            "chatbot:ListMicrosoftTeamsChannelConfigurations"
          ],
          "Resource": "*"
        },
```

```
        {
            "Sid": "CodeStarConnectionsReadOnlyAccess",
            "Effect": "Allow",
            "Action": [
                "codestar-connections:ListConnections",
                "codestar-connections:GetConnection"
            ],
            "Resource": "arn:aws:codestar-connections:*:*:connection/*"
        }
      ]
    }
```

AWSverwaltete Richtlinie: AWSCodeCommitPowerUser

Sie können die `AWSCodeCommitPowerUser`-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie ermöglicht Benutzern den Zugriff auf alle Funktionen CodeCommit und Ressourcen im Zusammenhang mit Repositorys, mit der Ausnahme, dass sie ihnen nicht erlaubt, Repositorys zu löschen oder CodeCommit repository-bezogene Ressourcen in anderen Diensten wie Amazon Events zu erstellen oder zu löschen. AWS CloudWatch Wir empfehlen, dass diese Richtlinie auf die meisten Benutzer anzuwenden.

Die AWSCodeCommitPowerUser Richtlinie enthält die folgende Grundsatzerklärung:

```
    {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Effect": "Allow",
          "Action": [
            "codecommit:AssociateApprovalRuleTemplateWithRepository",
            "codecommit:BatchAssociateApprovalRuleTemplateWithRepositories",
            "codecommit:BatchDisassociateApprovalRuleTemplateFromRepositories",
            "codecommit:BatchGet*",
            "codecommit:BatchDescribe*",
            "codecommit:Create*",
            "codecommit:DeleteBranch",
            "codecommit:DeleteFile",
            "codecommit:Describe*",
            "codecommit:DisassociateApprovalRuleTemplateFromRepository",
            "codecommit:EvaluatePullRequestApprovalRules",
            "codecommit:Get*",
            "codecommit:List*",
            "codecommit:Merge*",
```

```
            "codecommit:OverridePullRequestApprovalRules",
            "codecommit:Put*",
            "codecommit:Post*",
            "codecommit:TagResource",
            "codecommit:Test*",
            "codecommit:UntagResource",
            "codecommit:Update*",
            "codecommit:GitPull",
            "codecommit:GitPush"
          ],
          "Resource": "*"
        },
        {
          "Sid": "CloudWatchEventsCodeCommitRulesAccess",
          "Effect": "Allow",
          "Action": [
            "events:DeleteRule",
            "events:DescribeRule",
            "events:DisableRule",
            "events:EnableRule",
            "events:PutRule",
            "events:PutTargets",
            "events:RemoveTargets",
            "events:ListTargetsByRule"
          ],
          "Resource": "arn:aws:events:*:*:rule/codecommit*"
        },
        {
          "Sid": "SNSTopicAndSubscriptionAccess",
          "Effect": "Allow",
          "Action": [
            "sns:Subscribe",
            "sns:Unsubscribe"
          ],
          "Resource": "arn:aws:sns:*:*:codecommit*"
        },
        {
          "Sid": "SNSTopicAndSubscriptionReadAccess",
          "Effect": "Allow",
          "Action": [
            "sns:ListTopics",
            "sns:ListSubscriptionsByTopic",
            "sns:GetTopicAttributes"
          ],
```

```
      "Resource": "*"
    },
    {
      "Sid": "LambdaReadOnlyListAccess",
      "Effect": "Allow",
      "Action": [
        "lambda:ListFunctions"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IAMReadOnlyListAccess",
      "Effect": "Allow",
      "Action": [
        "iam:ListUsers"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IAMReadOnlyConsoleAccess",
      "Effect": "Allow",
      "Action": [
        "iam:ListAccessKeys",
        "iam:ListSSHPublicKeys",
        "iam:ListServiceSpecificCredentials"
      ],
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
      "Sid": "IAMUserSSHKeys",
      "Effect": "Allow",
      "Action": [
        "iam:DeleteSSHPublicKey",
        "iam:GetSSHPublicKey",
        "iam:ListSSHPublicKeys",
        "iam:UpdateSSHPublicKey",
        "iam:UploadSSHPublicKey"
      ],
      "Resource": "arn:aws:iam::*:user/${aws:username}"
    },
    {
      "Sid": "IAMSelfManageServiceSpecificCredentials",
      "Effect": "Allow",
      "Action": [
```

```
            "iam:CreateServiceSpecificCredential",
            "iam:UpdateServiceSpecificCredential",
            "iam:DeleteServiceSpecificCredential",
            "iam:ResetServiceSpecificCredential"
          ],
          "Resource": "arn:aws:iam::*:user/${aws:username}"
        },
        {
          "Sid": "CodeStarNotificationsReadWriteAccess",
          "Effect": "Allow",
          "Action": [
            "codestar-notifications:CreateNotificationRule",
            "codestar-notifications:DescribeNotificationRule",
            "codestar-notifications:UpdateNotificationRule",
            "codestar-notifications:Subscribe",
            "codestar-notifications:Unsubscribe"
          ],
          "Resource": "*",
          "Condition": {
            "StringLike": {
              "codestar-notifications:NotificationsForResource": "arn:aws:codecommit:*"
            }
          }
        },
        {
          "Sid": "CodeStarNotificationsListAccess",
          "Effect": "Allow",
          "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListTargets",
            "codestar-notifications:ListTagsforResource",
            "codestar-notifications:ListEventTypes"
          ],
          "Resource": "*"
        },
        {
          "Sid": "AmazonCodeGuruReviewerFullAccess",
          "Effect": "Allow",
          "Action": [
            "codeguru-reviewer:AssociateRepository",
            "codeguru-reviewer:DescribeRepositoryAssociation",
            "codeguru-reviewer:ListRepositoryAssociations",
            "codeguru-reviewer:DisassociateRepository",
            "codeguru-reviewer:DescribeCodeReview",
```

```
            "codeguru-reviewer:ListCodeReviews"
          ],
          "Resource": "*"
        },
        {
          "Sid": "AmazonCodeGuruReviewerSLRCreation",
          "Action": "iam:CreateServiceLinkedRole",
          "Effect": "Allow",
          "Resource": "arn:aws:iam::*:role/aws-service-role/codeguru-
reviewer.amazonaws.com/AWSServiceRoleForAmazonCodeGuruReviewer",
          "Condition": {
            "StringLike": {
              "iam:AWSServiceName": "codeguru-reviewer.amazonaws.com"
            }
          }
        },
        {
          "Sid": "CloudWatchEventsManagedRules",
          "Effect": "Allow",
          "Action": [
            "events:PutRule",
            "events:PutTargets",
            "events:DeleteRule",
            "events:RemoveTargets"
          ],
          "Resource": "*",
          "Condition": {
            "StringEquals": {
              "events:ManagedBy": "codeguru-reviewer.amazonaws.com"
            }
          }
        },
        {
          "Sid": "CodeStarNotificationsChatbotAccess",
          "Effect": "Allow",
          "Action": [
            "chatbot:DescribeSlackChannelConfigurations",
            "chatbot:ListMicrosoftTeamsChannelConfigurations"
          ],
          "Resource": "*"
        },
        {
            "Sid": "CodeStarConnectionsReadOnlyAccess",
            "Effect": "Allow",
```

```
            "Action": [
                "codestar-connections:ListConnections",
                "codestar-connections:GetConnection"
            ],
            "Resource": "arn:aws:codestar-connections:*:*:connection/*"
        }
      ]
    }
```

AWSverwaltete Richtlinie: AWSCodeCommitReadOnly

Sie können die `AWSCodeCommitReadOnly`-Richtlinie an Ihre IAM-Identitäten anfügen. Diese Richtlinie gewährt schreibgeschützten Zugriff auf CodeCommit und Repository-bezogene Ressourcen in anderen AWS Diensten sowie die Möglichkeit, eigene CodeCommit zugehörige Ressourcen zu erstellen und zu verwalten (z. B. Git-Anmeldeinformationen und SSH-Schlüssel, die ihre IAM-Benutzer beim Zugriff auf Repositorys verwenden können). Wenden Sie diese Richtlinie auf Benutzer an, denen Sie die Möglichkeit geben möchten, den Inhalt eines Repositorys zu lesen, ohne jedoch dessen Inhalt zu ändern.

Die AWSCodeCommitReadOnly Richtlinie enthält die folgende Grundsatzerklärung:

```
    {
       "Version":"2012-10-17",
       "Statement":[
          {
             "Effect":"Allow",
             "Action":[
                "codecommit:BatchGet*",
                "codecommit:BatchDescribe*",
                "codecommit:Describe*",
                "codecommit:EvaluatePullRequestApprovalRules",
                "codecommit:Get*",
                "codecommit:List*",
                "codecommit:GitPull"
             ],
             "Resource":"*"
          },
          {
             "Sid":"CloudWatchEventsCodeCommitRulesReadOnlyAccess",
             "Effect":"Allow",
             "Action":[
                "events:DescribeRule",
```

```
            "events:ListTargetsByRule"
         ],
         "Resource":"arn:aws:events:*:*:rule/codecommit*"
      },
      {
         "Sid":"SNSSubscriptionAccess",
         "Effect":"Allow",
         "Action":[
            "sns:ListTopics",
            "sns:ListSubscriptionsByTopic",
            "sns:GetTopicAttributes"
         ],
         "Resource":"*"
      },
      {
         "Sid":"LambdaReadOnlyListAccess",
         "Effect":"Allow",
         "Action":[
            "lambda:ListFunctions"
         ],
         "Resource":"*"
      },
      {
         "Sid":"IAMReadOnlyListAccess",
         "Effect":"Allow",
         "Action":[
            "iam:ListUsers"
         ],
         "Resource":"*"
      },
      {
         "Sid":"IAMReadOnlyConsoleAccess",
         "Effect":"Allow",
         "Action":[
            "iam:ListAccessKeys",
            "iam:ListSSHPublicKeys",
            "iam:ListServiceSpecificCredentials",
            "iam:GetSSHPublicKey"
         ],
         "Resource":"arn:aws:iam::*:user/${aws:username}"
      },
      {
         "Sid":"CodeStarNotificationsReadOnlyAccess",
         "Effect":"Allow",
```

```
            "Action":[
               "codestar-notifications:DescribeNotificationRule"
            ],
            "Resource":"*",
            "Condition":{
               "StringLike":{
                  "codestar-
notifications:NotificationsForResource":"arn:aws:codecommit:*"
               }
            }
         },
         {
            "Sid":"CodeStarNotificationsListAccess",
            "Effect":"Allow",
            "Action":[
               "codestar-notifications:ListNotificationRules",
               "codestar-notifications:ListEventTypes",
               "codestar-notifications:ListTargets"
            ],
            "Resource":"*"
         },
         {
            "Sid": "AmazonCodeGuruReviewerReadOnlyAccess",
            "Effect": "Allow",
            "Action": [
               "codeguru-reviewer:DescribeRepositoryAssociation",
               "codeguru-reviewer:ListRepositoryAssociations",
               "codeguru-reviewer:DescribeCodeReview",
               "codeguru-reviewer:ListCodeReviews"
            ],
            "Resource": "*"
         },
         {
           "Sid": "CodeStarConnectionsReadOnlyAccess",
           "Effect": "Allow",
           "Action": [
               "codestar-connections:ListConnections",
               "codestar-connections:GetConnection"
           ],
           "Resource": "arn:aws:codestar-connections:*:*:connection/*"
        }
    ]
}
```

CodeCommit verwaltete Richtlinien und Benachrichtigungen

AWS CodeCommit unterstützt Benachrichtigungen, die Benutzer über wichtige Änderungen in Repositorys informieren können. Zu den verwalteten Richtlinien CodeCommit gehören Richtlinienerklärungen für die Benachrichtigungsfunktion. Weitere Informationen finden Sie unter Was sind Benachrichtigungen?.

Berechtigungen in Zusammenhang mit Benachrichtigungen in verwalteten Vollzugriffsrichtlinien

Die von `AWSCodeCommitFullAccess` verwaltete Richtlinie enthält die folgenden Anweisungen, um den vollständigen Zugriff auf Benachrichtigungen zu ermöglichen. Benutzer, für die diese verwaltete Richtlinie gilt, können auch Amazon SNS SNS-Themen für Benachrichtigungen erstellen und verwalten, Benutzer für Themen abonnieren und abbestellen, Themen auflisten, die als Ziele für Benachrichtigungsregeln ausgewählt werden sollen, und für Slack konfigurierte AWS Chatbot Clients auflisten.

```
    {
        "Sid": "CodeStarNotificationsReadWriteAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:CreateNotificationRule",
            "codestar-notifications:DescribeNotificationRule",
            "codestar-notifications:UpdateNotificationRule",
            "codestar-notifications:DeleteNotificationRule",
            "codestar-notifications:Subscribe",
            "codestar-notifications:Unsubscribe"
        ],
        "Resource": "*",
        "Condition" : {
            "StringLike" : {"codestar-notifications:NotificationsForResource" :
"arn:aws:codecommit:*"}
        }
    },
    {
        "Sid": "CodeStarNotificationsListAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListTargets",
            "codestar-notifications:ListTagsforResource,"
            "codestar-notifications:ListEventTypes"
        ],
```

```
        "Resource": "*"
    },
    {
        "Sid": "CodeStarNotificationsSNSTopicCreateAccess",
        "Effect": "Allow",
        "Action": [
            "sns:CreateTopic",
            "sns:SetTopicAttributes"
        ],
        "Resource": "arn:aws:sns:*:*:codestar-notifications*"
    },
    {
        "Sid": "CodeStarNotificationsChatbotAccess",
        "Effect": "Allow",
        "Action": [
            "chatbot:DescribeSlackChannelConfigurations",
            "chatbot:ListMicrosoftTeamsChannelConfigurations"
          ],
       "Resource": "*"
    }
```

Berechtigungen in Zusammenhang mit Benachrichtigungen in schreibgeschützten verwalteten Richtlinien

Die verwaltete Richtlinie `AWSCodeCommitReadOnlyAccess` enthält die folgenden Anweisungen, um schreibgeschützten Zugriff auf Benachrichtigungen zu ermöglichen. Benutzer mit dieser verwalteten Richtlinie können Benachrichtigungen für Ressourcen anzeigen, sie können sie jedoch nicht erstellen, verwalten oder abonnieren.

```
   {
        "Sid": "CodeStarNotificationsPowerUserAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:DescribeNotificationRule"
        ],
        "Resource": "*",
        "Condition" : {
            "StringLike" : {"codestar-notifications:NotificationsForResource" :
 "arn:aws:codecommit:*"}
        }
    },
    {
        "Sid": "CodeStarNotificationsListAccess",
```

```
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListEventTypes",
            "codestar-notifications:ListTargets"
        ],
        "Resource": "*"
    }
```

Berechtigungen in Zusammenhang mit Benachrichtigungen in anderen verwalteten Richtlinien

Die verwaltete Richtlinie `AWSCodeCommitPowerUser` enthält die folgenden Anweisungen, mit denen Sie Benutzern erlauben können, Benachrichtigungen zu erstellen, zu bearbeiten und zu abonnieren. Benutzer können Benachrichtigungsregeln nicht löschen und auch keine Tags für Ressourcen verwalten.

```
    {
        "Sid": "CodeStarNotificationsReadWriteAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:CreateNotificationRule",
            "codestar-notifications:DescribeNotificationRule",
            "codestar-notifications:UpdateNotificationRule",
            "codestar-notifications:DeleteNotificationRule",
            "codestar-notifications:Subscribe",
            "codestar-notifications:Unsubscribe"
        ],
        "Resource": "*",
        "Condition" : {
            "StringLike" : {"codestar-notifications:NotificationsForResource" :
 "arn:aws:codecommit*"}
        }
    },
    {
        "Sid": "CodeStarNotificationsListAccess",
        "Effect": "Allow",
        "Action": [
            "codestar-notifications:ListNotificationRules",
            "codestar-notifications:ListTargets",
            "codestar-notifications:ListTagsforResource",
            "codestar-notifications:ListEventTypes"
        ],
        "Resource": "*"
```

```
    },
    {
        "Sid": "SNSTopicListAccess",
        "Effect": "Allow",
        "Action": [
            "sns:ListTopics"
        ],
        "Resource": "*"
    },
    {
        "Sid": "CodeStarNotificationsChatbotAccess",
        "Effect": "Allow",
        "Action": [
            "chatbot:DescribeSlackChannelConfigurations",
            "chatbot:ListMicrosoftTeamsChannelConfigurations"
          ],
       "Resource": "*"
    }
```

Weitere Informationen zu IAM und Benachrichtigungen finden Sie unter Identity and Access Management für AWS CodeStar Benachrichtigungen.

AWS CodeCommitverwaltete Richtlinien und Amazon CodeGuru Reviewer

CodeCommit unterstützt Amazon CodeGuru Reviewer, einen automatisierten Code-Review-Service, der Programmanalyse und maschinelles Lernen nutzt, um häufig auftretende Probleme zu erkennen und Korrekturen in Ihrem Java- oder Python-Code zu empfehlen. Zu den verwalteten Richtlinien CodeCommit gehören auch Richtlinienerklärungen für die CodeGuru Reviewer-Funktionalität. Weitere Informationen finden Sie unter Was ist Amazon CodeGuru Reviewer.

Berechtigungen im Zusammenhang mit dem CodeGuru Prüfer in AWSCodeCommitFullAccess

Die `AWSCodeCommitFullAccess` verwaltete Richtlinie enthält die folgenden Anweisungen, mit denen CodeGuru Prüfer CodeCommit Repositorys zugeordnet oder getrennt werden können. Benutzer, auf die diese verwaltete Richtlinie angewendet wurde, können auch den Zuordnungsstatus zwischen CodeCommit Repositorys und CodeGuru Reviewer sowie den Status von Review-Jobs für Pull Requests einsehen.

```
    {
      "Sid": "AmazonCodeGuruReviewerFullAccess",
      "Effect": "Allow",
```

```
      "Action": [
        "codeguru-reviewer:AssociateRepository",
        "codeguru-reviewer:DescribeRepositoryAssociation",
        "codeguru-reviewer:ListRepositoryAssociations",
        "codeguru-reviewer:DisassociateRepository",
        "codeguru-reviewer:DescribeCodeReview",
        "codeguru-reviewer:ListCodeReviews"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AmazonCodeGuruReviewerSLRCreation",
      "Action": "iam:CreateServiceLinkedRole",
      "Effect": "Allow",
      "Resource": "arn:aws:iam::*:role/aws-service-role/codeguru-
reviewer.amazonaws.com/AWSServiceRoleForAmazonCodeGuruReviewer",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "codeguru-reviewer.amazonaws.com"
        }
      }
    },
    {
      "Sid": "CloudWatchEventsManagedRules",
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "events:DeleteRule",
        "events:RemoveTargets"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:ManagedBy": "codeguru-reviewer.amazonaws.com"
        }
      }
    }
```

Berechtigungen im Zusammenhang mit dem CodeGuru Prüfer in AWSCodeCommitPowerUser

Die `AWSCodeCommitPowerUser` verwaltete Richtlinie umfasst die folgenden Anweisungen, die es Benutzern ermöglichen, Repositorys dem CodeGuru Prüfer zuzuordnen und die Verknüpfung

aufzuheben, den Zuordnungsstatus einzusehen und den Status von Review-Jobs für Pull Requests einzusehen.

```
    {
      "Sid": "AmazonCodeGuruReviewerFullAccess",
      "Effect": "Allow",
      "Action": [
        "codeguru-reviewer:AssociateRepository",
        "codeguru-reviewer:DescribeRepositoryAssociation",
        "codeguru-reviewer:ListRepositoryAssociations",
        "codeguru-reviewer:DisassociateRepository",
        "codeguru-reviewer:DescribeCodeReview",
        "codeguru-reviewer:ListCodeReviews"
      ],
      "Resource": "*"
    },
    {
      "Sid": "AmazonCodeGuruReviewerSLRCreation",
      "Action": "iam:CreateServiceLinkedRole",
      "Effect": "Allow",
      "Resource": "arn:aws:iam::*:role/aws-service-role/codeguru-
reviewer.amazonaws.com/AWSServiceRoleForAmazonCodeGuruReviewer",
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": "codeguru-reviewer.amazonaws.com"
        }
      }
    },
    {
      "Sid": "CloudWatchEventsManagedRules",
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "events:DeleteRule",
        "events:RemoveTargets"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "events:ManagedBy": "codeguru-reviewer.amazonaws.com"
        }
      }
```

```
    }
```

Berechtigungen im Zusammenhang mit Reviewer in CodeGuru AWSCodeCommitReadOnly

Die `AWSCodeCommitReadOnlyAccess` verwaltete Richtlinie umfasst die folgenden Anweisungen, um nur Lesezugriff auf den Zuordnungsstatus des CodeGuru Prüfers zu gewähren und den Status von Überprüfungsaufträgen für Pull-Requests anzuzeigen. Benutzer, denen diese verwaltete Richtlinie zugewiesen wurde, können Repositorys nicht zuordnen und deren Zuordnung nicht aufheben.

```
    {
     "Sid": "AmazonCodeGuruReviewerReadOnlyAccess",
     "Effect": "Allow",
     "Action": [
           "codeguru-reviewer:DescribeRepositoryAssociation",
           "codeguru-reviewer:ListRepositoryAssociations",
           "codeguru-reviewer:DescribeCodeReview",
           "codeguru-reviewer:ListCodeReviews"
     ],
     "Resource": "*"
    }
```

Rolle im Zusammenhang mit dem Service von Amazon CodeGuru Reviewer

Wenn Sie CodeGuru Reviewer ein Repository zuordnen, wird eine serviceverknüpfte Rolle erstellt, sodass CodeGuru Reviewer Probleme erkennen und Korrekturen für Java- oder Python-Code in Pull-Requests empfehlen kann. Die Service-verknüpfte Rolle wird AWSServiceRoleForAmazonCodeGuruReviewer benannt. Weitere Informationen finden Sie unter Verwenden von serviceverknüpften Rollen für Amazon CodeGuru Reviewer.

Weitere Informationen finden Sie unter AWS-verwaltete Richtlinien im IAM-Benutzerhandbuch.

CodeCommit Aktualisierungen der AWS verwalteten Richtlinien

Hier finden Sie Informationen zu Aktualisierungen AWS verwalteter Richtlinien, die CodeCommit seit Beginn der Nachverfolgung dieser Änderungen durch diesen Dienst vorgenommen wurden. Abonnieren Sie den RSS-Feed auf, um automatische Benachrichtigungen über Änderungen an dieser Seite zu erhaltenDokumentverlauf für das AWS CodeCommit-Benutzerhandbuch.

Änderung	Beschreibung	Datum
AWSverwaltete Richtlinie: AWSCodeCommitFullAccessund AWSverwaltete Richtlinie: AWSCodeCommitPowerUser — Aktualisierung vorhandener Richtlinien	CodeCommit diesen Richtlinien wurde eine Berechtigung hinzugefügt, um einen zusätzlichen Benachrichtigungstyp zu unterstützen, indem AWS Chatbot Die AWSCodeCommitFullAccess Richtlinien AWSCodeCommitPowerUser und wurden geändert, um eine Berechtigung hinzuzufügen,`chatbot:ListMicrosoftTeamsChannelConfigurations` .	16. Mai 2023
AWSverwaltete Richtlinie: AWSCodeCommitReadOnly – Aktualisierung auf eine bestehende Richtlinie	CodeCommit eine doppelte Berechtigung wurde aus der Richtlinie entfernt. Die AWSCodeCommitReadOnly wurde geändert, um eine doppelte Berechtigung zu entfernen,`"iam:ListAccessKeys"` .	18. August 2021
CodeCommit hat begonnen, Änderungen zu verfolgen	CodeCommit hat begonnen, Änderungen für die AWS verwalteten Richtlinien zu verfolgen.	18. August 2021

Beispiele für vom Kunden verwaltete Richtlinien

Sie können Ihre eigenen benutzerdefinierten IAM-Richtlinien erstellen, um Berechtigungen für CodeCommit Aktionen und Ressourcen zu gewähren. Die benutzerdefinierten Richtlinien können Sie

dann den IAM-Benutzern oder -Gruppen zuweisen, die diese Berechtigungen benötigen. Sie können auch Ihre eigenen benutzerdefinierten IAM-Richtlinien für die Integration zwischen CodeCommit und anderen AWS Diensten erstellen.

Themen

- Beispiele für vom Kunden verwaltete Identitätsrichtlinien
- Beispiele für vom Kunden verwaltete Integrationsrichtlinien

Beispiele für vom Kunden verwaltete Identitätsrichtlinien

Das folgende Beispiel für IAM-Richtlinien gewährt Berechtigungen für verschiedene CodeCommit Aktionen. Verwenden Sie sie, um den CodeCommit Zugriff für Ihre IAM-Benutzer und -Rollen einzuschränken. Diese Richtlinien steuern die Durchführung von Aktionen mit der CodeCommit -Konsole, API, AWS SDKs oder der AWS CLI.

> **Note**
>
> In allen Beispielen werden die Region USA West (Oregon) (us-west-2) und fiktive Konto-IDs verwendet.

Beispiele

- Beispiel 1: Erlauben Sie einem Benutzer, CodeCommit Operationen in einem einzigen System auszuführen AWS-Region
- Beispiel 2: Erlaube einem Benutzer, Git für ein einzelnes Repository zu verwenden
- Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine Verbindung herstellt, Zugriff auf ein Repository
- Beispiel 4: Aktionen für Branches verweigern oder zulassen
- Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen

Beispiel 1: Erlauben Sie einem Benutzer, CodeCommit Operationen in einem einzigen System auszuführen AWS-Region

In der folgenden Berechtigungsrichtlinie wird ein Platzhalterzeichen (`"codecommit:*"`) verwendet, damit Benutzer alle CodeCommit Aktionen in der Region us-east-2 und nicht von anderen aus ausführen können. AWS-Regionen

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "codecommit:*",
            "Resource": "arn:aws:codecommit:us-east-2:111111111111:*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": "us-east-2"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "codecommit:ListRepositories",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestedRegion": "us-east-2"
                }
            }
        }
    ]
}
```

Beispiel 2: Erlaube einem Benutzer, Git für ein einzelnes Repository zu verwenden

CodeCommitIn gelten die `GitPull` IAM-Richtlinienberechtigungen für jeden Git-Client-Befehl CodeCommit, von dem Daten abgerufen werden git fetchgit clone, einschließlich, usw. In ähnlicher Weise gelten die `GitPush` IAM-Richtlinienberechtigungen für jeden Git-Client-Befehl, an den Daten gesendet CodeCommit werden. Wenn die `GitPush` IAM-Richtlinienberechtigung beispielsweise auf gesetzt ist`Allow`, kann ein Benutzer das Löschen eines Branches mithilfe des Git-Protokolls vorantreiben. Dieser Push wird nicht von den Berechtigungen beeinflusst, die für diesen IAM-Benutzer auf den `DeleteBranch` Vorgang angewendet wurden. Die `DeleteBranch`-Berechtigung

bezieht sich auf Aktionen, die mit der Konsole, der AWS CLI den SDKs und der API ausgeführt werden, aber nicht mit dem Git-Protokoll.

Das folgende Beispiel ermöglicht es dem angegebenen Benutzer, Daten aus dem angegebenen CodeCommit Repository abzurufen und dorthin zu pushen: `MyDemoRepo`

```
{
  "Version": "2012-10-17",
  "Statement" : [
    {
      "Effect" : "Allow",
      "Action" : [
        "codecommit:GitPull",
        "codecommit:GitPush"
      ],
      "Resource" : "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo"
    }
  ]
}
```

Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine Verbindung herstellt, Zugriff auf ein Repository

Sie können eine Richtlinie erstellen, die es Benutzern nur dann erlaubt, sich mit einem CodeCommit-Repository zu verbinden, wenn ihre IP-Adresse innerhalb eines bestimmten IP-Adressbereichs liegt. Dazu gibt es zwei gleichwertige Ansätze. Sie können eine `Deny` Richtlinie erstellen, die CodeCommit Operationen verbietet, wenn sich die IP-Adresse des Benutzers nicht in einem bestimmten Block befindet, oder Sie können eine `Allow` Richtlinie erstellen, die CodeCommit Operationen zulässt, wenn sich die IP-Adresse des Benutzers innerhalb eines bestimmten Blocks befindet.

Sie können eine `Deny`-Richtlinie erstellen, die den Zugriff für alle Benutzer verweigert, die sich nicht aus einem bestimmten IP-Adressbereich stammen. Beispielsweise können Sie die verwaltete Richtlinie AWSCodeCommitPowerUser und eine vom Kunden verwaltete Richtlinie allen Benutzern anfügen, die Zugriff auf Ihr Repository benötigen. Die folgende Beispielrichtlinie verweigert Benutzern, deren IP-Adressen nicht innerhalb des angegebenen IP-Adressblocks 203.0.113.0/16 liegen, alle CodeCommit Berechtigungen:

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

```
         "Effect": "Deny",
         "Action": [
            "codecommit:*"
         ],
         "Resource": "*",
         "Condition": {
            "NotIpAddress": {
               "aws:SourceIp": [
                  "203.0.113.0/16"
               ]
            }
         }
      }
   ]
}
```

Mit der folgenden Beispielrichtlinie kann der angegebene Benutzer nur dann auf ein CodeCommit Repository zugreifen, das MyDemoRepo mit den entsprechenden Berechtigungen der AWSCodeCommitPowerUser verwalteten Richtlinie benannt ist, wenn sich seine IP-Adresse innerhalb des angegebenen Adressblocks 203.0.113.0/16 befindet:

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Effect": "Allow",
         "Action": [
            "codecommit:BatchGetRepositories",
            "codecommit:CreateBranch",
            "codecommit:CreateRepository",
            "codecommit:Get*",
            "codecommit:GitPull",
            "codecommit:GitPush",
            "codecommit:List*",
            "codecommit:Put*",
            "codecommit:Post*",
            "codecommit:Merge*",
            "codecommit:TagResource",
            "codecommit:Test*",
            "codecommit:UntagResource",
            "codecommit:Update*"
         ],
         "Resource": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
```

```
         "Condition": {
            "IpAddress": {
               "aws:SourceIp": [
                  "203.0.113.0/16"
               ]
            }
         }
      }
   ]
}
```

Beispiel 4: Aktionen für Branches verweigern oder zulassen

Sie können eine Richtlinie erstellen, die Benutzern die Berechtigungen für Aktionen verweigert, die Sie für eine oder mehrere Verzweigungen angeben. Alternativ können Sie eine Richtlinie erstellen, die Aktionen für eine oder mehrere Verzweigungen erlaubt, die sie sonst nicht in anderen Verzweigungen eines Repositories haben. Sie können diese Richtlinien mit den entsprechenden verwalteten (vordefinierten) Richtlinien verwenden. Weitere Informationen finden Sie unter Beschränken Sie die Anzahl von Pushs und Merges auf Branches in AWS CodeCommit.

Sie können beispielsweise eine `Deny` Richtlinie erstellen, die Benutzern die Möglichkeit verweigert, Änderungen an einem Branch namens main vorzunehmen, einschließlich des Löschens dieses Branches, in einem Repository mit dem Namen *`MyDemoRepo`*. Sie können diese Richtlinien mit der verwalteten Richtlinie AWSCodeCommitPowerUser verwenden. Benutzer, auf die diese beiden Richtlinien angewendet wurden, könnten Branches erstellen und löschen, Pull Requests erstellen und alle anderen Aktionen ausführen, sofern dies von erlaubt ist. Sie wären jedoch nicht in der Lage AWSCodeCommitPowerUser, Änderungen an den Branch mit dem Namen main zu pushen, eine Datei im Main Branch in der CodeCommit Konsole hinzuzufügen oder zu bearbeiten oder Branches oder eine Pull-Anfrage mit dem Main Branch zusammenzuführen. Da `Deny` auf `GitPush` angewendet wird,, müssen Sie eine `Null`-Anweisung in die Richtlinie aufnehmen, um die Analyse anfänglicher `GitPush`-Aufrufe auf Gültigkeit zu gestatten, wenn Benutzer Sendeoperationen von ihren lokalen Repositories aus ausführen.

> **Tip**
>
> Wenn Sie eine Richtlinie erstellen möchten, die für alle Branches mit dem Namen main in allen Repositorys in Ihrem Amazon Web Services Services-Konto gilt`Resource`, geben Sie statt eines Repository-ARN ein Sternchen (*) an.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "codecommit:GitPush",
                "codecommit:DeleteBranch",
                "codecommit:PutFile",
                "codecommit:Merge*"
            ],
            "Resource": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
            "Condition": {
                "StringEqualsIfExists": {
                    "codecommit:References": [
                        "refs/heads/main"
                    ]
                },
                "Null": {
                    "codecommit:References": "false"
                }
            }
        }
    ]
}
```

Die folgende Beispielrichtlinie ermöglicht es einem Benutzer, Änderungen an einem Branch namens main in allen Repositorys eines Amazon Web Services Services-Kontos vorzunehmen. Sie erlaubt keine Änderungen an anderen Zweigen. Sie können diese Richtlinie zusammen mit der AWSCodeCommitReadOnly verwalteten Richtlinie verwenden, um automatisierte Pushs an das Repository im Hauptzweig zu ermöglichen. Der Effekt ist `Allow`, deshalb würde diese Beispielrichtlinie nicht mit verwalteten Richtlinien funktionieren, wie z. B. AWSCodeCommitPowerUser.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "codecommit:GitPush",
                "codecommit:Merge*"
```

```
            ],
            "Resource": "*",
            "Condition": {
                "StringEqualsIfExists": {
                    "codecommit:References": [
                        "refs/heads/main"
                    ]
                }
            }
        }
    ]
}
```

Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen

Sie können eine Richtlinie erstellen, die Aktionen für Repositorys auf der Grundlage der mit diesen Repositorys verknüpften AWS Tags erlaubt oder verweigert, und diese Richtlinien dann auf die IAM-Gruppen anwenden, die Sie für die Verwaltung von IAM-Benutzern konfigurieren. *Sie können beispielsweise eine Richtlinie erstellen, die alle CodeCommit Aktionen in beliebigen Repositorys mit dem AWS Tag-Schlüssel Status und dem Schlüsselwert Secret verweigert, und diese Richtlinie dann auf die IAM-Gruppe anwenden, die Sie für allgemeine Entwickler (Entwickler) erstellt haben.* Anschließend müssen Sie sicherstellen, dass die Entwickler, die an diesen markierten Repositorys arbeiten, nicht Mitglieder dieser allgemeinen *Entwicklergruppe* sind, sondern einer anderen IAM-Gruppe angehören, auf die die restriktive Richtlinie nicht angewendet wurde (). SecretDevelopers

Im folgenden Beispiel werden alle CodeCommit Aktionen für Repositorys verweigert, die mit dem Schlüssel Status und dem Schlüsselwert Secret gekennzeichnet sind:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "codecommit:Associate*",
        "codecommit:Batch*",
        "codecommit:CancelUploadArchive",
        "codecommit:CreateBranch",
        "codecommit:CreateCommit",
```

```
        "codecommit:CreatePullRequest*",
        "codecommit:CreateRepository",
        "codecommit:CreateUnreferencedMergeCommit",
        "codecommit:DeleteBranch",
        "codecommit:DeleteCommentContent",
        "codecommit:DeleteFile",
        "codecommit:DeletePullRequest*",
        "codecommit:DeleteRepository",
        "codecommit:Describe*",
        "codecommit:DisassociateApprovalRuleTemplateFromRepository",
        "codecommit:EvaluatePullRequestApprovalRules",
        "codecommit:GetBlob",
        "codecommit:GetBranch",
        "codecommit:GetComment*",
        "codecommit:GetCommit",
        "codecommit:GetDifferences*",
        "codecommit:GetFile",
        "codecommit:GetFolder",
        "codecommit:GetMerge*",
        "codecommit:GetObjectIdentifier",
        "codecommit:GetPullRequest*",
        "codecommit:GetReferences",
        "codecommit:GetRepository*",
        "codecommit:GetTree",
        "codecommit:GetUploadArchiveStatus",
        "codecommit:Git*",
        "codecommit:ListAssociatedApprovalRuleTemplatesForRepository",
        "codecommit:ListBranches",
        "codecommit:ListPullRequests",
        "codecommit:ListTagsForResource",
        "codecommit:Merge*",
        "codecommit:OverridePullRequestApprovalRules",
        "codecommit:Post*",
        "codecommit:Put*",
        "codecommit:TagResource",
        "codecommit:TestRepositoryTriggers",
        "codecommit:UntagResource",
        "codecommit:UpdateComment",
        "codecommit:UpdateDefaultBranch",
        "codecommit:UpdatePullRequest*",
        "codecommit:UpdateRepository*",
        "codecommit:UploadArchive"
      ],
      "Resource": "*",
```

```
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Status": "Secret"
        }
      }
    }
  ]
}
```

Sie können diese Strategie weiter verfeinern, indem Sie bestimmte Repositorys und nicht alle Repositorys als Ressourcen angeben. Sie können auch Richtlinien erstellen, die CodeCommit Aktionen für alle Repositorys zulassen, die nicht mit bestimmten Tags gekennzeichnet sind. Die folgende Richtlinie erlaubt beispielsweise das Äquivalent von AWSCodeCommitPowerUserBerechtigungen für CodeCommit Aktionen, mit der Ausnahme, dass sie nur Aktionen für Repositorys zulässt CodeCommit , die nicht mit den angegebenen Tags gekennzeichnet sind:

> **Note**
>
> Dieses Richtlinienbeispiel umfasst nur Aktionen für CodeCommit. Es umfasst keine Aktionen für andere AWS Dienste, die in der AWSCodeCommitPowerUserverwalteten Richtlinie enthalten sind. Weitere Informationen finden Sie unter AWSverwaltete Richtlinie: AWSCodeCommitPowerUser..

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:Associate*",
        "codecommit:Batch*",
        "codecommit:CancelUploadArchive",
        "codecommit:CreateBranch",
        "codecommit:CreateCommit",
        "codecommit:CreatePullRequest*",
        "codecommit:CreateRepository",
        "codecommit:CreateUnreferencedMergeCommit",
        "codecommit:DeleteBranch",
        "codecommit:DeleteCommentContent",
```

```
        "codecommit:DeleteFile",
        "codecommit:DeletePullRequest*",
        "codecommit:Describe*",
        "codecommit:DisassociateApprovalRuleTemplateFromRepository",
        "codecommit:EvaluatePullRequestApprovalRules",
        "codecommit:GetBlob",
        "codecommit:GetBranch",
        "codecommit:GetComment*",
        "codecommit:GetCommit",
        "codecommit:GetDifferences*",
        "codecommit:GetFile",
        "codecommit:GetFolder",
        "codecommit:GetMerge*",
        "codecommit:GetObjectIdentifier",
        "codecommit:GetPullRequest*",
        "codecommit:GetReferences",
        "codecommit:GetRepository*",
        "codecommit:GetTree",
        "codecommit:GetUploadArchiveStatus",
        "codecommit:Git*",
        "codecommit:ListAssociatedApprovalRuleTemplatesForRepository",
        "codecommit:ListBranches",
        "codecommit:ListPullRequests",
        "codecommit:ListTagsForResource",
        "codecommit:Merge*",
        "codecommit:OverridePullRequestApprovalRules",
        "codecommit:Post*",
        "codecommit:Put*",
        "codecommit:TagResource",
        "codecommit:TestRepositoryTriggers",
        "codecommit:UntagResource",
        "codecommit:UpdateComment",
        "codecommit:UpdateDefaultBranch",
        "codecommit:UpdatePullRequest*",
        "codecommit:UpdateRepository*",
        "codecommit:UploadArchive"
      ],
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceTag/Status": "Secret",
          "aws:ResourceTag/Team": "Saanvi"
        }
      }
```

```
    },
    {
      "Effect": "Allow",
      "Action": [
        "codecommit:CreateApprovalRuleTemplate",
        "codecommit:GetApprovalRuleTemplate",
        "codecommit:ListApprovalRuleTemplates",
        "codecommit:ListRepositories",
        "codecommit:ListRepositoriesForApprovalRuleTemplate",
        "codecommit:UpdateApprovalRuleTemplateContent",
        "codecommit:UpdateApprovalRuleTemplateDescription",
        "codecommit:UpdateApprovalRuleTemplateName"
      ],
      "Resource": "*"
    }
  ]
}
```

Beispiele für vom Kunden verwaltete Integrationsrichtlinien

Dieser Abschnitt enthält Beispiele für vom Kunden verwaltete Benutzerrichtlinien, die Berechtigungen für Integrationen zwischen CodeCommit und anderen AWS Diensten gewähren. Spezifische Beispiele für Richtlinien, die den kontoübergreifenden Zugriff auf ein CodeCommit-Repository erlauben, finden Sie unter Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen.

> **Note**
>
> Alle Beispiele verwenden die Region USA West (Oregon) (us-west-2), wenn eine erforderlich AWS-Region ist, und enthalten fiktive Konto-IDs.

Beispiele

- Beispiel 1: Erstellen Sie eine Richtlinie, die den kontoübergreifenden Zugriff auf ein Amazon SNS SNS-Thema ermöglicht
- Beispiel 2: Erstellen Sie eine Amazon Simple Notification Service (Amazon SNS) -Themenrichtlinie, damit Amazon CloudWatch Events Ereignisse zu diesem Thema veröffentlichen CodeCommit kann
- Beispiel 3: Erstellen Sie eine Richtlinie für die AWS Lambda Integration mit einem Trigger CodeCommit

Beispiel 1: Erstellen Sie eine Richtlinie, die den kontoübergreifenden Zugriff auf ein Amazon SNS SNS-Thema ermöglicht

Sie können ein CodeCommit Repository so konfigurieren, dass Code-Pushs oder andere Ereignisse Aktionen auslösen, z. B. das Senden einer Benachrichtigung von Amazon Simple Notification Service (Amazon SNS). Wenn Sie das Amazon SNS SNS-Thema mit demselben Konto erstellen, das für die Erstellung des CodeCommit Repositorys verwendet wurde, müssen Sie keine zusätzlichen IAM-Richtlinien oder -Berechtigungen konfigurieren. Sie können das Thema und dann den Auslöser für das Repository erstellen. Weitere Informationen finden Sie unter Einen Auslöser für ein Amazon SNS SNS-Thema erstellen.

Wenn Sie Ihren Auslöser jedoch so konfigurieren möchten, dass er ein Amazon SNS SNS-Thema in einem anderen Amazon Web Services Services-Konto verwendet, müssen Sie dieses Thema zunächst mit einer Richtlinie konfigurieren, die es ermöglicht, CodeCommit zu diesem Thema zu veröffentlichen. Öffnen Sie von diesem anderen Konto aus die Amazon SNS SNS-Konsole, wählen Sie das Thema aus der Liste aus und wählen Sie für Andere Themenaktionen die Option Themenrichtlinie bearbeiten aus. Ändern Sie auf der Registerkarte „Erweitert“ die Richtlinie für das Thema, sodass es CodeCommit zu diesem Thema veröffentlicht werden kann. Wenn es sich bei der Richtlinie beispielsweise um die Standardrichtlinie handelt, würden Sie die Richtlinie wie folgt ändern und die Elemente in *rotem kursivem Text* so ändern, dass sie den Werten für Ihr Repository, Ihr Amazon SNS SNS-Thema und Ihr Konto entsprechen:

```
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__default_statement_ID",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "sns:Subscribe",
        "sns:ListSubscriptionsByTopic",
        "sns:DeleteTopic",
        "sns:GetTopicAttributes",
        "sns:Publish",
        "sns:RemovePermission",
        "sns:AddPermission",        "sns:SetTopicAttributes"
      ],
```

```
      "Resource": "arn:aws:sns:us-east-2:111111111111:NotMySNSTopic",
      "Condition": {
        "StringEquals": {
          "AWS:SourceOwner": "111111111111"
        }
      }
     },
     {
      "Sid": "CodeCommit-Policy_ID",
      "Effect": "Allow",
      "Principal": {
        "Service": "codecommit.amazonaws.com"
      },
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:us-east-2:111111111111:NotMySNSTopic",
      "Condition": {
        "StringEquals": {
          "AWS:SourceArn": "arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
          "AWS:SourceAccount": "111111111111"
        }
      }
    }
  ]
}
```

Beispiel 2: Erstellen Sie eine Amazon Simple Notification Service (Amazon SNS) -Themenrichtlinie, damit Amazon CloudWatch Events Ereignisse zu diesem Thema veröffentlichen CodeCommit kann

Sie können CloudWatch Ereignisse so konfigurieren, dass sie bei Eintreten von Ereignissen, einschließlich CodeCommit Ereignissen, in einem Amazon SNS SNS-Thema veröffentlicht werden. Dazu müssen Sie sicherstellen, dass CloudWatch Events berechtigt ist, Ereignisse zu Ihrem Amazon SNS SNS-Thema zu veröffentlichen, indem Sie eine Richtlinie für das Thema erstellen oder eine bestehende Richtlinie für das Thema ändern, die der folgenden ähnelt:

```
{
  "Version": "2008-10-17",
  "Id": "__default_policy_ID",
  "Statement": [
    {
      "Sid": "__default_statement_ID",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
```

```
      },
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic",
      "Condition": {
        "StringEquals": {
          "AWS:SourceOwner": "123456789012"
        }
      }
    },
    {
      "Sid": "Allow_Publish_Events",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sns:Publish",
      "Resource": "arn:aws:sns:us-east-2:123456789012:MyTopic"
    }
  ]
}
```

Weitere Informationen zu CodeCommit und CloudWatch Veranstaltungen finden Sie unter Beispiele für CloudWatch Ereignisse und Ereignisse von Supported Services. Weitere Informationen zu IAM und der Richtliniensprache finden Sie unter Grammatik der IAM-JSON-Richtliniensprache.

Beispiel 3: Erstellen Sie eine Richtlinie für die AWS Lambda Integration mit einem Trigger CodeCommit

Sie können ein CodeCommit Repository so konfigurieren, dass Code-Pushs oder andere Ereignisse Aktionen auslösen, z. B. das Aufrufen einer Funktion in. AWS Lambda Weitere Informationen finden Sie unter Erstellen Sie einen Trigger für eine Lambda-Funktion. Diese Informationen beziehen sich nur auf Trigger und nicht CloudWatch auf Ereignisse.

Wenn Sie möchten, dass Ihr Trigger eine Lambda-Funktion direkt ausführt (anstatt ein Amazon SNS SNS-Thema zum Aufrufen der Lambda-Funktion zu verwenden), und Sie den Trigger nicht in der Lambda-Konsole konfigurieren, müssen Sie eine Anweisung ähnlich der folgenden in die ressourcenbasierte Richtlinie der Funktion aufnehmen:

```
{
  "Statement":{
     "StatementId":"Id-1",
     "Action":"lambda:InvokeFunction",
```

```
      "Principal":"codecommit.amazonaws.com",
      "SourceArn":"arn:aws:codecommit:us-east-2:111111111111:MyDemoRepo",
      "SourceAccount":"111111111111"
    }
}
```

Wenn Sie einen CodeCommit Trigger manuell konfigurieren, der eine Lambda-Funktion aufruft, müssen Sie auch den AddPermissionLambda-Befehl verwenden, um die Erlaubnis zum Aufrufen der Funktion CodeCommit zu erteilen. Ein Beispiel finden Sie im Abschnitt Um die Ausführung einer Lambda-Funktion CodeCommit zu ermöglichen von Erstellen Sie einen Trigger für eine bestehende Lambda-Funktion.

Weitere Informationen zu Ressourcenrichtlinien für Lambda-Funktionen finden Sie unter AddPermissionund The Pull/Push Event Models im AWS Lambda Developer Guide.

Referenz für CodeCommit-Berechtigungen

In den folgenden Tabellen sind die einzelnen CodeCommit API-Operationen, die entsprechenden Aktionen, für die Sie Berechtigungen erteilen können, und das Format des Ressourcen-ARN aufgeführt, der für die Erteilung von Berechtigungen verwendet werden soll. Die CodeCommit APIs sind auf der Grundlage des Umfangs der Aktionen, die von dieser API zugelassen sind, in Tabellen gruppiert. Beziehen Sie sich darauf, wenn Sie Berechtigungsrichtlinien einrichten Zugriffskontrolle und schreiben, die Sie einer IAM-Identität zuordnen können (identitätsbasierte Richtlinien).

Beim Erstellen einer Berechtigungsrichtlinie geben Sie die Aktionen im Feld `Action` der Richtlinie an. Sie geben den Ressourcenwert im Feld `Resource` der Richtlinie als ARN mit oder ohne Platzhalterzeichen (*) an.

Verwenden AWS Sie Bedingungsschlüssel, um Bedingungen in Ihren CodeCommit Richtlinien auszudrücken. Eine vollständige Liste der AWS-weiten Schlüssel enthält der Abschnitt Verfügbare Schlüssel im IAM Benutzerhandbuch. Vollständige Informationen zu Aktionen, Ressourcen und Bedingungsschlüsseln für CodeCommit in IAM-Richtlinien finden Sie unter Aktionen, Ressourcen und Bedingungsschlüssel für. AWS CodeCommit

> **Note**
>
> Um eine Aktion anzugeben, verwenden Sie das Präfix `codecommit:` gefolgt vom Namen der API-Operation (z. B. `codecommit:GetRepository` oder `codecommit:CreateRepository`).

Verwenden von Platzhaltern

Sie können ein Platzhalterzeichen (*) in Ihrem ARN verwenden, um mehrere Aktionen oder Ressourcen anzugeben. `codecommit:*`Gibt beispielsweise alle Aktionen an und `codecommit:Get*` gibt alle CodeCommit CodeCommit Aktionen an, die mit dem Wort `Get` beginnen. Im folgenden Beispiel wird der Zugriff auf alle Repositorys erteilt, deren Name mit `MyDemo` beginnt:

```
arn:aws:codecommit:us-west-2:111111111111:MyDemo*
```

Sie können Platzhalterzeichen nur für die in der folgenden Tabelle aufgeführten *`repository-name`*-Ressourcen verwenden. Platzhalter können nicht für *`region`*- oder *`account-id`*-Ressourcen verwendet werden. Weitere Informationen zu Platzhaltern finden Sie unter IAM-Identifikatoren im IAM-Benutzerhandbuch.

Themen

- Erforderliche Berechtigungen für Git-Client-Befehle
- Berechtigungen für Aktionen in Branches
- Berechtigungen für Aktionen bei Zusammenführungen
- Berechtigungen für Aktionen bei Pull-Requests
- Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln
- Berechtigungen für Aktionen an einzelnen Dateien
- Berechtigungen für Aktionen im Zusammenhang mit Kommentaren
- Berechtigungen für Aktionen an festgeschriebenem Code
- Berechtigungen für Aktionen in Repositorys
- Berechtigungen für Aktionen mit Tags
- Berechtigungen für Aktionen auf Triggern
- Berechtigungen für Aktionen bei der CodePipeline Integration

Erforderliche Berechtigungen für Git-Client-Befehle

CodeCommitIn gelten die `GitPull` IAM-Richtlinienberechtigungen für jeden Git-Client-Befehl CodeCommit, von dem Daten abgerufen werden git fetchgit clone, einschließlich, usw. In ähnlicher

Weise gelten die `GitPush` IAM-Richtlinienberechtigungen für jeden Git-Client-Befehl, an den Daten gesendet CodeCommit werden. Wenn die `GitPush` IAM-Richtlinienberechtigung beispielsweise auf gesetzt ist`Allow`, kann ein Benutzer das Löschen eines Branches mithilfe des Git-Protokolls vorantreiben. Dieser Push wird nicht von den Berechtigungen beeinflusst, die für diesen IAM-Benutzer auf den `DeleteBranch` Vorgang angewendet wurden. Die `DeleteBranch`-Berechtigung bezieht sich auf Aktionen, die mit der Konsole, der AWS CLI den SDKs und der API ausgeführt werden, aber nicht mit dem Git-Protokoll.

`GitPull`und `GitPush` sind IAM-Richtlinienberechtigungen. Es handelt sich dabei nicht um API-Aktionen.

CodeCommit Erforderliche Berechtigungen für Aktionen für Git-Client-Befehle

GitPull

Aktion(en): `codecommit:GitPull`

Erforderlich, um Informationen aus einem CodeCommit Repository in ein lokales Repository abzurufen. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

GitPush

Aktion(en): `codecommit:Git Push`

Erforderlich, um Informationen von einem lokalen Repository in ein CodeCommit Repository zu übertragen. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

Berechtigungen für Aktionen in Branches

Die folgenden Berechtigungen erlauben oder verweigern Aktionen für Branches in CodeCommit Repositorys. Diese Berechtigungen beziehen sich nur auf Aktionen, die in der CodeCommit Konsole und mit der CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der ausgeführt werden. AWS CLI Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Beispielsweise zeigt der Befehl git show-branch -r eine Liste externer Branches für ein Repository und seine Commits unter Verwendung des Git-Protokolls an. Sie wird durch keine Berechtigungen für den CodeCommit `ListBranches` Vorgang beeinflusst.

Weitere Informationen zu Richtlinien für Filialen finden Sie unter Beschränken Sie die Anzahl von Pushs und Merges auf Branches in AWS CodeCommit undBeispiele für vom Kunden verwaltete Richtlinien.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen in Branches

CreateBranch

Aktion(en): `codecommit:CreateBranch`

Erforderlich, um einen Branch in einem CodeCommit Repository zu erstellen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

DeleteBranch

Aktion(en): `codecommit:DeleteBranch`

Erforderlich, um einen Branch aus einem CodeCommit Repository zu löschen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

GetBranch

Aktion(en): `codecommit:GetBranch`

Erforderlich, um Details zu einem Branch in einem CodeCommit Repository abzurufen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

ListBranches

Aktion(en): `codecommit:ListBranches`

Erforderlich, um eine Liste von Branches in einem CodeCommit Repository abzurufen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

MergeBranchesByFastForward

Aktion(en): `codecommit:MergeBranchesByFastForward`

Erforderlich, um zwei Branches mithilfe der Fast-Forward-Merge-Strategie in einem CodeCommit Repository zusammenzuführen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

MergeBranchesBySquash

Aktion(en): `codecommit:ListBranches`

Erforderlich zum Zusammenführen von zwei Branches mit der Squashmerge-Strategie in einem CodeCommit -Repository.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`

MergeBranchesByThreeWay

Aktion(en): `codecommit:ListBranches`

Erforderlich zum Zusammenführen von zwei Branches mithilfe der Dreiwegemerge-Strategie in einem CodeCommit -Repository.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`

UpdateDefaultBranch

Aktion(en): `codecommit:UpdateDefaultBranch`

Erforderlich, um den Standard-Branch in einem CodeCommit Repository zu ändern.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`

Berechtigungen für Aktionen bei Zusammenführungen

Die folgenden Berechtigungen erlauben oder verweigern Aktionen bei Zusammenführungen in CodeCommit Repositorys. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole und der CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der ausgeführt werden. AWS CLI Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Entsprechende Berechtigungen für Branches finden Sie unter Berechtigungen für Aktionen in Branches. Entsprechende Berechtigungen für Pull-Anforderungen finden Sie unter Berechtigungen für Aktionen bei Pull-Requests.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen für Merge-Befehle

BatchDescribeMergeConflicts

Aktion(en): `codecommit:BatchDescribeMergeConflicts`

Erforderlich, um Informationen über Konflikte bei einer Zusammenführung zwischen Commits in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

CreateUnreferencedMergeCommit

Aktion(en): `codecommit:CreateUnreferencedMergeCommit`

Erforderlich, um einen nicht referenzierten Commit zwischen zwei Branches oder Commits in einem CodeCommit Repository zu erstellen, um diese zu vergleichen und mögliche Konflikte zu identifizieren.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

DescribeMergeConflicts

Aktion(en): `codecommit:DescribeMergeConflicts`

Erforderlich zum Abruf von Informationen über Zusammenführungskonflikte zwischen den Basis-, Quell- und Ziel-Versionen einer Datei in einer potenziellen Zusammenführung in einem CodeCommit -Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetMergeCommit

Aktion(en): `codecommit:GetMergeCommit`

Erforderlich, um Informationen über die Zusammenführung zwischen einem Quell- und einem Ziel-Commit in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetMergeOptions

Aktion(en): `codecommit:GetMergeOptions`

Erforderlich, um Informationen über die verfügbaren Zusammenführungsoptionen zwischen zwei Branches oder Commit-Spezifizierern in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen bei Pull-Requests

Die folgenden Berechtigungen erlauben oder verweigern Aktionen für Pull-Anfragen in CodeCommit-Repositorys. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole

und der CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der AWS CLI ausgeführt werden. Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Weitere Berechtigungen für Kommentare finden Sie unter Berechtigungen für Aktionen im Zusammenhang mit Kommentaren.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen auf Pull-Requests

BatchGetPullRequests

Aktion(en): `codecommit:BatchGetPullRequests`

Erforderlich, um Informationen zu einer oder mehreren Pull-Anforderungen in einem CodeCommit -Repository zurückzugeben. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

CreatePullRequest

Aktion(en): `codecommit:CreatePullRequest`

Erforderlich, um eine Pull-Anfrage in einem CodeCommit Repository zu erstellen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

CreatePullRequestApprovalRule

Aktion(en): `codecommit:CreatePullRequestApprovalRule`

Erforderlich, um eine Genehmigungsregel für eine Pull-Anforderung in einem CodeCommit-Repository zu erstellen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

DeletePullRequestApprovalRule

Aktion(en): `codecommit:DeletePullRequestApprovalRule`

Erforderlich, um eine Genehmigungsregel für eine Pull-Anforderung in einem CodeCommit-Repository zu löschen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

DescribePullRequestEvents

Aktion(en): `codecommit:DescribePullRequestEvents`

Erforderlich, um Informationen über ein oder mehrere Pull-Request-Ereignisse in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

EvaluatePullRequestApprovalRules

Aktion(en): `codecommit:EvaluatePullRequestApprovalRules`

Erforderlich, um zu bewerten, ob eine Pull-Anfrage alle Bedingungen erfüllt hat, die in den zugehörigen Genehmigungsregeln in einem CodeCommit Repository angegeben sind.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetCommentsForPullRequest

Aktion(en): `codecommit:GetCommentsForPullRequest`

Erforderlich, um Kommentare in einer Pull-Anfrage zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetCommitsFromMergeBase

Aktion(en): `codecommit:GetCommitsFromMergeBase`

Erforderlich, um Informationen über den Unterschied zwischen Commits im Zusammenhang mit einer potenziellen Zusammenführung zurückzugeben. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetMergeConflicts

Aktion(en): `codecommit:GetMergeConflicts`

Erforderlich, um Informationen über Merge-Konflikte zwischen dem Quell- und dem Ziel-Branch in einer Pull-Anfrage zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetPullRequest

Aktion(en): `codecommit:GetPullRequest`

Erforderlich, um Informationen zu einer Pull-Anforderung in einem CodeCommit-Repository zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetPullRequestApprovalStates

Aktion(en): `codecommit:GetPullRequestApprovalStates`

Erforderlich, um Informationen zu den Genehmigungsstatus für eine angegebene Pull-Anforderung zurückzugeben.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetPullRequestOverrideState

Aktion(en): `codecommit:GetPullRequestOverrideState`

Erforderlich, um Informationen darüber zurückzugeben, ob Genehmigungsregeln für eine Pull-Anfrage reserviert (überschrieben) wurden, und falls ja, den Amazon-Ressourcennamen (ARN) des Benutzers oder der Identität, der die Regeln und deren Anforderungen für die Pull-Anfrage außer Kraft gesetzt hat.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

ListPullRequests

Aktion(en): `codecommit:ListPullRequests`

Erforderlich für das Auflisten von Pull-Anfragen in einem Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

MergePullRequestByFastForward

Aktion(en): `codecommit:MergePullRequestByFastForward`

Erforderlich für das Schließen einer Pull-Anforderung und die versuchte Zusammenführung des Quell-Branch mit dem Ziel-Branch einer Pull-Anforderung unter Verwendung der Mergestrategie mit Vorlauf.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

MergePullRequestBySquash

Aktion(en): `codecommit:MergePullRequestBySquash`

Erforderlich für das Schließen einer Pull-Anforderung und für die versuchte Zusammenführung des Quell-Branch mit dem Ziel-Branch einer Pull-Anforderung unter Verwendung der Squashmerge-Strategie.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

MergePullRequestByThreeWay

Aktion(en): `codecommit:MergePullRequestByThreeWay`

Erforderlich für das Schließen einer Pull-Anforderung und die versuchte Zusammenführung des Quell-Branch mit dem Ziel-Branch einer Pull-Anforderung unter Verwendung der „Three-Way“-Strategie.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

OverridePullRequestApprovalRules

Aktion(en): `codecommit:OverridePullRequestApprovalRules`

Erforderlich, um alle Anforderungen an Genehmigungsregeln für eine Pull-Anfrage in einem Repository aufzuheben. CodeCommit

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PostCommentForPullRequest

Aktion(en): `codecommit:PostCommentForPullRequest`

Erforderlich, um einen Kommentar zu einer Pull-Anforderung in einem CodeCommit-Repository zu veröffentlichen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdatePullRequestApprovalRuleContent

Aktion(en): `codecommit:UpdatePullRequestApprovalRuleContent`

Erforderlich, um die Struktur einer Genehmigungsregel für eine Pull-Anfrage in einem CodeCommit Repository zu ändern.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdatePullRequestApprovalState

Aktion(en): `codecommit:UpdatePullRequestApprovalState`

Erforderlich, um den Status einer Genehmigung für eine Pull-Anforderung in einem CodeCommit -Repository zu aktualisieren.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdatePullRequestDescription

Aktion(en): `codecommit:UpdatePullRequestDescription`

Erforderlich zum Ändern der Beschreibung einer Pull-Anfrage in einem CodeCommit-Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdatePullRequestStatus

Aktion(en): `codecommit:UpdatePullRequestStatus`

Erforderlich zum Ändern des Status einer Pull-Anfrage in einem CodeCommit-Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdatePullRequestTitle

Aktion(en): `codecommit:UpdatePullRequestTitle`

Erforderlich zum Ändern des Titels einer Pull-Anfrage in einem CodeCommit-Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln

Die folgenden Berechtigungen erlauben oder verweigern Aktionen mit Genehmigungsregelvorlagen in CodeCommit -Repositorys. Diese Berechtigungen beziehen sich nur auf Aktionen, die in der CodeCommit Konsole und der CodeCommit API ausgeführt werden, und auf Befehle, die mit der AWS CLI ausgeführt werden. Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Entsprechende Berechtigungen für Pull-Anforderungen finden Sie unter Berechtigungen für Aktionen bei Pull-Requests.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen auf Vorlagen für Genehmigungsregeln

AssociateApprovalRuleTemplateWithRepository

Aktion(en): `codecommit:AssociateApprovalRuleTemplateWithRepository`

Erforderlich, um eine Vorlage mit einem bestimmten Repository in einem Amazon Web Services Services-Konto zu verknüpfen. Nach der Zuordnung werden automatisch Genehmigungsregeln erstellt, die den Vorlagenbedingungen für jede im angegebenen Repository erstellte Pull-Anforderung entsprechen.

Ressource: *

BatchAssociateApprovalRuleTemplateWithRepositories

Aktion(en): `codecommit:BatchAssociateApprovalRuleTemplateWithRepositories`

Erforderlich, um eine Vorlage mit einem oder mehreren angegebenen Repositorys in einem Amazon Web Services Services-Konto zu verknüpfen.

Ressource: *

BatchDisassociateApprovalRuleTemplateFromRepositories

Aktion(en): `codecommit:BatchDisassociateApprovalRuleTemplateFromRepositories`

Erforderlich, um eine Vorlage von einem oder mehreren angegebenen Repositorys in einem Amazon Web Services Services-Konto zu trennen.

Ressource: *

CreateApprovalRuleTemplate

Aktion(en): `codecommit:CreateApprovalRuleTemplate`

Erforderlich, um eine Vorlage für Genehmigungsregeln zu erstellen, die dann mit einem oder mehreren Repositorys in Ihrem Amazon Web Services Services-Konto verknüpft werden kann.

Ressource: *

DeleteApprovalRuleTemplate

Aktion(en): `codecommit:DeleteApprovalRuleTemplate`

Erforderlich, um eine Genehmigungsregelvorlage aus einem AWS-Konto zu löschen.

Ressource: *

DisassociateApprovalRuleTemplateFromRepository

Aktion(en): `codecommit:DisassociateApprovalRuleTemplateFromRepository`

Erforderlich, um die angegebene Vorlage von einem Repository in einem Amazon Web Services Services-Konto zu trennen. Genehmigungsregeln für bereits mit der Vorlage erstellte Pull-Anforderungen werden nicht entfernt.

Ressource: *

GetApprovalRuleTemplate

Aktion(en): `codecommit:GetApprovalRuleTemplate`

Erforderlich, um Informationen zu einer Vorlage für Genehmigungsregeln in einem Amazon Web Services Services-Konto zurückzugeben.

Ressource: *

ListApprovalRuleTemplates

Aktion(en): `codecommit:ListApprovalRuleTemplates`

Erforderlich, um Vorlagen für Genehmigungsregeln in einem Amazon Web Services Services-Konto aufzulisten.

Ressource: *

ListAssociatedApprovalRuleTemplatesForRepository

Aktion(en): `codecommit:ListAssociatedApprovalRuleTemplatesForRepository`

Erforderlich, um alle Vorlagen für Genehmigungsregeln aufzulisten, die einem bestimmten Repository in einem Amazon Web Services Services-Konto zugeordnet sind.

Ressource: *

ListRepositoriesForApprovalRuleTemplate

Aktion(en): `codecommit:ListRepositoriesForApprovalRuleTemplate`

Erforderlich, um alle Repositorys aufzulisten, die mit einer bestimmten Genehmigungsregelvorlage in einem Amazon Web Services Services-Konto verknüpft sind.

Ressource: *

UpdateApprovalRuleTemplateContent

Aktion(en): `codecommit:UpdateApprovalRuleTemplateContent`

Erforderlich, um den Inhalt einer Vorlage für Genehmigungsregeln in einem Amazon Web Services Services-Konto zu aktualisieren.

Ressource: *

UpdateApprovalRuleTemplateDescription

Aktion(en): `codecommit:UpdateApprovalRuleTemplateDescription`

Erforderlich, um die Beschreibung einer Vorlage für Genehmigungsregeln in einem Amazon Web Services Services-Konto zu aktualisieren.

Ressource: *

UpdateApprovalRuleTemplateName

Aktion(en): `codecommit:UpdateApprovalRuleTemplateName`

Erforderlich, um den Namen einer Genehmigungsregelvorlage in einem AWS-Konto zu aktualisieren.

Ressource: *

Berechtigungen für Aktionen an einzelnen Dateien

Die folgenden Berechtigungen erlauben bzw. verweigern Aktionen auf einzelne Dateien in CodeCommit-Repositorys. Diese Berechtigungen beziehen sich nur auf Aktionen, die in der CodeCommit Konsole, der CodeCommit API ausgeführt werden, und auf Befehle, die mit der AWS CLI ausgeführt werden. Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Mit dem Befehl `git push` beispielsweise werden neue und geänderte Dateien im Push-Verfahren an ein CodeCommit-Repository mithilfe des Git-Protokolls übertragen. Sie wird durch keine Berechtigungen für den CodeCommit `PutFile` Vorgang beeinflusst.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen an einzelnen Dateien

DeleteFile

Aktion(en): `codecommit:DeleteFile`

Erforderlich, um eine bestimmte Datei aus einem bestimmten Zweig in einem CodeCommit Repository von der CodeCommit Konsole aus zu löschen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

GetBlob

Aktion(en): `codecommit:GetBlob`

Erforderlich, um den codierten Inhalt einer einzelnen Datei in einem CodeCommit Repository von der CodeCommit Konsole aus anzuzeigen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetFile

Aktion(en): `codecommit:GetFile`

Erforderlich, um den codierten Inhalt einer bestimmten Datei und ihrer Metadaten in einem CodeCommit Repository von der CodeCommit Konsole aus anzuzeigen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetFolder

Aktion(en): `codecommit:GetFolder`

Erforderlich, um den Inhalt eines bestimmten Ordners in einem CodeCommit Repository von der CodeCommit Konsole aus anzuzeigen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PutFile

Aktion(en): `codecommit:PutFile`

Erforderlich, um einem CodeCommit Repository über die CodeCommit Konsole, CodeCommit API oder die eine neue oder geänderte Datei hinzuzufügenAWS CLI.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen im Zusammenhang mit Kommentaren

Die folgenden Berechtigungen erlauben oder verbieten Aktionen für Kommentare in CodeCommit Repositorys. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole und der CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der ausgeführt werden. AWS CLI Weitere Berechtigungen für Kommentare in Pull-Anfragen finden Sie unter Berechtigungen für Aktionen bei Pull-Requests.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen in Repositorys

DeleteCommentContent

Aktion(en): `codecommit:DeleteCommentContent`

Erforderlich zum Löschen des Inhalts eines Kommentars an einer Änderung, eine Datei oder einem Commit in einem Repository. Kommentare können nicht gelöscht werden, aber der Inhalt eines Kommentars kann entfernt werden, wenn der Benutzer über die entsprechende Berechtigung verfügt.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

GetComment

Aktion(en): `codecommit:GetComment`

Erforderlich, um Informationen über einen Kommentar zu einer Änderung, Datei oder einem Commit in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

GetCommentReactions

Aktion(en): `codecommit:GetCommentReactions`

Erforderlich, um Informationen über Emoji-Reaktionen auf einen Kommentar zu einer Änderung, Datei oder einem Commit in einem CodeCommit Repository zurückzugeben.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

GetCommentsForComparedCommit

Aktion(en): `codecommit:GetCommentsForComparedCommit`

Erforderlich, um Informationen über Kommentare zurückzugeben, die beim Vergleich zwischen zwei Commits in einem CodeCommit Repository abgegeben wurden.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

PostCommentForComparedCommit

Aktion(en): `codecommit:PostCommentForComparedCommit`

Erforderlich zum Erstellen eines Kommentars zum Vergleich zwischen zwei Commits in einem CodeCommit-Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PostCommentReply

Aktion(en): `codecommit:PostCommentReply`

Erforderlich, um eine Antwort auf einen Kommentar zu einem Vergleich zwischen Commits oder zu einer Pull-Anfrage in einem CodeCommit Repository zu erstellen.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PutCommentReaction

Aktion(en): `codecommit:PutCommentReaction`

Erforderlich, um auf einen Kommentar mit einem Emoji zu einem Commit oder einer Pull-Anfrage in einem CodeCommit Repository zu antworten.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UpdateComment

Aktion(en): `codecommit:UpdateComment`

Erforderlich zum Bearbeiten eines Kommentars zu einem Vergleich zwischen Commits oder zu einer Pull-Anfrage. Kommentare können nur von ihrem Autor bearbeitet werden.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen an festgeschriebenem Code

Die folgenden Berechtigungen erlauben oder verweigern Aktionen für festgeschriebenen Code CodeCommit-Repositorys. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole und der CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der AWS CLI ausgeführt werden. Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können. Beispiel: Der Befehl git commit erstellt einen Commit für einen Branch in einem Repository mithilfe des Git-Protokolls. Es wird durch keine Berechtigungen für den CodeCommit `CreateCommit` Vorgang beeinflusst.

Die ausdrückliche Verweigerung einiger dieser Berechtigungen kann zu unerwarteten Konsequenzen in der CodeCommit Konsole führen. Wird beispielsweise `GetTree` auf `Deny` gesetzt, wird verhindert, dass Benutzer durch die Inhalte eines Repositorys in der Konsole navigieren. Benutzer werden jedoch nicht daran gehindert, die Inhalte einer Datei im Repository anzuzeigen (wenn sie

beispielsweise einen Link zu der Datei per E-Mail gesendet bekommen). Durch das Festlegen von `GetBlob` auf `Deny` wird verhindert, dass Benutzer die Inhalte von Dateien anzeigen. Benutzer werden jedoch nicht daran gehindert, die Struktur eines Repositorys zu durchsuchen. Durch das Festlegen von `GetCommit` auf `Deny` wird verhindert, dass Benutzer Details über Commits abrufen. Durch das Festlegen von `GetObjectIdentifier` auf `Deny` wird der Großteil der Funktionalität von Code-Browsing blockiert. Wenn Sie alle drei Aktionen `Deny` in einer Richtlinie auf festlegen, kann ein Benutzer mit dieser Richtlinie keinen Code in der CodeCommit Konsole durchsuchen.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen mit festgeschriebenem Code

BatchGetCommits

Aktion(en): `codecommit:BatchGetCommits`

Erforderlich, um Informationen zu einem oder mehreren Commits in einem CodeCommit-Repository zurückzugeben. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`

CreateCommit

Aktion(en): `codecommit:CreateCommit`

Erforderlich zum Erstellen eines Commits.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`

GetCommit

Aktion(en): `codecommit:GetCommit`

Erforderlich zum Abrufen von Informationen zu einem Commit.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`

GetCommitHistory

Aktion(en): `codecommit:GetCommitHistory`

Erforderlich zum Abfragen von Informationen über den Verlauf von Commits in einem Repository. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetDifferences

Aktion(en): `codecommit:GetDifferences`

Erforderlich zum Abfragen von Informationen über die Unterschiede in einem Commit-Spezifizierer (wie etwa ein Branch, ein Tag, HEAD, eine Commit-ID oder andere vollständig qualifizierte Referenzen).

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetObjectIdentifier

Aktion(en): `codecommit:GetObjectIdentifier`

Erforderlich zum Auflösen von Blobs, Strukturen und Commits zu ihrem Bezeichner. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetReferences

Aktion(en): `codecommit:GetReferences`

Erforderlich für die Rückgabe aller Referenzen, wie etwa Branches und Tags. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

GetTree

Aktion(en): `codecommit:GetTree`

Erforderlich, um den Inhalt einer bestimmten Struktur in einem CodeCommit Repository von der CodeCommit Konsole aus anzuzeigen. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen in Repositorys

Die folgenden Berechtigungen erlauben oder verweigern Aktionen in CodeCommit Repositorys. Diese Berechtigungen beziehen sich auf Aktionen, die mit der CodeCommit Konsole und der

CodeCommit API ausgeführt werden, sowie auf Befehle, die mit der ausgeführt werden. AWS CLI Sie gelten nicht für ähnliche Aktionen, die mit dem Git-Protokoll ausgeführt werden können.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen in Repositorys

BatchGetRepositories

Aktion(en): `codecommit:BatchGetRepositories`

Erforderlich, um Informationen über mehrere CodeCommit Repositorys abzurufen, die sich in einem Amazon Web Services Services-Konto befinden. In `Resource` müssen Sie die Namen aller CodeCommit Repositorys angeben, für die einem Benutzer Informationen erlaubt (oder verweigert) werden.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

CreateRepository

Aktion(en): `codecommit:CreateRepository`

Erforderlich, um ein CodeCommit Repository zu erstellen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

DeleteRepository

Aktion(en): `codecommit:DeleteRepository`

Erforderlich, um ein CodeCommit Repository zu löschen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

GetRepository

Aktion(en): `codecommit:GetRepository`

Erforderlich, um Informationen über ein einzelnes CodeCommit Repository abzurufen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

ListRepositories

Aktion(en): `codecommit:ListRepositories`

Erforderlich, um eine Liste der Namen und System-IDs mehrerer CodeCommit Repositorys für ein Amazon Web Services Services-Konto abzurufen. Der einzige erlaubte Wert für `Resource` für diese Aktion bezieht sich auf alle Repositorys (*).

Ressource: *

UpdateRepositoryDescription

Aktion(en): `codecommit:UpdateRepositoryDescription`

Erforderlich, um die Beschreibung eines CodeCommit Repositorys zu ändern.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

UpdateRepositoryName

Aktion(en): `codecommit:UpdateRepositoryName`

Erforderlich, um den Namen eines CodeCommit Repositorys zu ändern. In `Resource` müssen Sie sowohl die CodeCommit Repositorys angeben, die geändert werden dürfen, als auch die neuen Repository-Namen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

Berechtigungen für Aktionen mit Tags

Die folgenden Berechtigungen erlauben oder verweigern Aktionen mit AWS Tags für CodeCommit Ressourcen.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen mit Tags

ListTagsForResource

Aktion(en): `codecommit:ListTagsForResource`

Erforderlich zum Abfragen von Informationen über AWS-Tags auf einer Ressource in CodeCommit.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

TagResource

Aktion(en): `codecommit:TagResource`

Erforderlich zum Hinzufügen oder Bearbeiten von AWS-Tags für ein Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

UntagResource

Aktion(en): `codecommit:UntagResource`

Erforderlich, um AWS Tags aus einer Ressource in zu entfernen CodeCommit.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen auf Triggern

Die folgenden Berechtigungen genehmigen oder verweigern Aktionen für Auslöser von CodeCommit-Repositorys.

CodeCommit API-Operationen und erforderliche Berechtigungen für Aktionen auf Triggern

GetRepositoryTriggers

Aktion(en): `codecommit:GetRepositoryTriggers`

Erforderlich für die Rückgabe von Informationen über für ein Repository konfigurierte Auslöser.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

PutRepositoryTriggers

Aktion(en): `codecommit:PutRepositoryTriggers`

Erforderlich zum Erstellen, Bearbeiten oder Löschen von Auslösern für ein Repository.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

TestRepositoryTriggers

Aktion(en): `codecommit:TestRepositoryTriggers`

Erforderlich zum Testen der Funktionalität eines Repository-Auslösers, indem Daten an das Thema oder die Funktion gesendet werden, das bzw. die für den Auslöser konfiguriert wurde.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Berechtigungen für Aktionen bei der CodePipeline Integration

Um ein CodeCommit Repository in einer Quellaktion für eine Pipeline verwenden CodePipeline zu können, müssen Sie der Servicerolle für alle in der folgenden Tabelle aufgeführten Berechtigungen gewähren CodePipeline. Wenn diese Berechtigungen nicht in der Servicerolle angegeben oder auf **`Deny`** festgelegt sind, wird die Pipeline nicht automatisch ausgeführt, wenn eine Änderung an dem Repository vorgenommen wird, und Änderungen können nicht manuell veröffentlicht werden.

CodeCommit API-Operationen und erforderliche Berechtigungen für Integrationsaktionen CodePipeline

GetBranch

Aktion(en): `codecommit:GetBranch`

Erforderlich, um Details zu einem Branch in einem CodeCommit Repository abzurufen.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

GetCommit

Aktion(en): `codecommit:GetCommit`

Erforderlich zum Abrufen von Informationen zu einem Commit.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

UploadArchive

Aktion(en): `codecommit:UploadArchive`

Erforderlich, damit die Servicerolle für CodePipeline Repository-Änderungen in eine Pipeline hochladen kann. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:`*`region`*`:`*`account-id`*`:`*`repository-name`*

GetUploadArchiveStatus

Aktion(en): `codecommit:GetUploadArchiveStatus`

Erforderlich zum Bestimmen des Status des Hochladens des Archivs, etwa ob es sich in Bearbeitung befindet, abgeschlossen ist, abgebrochen wurde oder ob ein Fehler aufgetreten ist. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

CancelUploadArchive

Aktion(en): `codecommit:CancelUploadArchive`

Erforderlich zum Abbrechen des Uploads eines Archivs in eine Pipeline. Dies ist nur eine IAM-Richtlinienberechtigung, keine API-Aktion, die Sie aufrufen können.

Ressource: `arn:aws:codecommit:region:account-id:repository-name`

Featuresweise von AWS CodeCommit mit IAM

Bevor Sie IAM verwenden, um den Zugriff auf zu verwalten CodeCommit, sollten Sie wissen, welche IAM-Funktionen zur Verfügung stehen. CodeCommit Einen Überblick über das Zusammenwirken von CodeCommit und anderen AWS-Services mit IAM finden Sie unter AWS-Services, die mit IAM funktionieren im IAM-Benutzerhandbuch.

Themen

- Bedingungsschlüssel
- Beispiele

Bedingungsschlüssel

Administratoren können mithilfe von AWS-JSON-Richtlinien festlegen, wer zum Zugriff auf was berechtigt ist. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Condition` (oder `Condition` block) ermöglicht Ihnen die Angabe der Bedingungen, unter denen eine Anweisung wirksam ist. Das Element `Condition` ist optional. Sie können bedingte Ausdrücke erstellen, die Bedingungsoperatoren verwenden, z. B. ist gleich oder kleiner als, damit die Bedingung in der Richtlinie mit Werten in der Anforderung übereinstimmt.

Wenn Sie mehrere `Condition`-Elemente in einer Anweisung oder mehrere Schlüssel in einem einzelnen `Condition`-Element angeben, wertet AWS diese mittels einer logischen `AND`-Operation aus. Wenn Sie mehrere Werte für einen einzelnen Bedingungsschlüssel angeben, wertet AWS die Bedingung mittels einer logischen `OR`-Operation aus. Alle Bedingungen müssen erfüllt werden, bevor die Berechtigungen der Anweisung gewährt werden.

Sie können auch Platzhaltervariablen verwenden, wenn Sie Bedingungen angeben. Beispielsweise können Sie einem IAM-Benutzer die Berechtigung für den Zugriff auf eine Ressource nur dann gewähren, wenn sie mit dessen IAM-Benutzernamen gekennzeichnet ist. Weitere Informationen finden Sie unter IAM-Richtlinienelemente: Variablen und Tags im IAM-Benutzerhandbuch.

AWS unterstützt globale Bedingungsschlüssel und servicespezifische Bedingungsschlüssel. Eine Liste aller globalen AWS-Bedingungsschlüssel finden Sie unter Globale AWS-Bedingungskontextschlüssel im IAM-Benutzerhandbuch.

CodeCommit definiert seinen eigenen Satz von Bedingungsschlüsseln und unterstützt auch die Verwendung einiger globaler Bedingungsschlüssel. Eine Liste aller globalen AWS-Bedingungsschlüssel finden Sie unter Globale AWS-Bedingungskontextschlüssel im IAM-Benutzerhandbuch.

Einige CodeCommit Aktionen unterstützen den `codecommit:References` Bedingungsschlüssel. Eine Beispielrichtlinie, die diesen Schlüssel verwendet, finden Sie unter Beispiel 4: Aktionen für Branches verweigern oder zulassen.

Eine Liste der CodeCommit Bedingungsschlüssel finden Sie unter Bedingungsschlüssel für AWS CodeCommit im IAM-Benutzerhandbuch. Informationen dazu, mit welchen Aktionen und Ressourcen Sie einen Bedingungsschlüssel verwenden können, finden Sie unter Von AWS CodeCommit definierte Aktionen.

Beispiele

Beispiele für CodeCommit identitätsbasierte Richtlinien finden Sie unter. AWS CodeCommitBeispiele für identitätsbasierte -Richtlinien

Ressourcenbasierte CodeCommit-Richtlinien

CodeCommit unterstützt keine ressourcenbasierten Richtlinien.

Autorisierung auf der Grundlage von Tags CodeCommit

Sie können Tags an CodeCommit Ressourcen anhängen oder Tags in einer Anfrage an übergeben CodeCommit. Um den Zugriff auf der Grundlage von Tags zu steuern, geben Sie im Bedingungselement einer Richtlinie Tag-Informationen an, indem Sie die Schlüssel `codecommit:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`*, oder `aws:TagKeys`Bedingung verwenden. Weitere Informationen zum Markieren von CodeCommit

Ressourcen finden Sie unterBeispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen. Weitere Informationen zu Markierungsstrategien finden Sie unter Markieren von AWS-Ressourcen.

CodeCommit unterstützt auch Richtlinien, die auf Sitzungs-Tags basieren. Weitere Informationen finden Sie unter Sitzungs-Tags

Verwendung von Tags zur Bereitstellung von Identitätsinformationen in CodeCommit

CodeCommit unterstützt die Verwendung von Sitzungs-Tags, bei denen es sich um Schlüssel-Wert-Paarattribute handelt, die Sie übergeben, wenn Sie eine IAM-Rolle übernehmen, temporäre Anmeldeinformationen verwenden oder einen Benutzer in () verbinden. AWS Security Token Service AWS STS Sie können Tags auch einem IAM-Benutzer zuordnen. Sie können die in diesen Tags enthaltenen Informationen verwenden, um leichter zu ermitteln, wer eine Änderung vorgenommen oder ein Ereignis verursacht hat. CodeCommit schließt die Werte für Tags mit den folgenden Schlüsselnamen in CodeCommit Ereignisse ein:

Tastenname	Wert
`displayName`	Der für den Benutzer lesbare Name, der angezeigt und mit dem Benutzer verknüpft werden soll (z. B. Mary Major oder Saanvi Sarkar).
`emailAddress`	Die E-Mail-Adresse, die für den Benutzer angezeigt und mit ihm verknüpft werden soll (z. B. mary_major@example.com oder saanvi_sarkar@example.com).

Wenn diese Informationen bereitgestellt werden, werden sie CodeCommit in Veranstaltungen aufgenommen, die an Amazon EventBridge und Amazon CloudWatch Events gesendet werden. Weitere Informationen finden Sie unter Überwachung CodeCommit Ereignisse bei Amazon EventBridge und Amazon CloudWatch Events.

Um die Sitzungsmarkierung zu verwenden, müssen Rollen Richtlinien enthalten, bei denen die `sts:TagSession`-Berechtigung auf `Allow` gesetzt ist. Wenn Sie den Verbundzugriff verwenden, können Sie den Anzeigenamen und die E-Mail-Tag-Informationen im Rahmen Ihrer Einrichtung konfigurieren. Wenn Sie beispielsweise Azure Active Directory verwenden, können Sie die folgenden Antragsinformationen angeben:

Name des Antrags	Wert
`https://aws.amazon.com/SAML/Attributes/PrincipalTag:displayNam e`	`user.displayname`
`https://aws.amazon.com/SAML/Attributes/PrincipalTag:emailAddress`	`user.mail`

Sie können die AWS CLI verwenden, um Sitzungs-Tags für `displayName` und `emailAddress` mithilfe von AssumeRole zu übergeben. Beispielsweise könnte ein Benutzer, der eine Rolle namens *`Developer`* übernehmen möchte und seinem Namen *`Mary Major`* zuordnen möchte, den Befehl assume-role verwenden, so wie im Folgenden:

```
aws sts assume-role \
--role-arn arn:aws:iam::123456789012:role/Developer \
--role-session-name Mary-Major \
--tags Key=displayName,Value="Mary Major"
 Key=emailAddress,Value="mary_major@example.com" \
--external-id Example987
```

Weitere Informationen finden Sie unter AssumeRole.

Sie können den Vorgang `AssumeRoleWithSAML` verwenden, um einen Satz temporärer Anmeldeinformationen zurückzugeben, die die Tags `emailAddress` und `displayName` enthalten. Sie können diese Tags verwenden, wenn Sie auf CodeCommit-Repositorys zugreifen. Dies erfordert, dass Ihr Unternehmen oder Ihre Gruppe bereits Ihre SAML-Lösung eines Drittanbieters in AWS integriert hat. Wenn dies der Fall ist, können Sie SAML-Attribute als Sitzungs-Tags übergeben. Wenn Sie beispielsweise Identitätsattribute für einen Anzeigenamen und eine E-Mail-Adresse für einen Benutzer namens *`Saanvi Sarkar`* als Sitzungs-Tags übergeben möchten:

```
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:displayName">
  <AttributeValue>Saanvi Sarkar</AttributeValue>
</Attribute>
<Attribute Name="https://aws.amazon.com/SAML/Attributes/PrincipalTag:emailAddress">
  <AttributeValue>saanvi_sarkar@example.com</AttributeValue>
</Attribute>
```

Weitere Informationen finden Sie unter Übergeben von Sitzungs-Tags mithilfe von AssumeRoleWith SAML.

Sie können den Vorgang `AssumeRoleWithIdentity` verwenden, um einen Satz temporärer Anmeldeinformationen zurückzugeben, die die Tags `emailAddress` und `displayName` enthalten. Sie können diese Tags verwenden, wenn Sie auf CodeCommit-Repositorys zugreifen. Um Sitzungs-Tags von OpenID Connect (OIDC) zu übergeben, müssen Sie die Sitzungs-Tags in das JSON Web Token (JWT) einbeziehen. Beispiel: Das dekodierte JWP-Token, das zum Aufrufen von `AssumeRoleWithWebIdentity` verwendet wird, das die Sitzungs-Tags`displayName` und `emailAddress` für einen Benutzer namens *`Li Juan`*enthält:

```
{
    "sub": "lijuan",
    "aud": "ac_oic_client",
    "jti": "ZYUCeREXAMPLE",
    "iss": "https://xyz.com",
    "iat": 1566583294,
    "exp": 1566583354,
    "auth_time": 1566583292,
    "https://aws.amazon.com/tags": {
        "principal_tags": {
            "displayName": ["Li Juan"],
            "emailAddress": ["li_juan@example.com"],
        },
        "transitive_tag_keys": [
            "displayName",
            "emailAddress"
        ]
    }
}
```

Weitere Informationen finden Sie unter Übergeben von Sitzungs-Tags mithilfe von AssumeRoleWithWebIdentity.

Sie können den Vorgang `GetFederationToken` verwenden, um einen Satz temporärer Anmeldeinformationen zurückzugeben, die die Tags `emailAddress` und `displayName` enthalten. Sie können diese Tags verwenden, wenn Sie auf CodeCommit-Repositorys zugreifen. Um beispielsweise mit AWS CLI ein Verbundtoken abrufen zu können, das die Tags `emailAddress` und `displayName` enthält:

```
aws sts get-federation-token \
```

```
--name my-federated-user \
--tags key=displayName,value="Nikhil Jayashankar"
 key=emailAddress,value=nikhil_jayashankar@example.com
```

Weitere Informationen finden Sie unter Übergeben von Sitzungs-Tags mithilfe von GetFederationToken.

CodeCommit IAM-Rollen

Eine IAM-Rolle ist eine Entität in Ihrem Amazon–Web-Services-Konto mit spezifischen Berechtigungen.

Verwenden temporärer Anmeldeinformationen mit CodeCommit

Sie können temporäre Anmeldeinformationen verwenden, um sich über einen Verbund anzumelden, eine IAM-Rolle anzunehmen oder eine kontenübergreifende Rolle anzunehmen. Sie erhalten temporäre Sicherheitsanmeldedaten, indem Sie AWS STS API-Operationen wie AssumeRoleoder aufrufen GetFederationToken.

CodeCommit unterstützt die Verwendung temporärer Anmeldeinformationen. Weitere Informationen finden Sie unter Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen.

Serviceverknüpfte Rollen

Serviceverknüpfte Rollen erlauben AWS-Services den Zugriff auf Ressourcen in anderen Services, um eine Aktion in Ihrem Auftrag auszuführen. Serviceverknüpfte Rollen werden in Ihrem IAM-Konto angezeigt und gehören zum Service. Ein IAM-Administrator kann die Berechtigungen für serviceverknüpfte Rollen anzeigen, aber nicht bearbeiten.

CodeCommit verwendet keine dienstbezogenen Rollen.

Servicerollen

Dieses Feature ermöglicht einem Service das Annehmen einer Servicerolle in Ihrem Namen. Diese Rolle gewährt dem Service Zugriff auf Ressourcen in anderen Diensten, um eine Aktion in Ihrem Namen auszuführen. Servicerollen werden in Ihrem IAM-Konto angezeigt und gehören zum Konto. Dies bedeutet, dass ein IAM-Administrator die Berechtigungen für diese Rolle ändern kann. Dies kann jedoch die Funktionalität des Dienstes beeinträchtigen.

CodeCommit verwendet keine Servicerollen.

AWS CodeCommitBeispiele für identitätsbasierte -Richtlinien

IAM-Benutzer besitzen keine Berechtigungen zum Erstellen oder Ändern von CodeCommit -Ressourcen. Sie können auch keine Aufgaben ausführen, die die AWS Management Console-, AWS CLI- oder AWS-API benutzen. Ein IAM-Administrator muss IAM-Richtlinien erstellen, die Benutzern und Rollen die Berechtigung zum Ausführen bestimmter API-Operationen für die angegebenen Ressourcen gewähren, die diese benötigen. Der Administrator muss diese Richtlinien anschließend den IAM-Benutzern oder -Gruppen anfügen, die diese Berechtigungen benötigen.

Beispiele für diese Richtlinien finden Sie unter:

- Beispiel 1: Erlauben Sie einem Benutzer, CodeCommit Operationen in einem einzigen System auszuführen AWS-Region
- Beispiel 2: Erlaube einem Benutzer, Git für ein einzelnes Repository zu verwenden
- Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine Verbindung herstellt, Zugriff auf ein Repository
- Beispiel 4: Aktionen für Branches verweigern oder zulassen
- Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen
- Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen

Informationen dazu, wie Sie unter Verwendung dieser beispielhaften JSON-Richtliniendokumente eine identitätsbasierte IAM-Richtlinie erstellen, finden Sie unter Erstellen von Richtlinien auf der JSON-Registerkarte im IAM-Benutzerhandbuch.

Themen

- Bewährte Methoden für Richtlinien
- Mithilfe der Konsole CodeCommit
- Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer
- CodeCommit Repositorys auf der Grundlage von Tags anzeigen

Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand CodeCommit Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder sie löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen

AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Erste Schritte mit AWS-verwaltete Richtlinien und Umstellung auf Berechtigungen mit den geringsten Berechtigungen – Um Ihren Benutzern und Workloads Berechtigungen zu gewähren, verwenden Sie die AWS-verwaltete Richtlinien die Berechtigungen für viele allgemeine Anwendungsfälle gewähren. Sie sind in Ihrem AWS-Konto verfügbar. Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie AWS-kundenverwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter AWS-verwaltete Richtlinien oder AWS-verwaltete Richtlinien für Auftragsfunktionen im IAM-Benutzerhandbuch.
- Anwendung von Berechtigungen mit den geringsten Rechten – Wenn Sie mit IAM-Richtlinien Berechtigungen festlegen, gewähren Sie nur die Berechtigungen, die für die Durchführung einer Aufgabe erforderlich sind. Sie tun dies, indem Sie die Aktionen definieren, die für bestimmte Ressourcen unter bestimmten Bedingungen durchgeführt werden können, auch bekannt als die geringsten Berechtigungen. Weitere Informationen zur Verwendung von IAM zum Anwenden von Berechtigungen finden Sie unter Richtlinien und Berechtigungen in IAM im IAM-Benutzerhandbuch.
- Verwenden von Bedingungen in IAM-Richtlinien zur weiteren Einschränkung des Zugriffs – Sie können Ihren Richtlinien eine Bedingung hinzufügen, um den Zugriff auf Aktionen und Ressourcen zu beschränken. Sie können beispielsweise eine Richtlinienbedingung schreiben, um festzulegen, dass alle Anforderungen mithilfe von SSL gesendet werden müssen. Sie können auch Bedingungen verwenden, um Zugriff auf Service-Aktionen zu gewähren, wenn diese durch ein bestimmtes AWS-Service, wie beispielsweise AWS CloudFormation, verwendet werden. Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.
- Verwenden von IAM Access Analyzer zur Validierung Ihrer IAM-Richtlinien, um sichere und funktionale Berechtigungen zu gewährleisten – IAM Access Analyzer validiert neue und vorhandene Richtlinien, damit die Richtlinien der IAM-Richtliniensprache (JSON) und den bewährten IAM-Methoden entsprechen. IAM Access Analyzer stellt mehr als 100 Richtlinienprüfungen und umsetzbare Empfehlungen zur Verfügung, damit Sie sichere und funktionale Richtlinien erstellen können. Weitere Informationen finden Sie unter Richtlinienvalidierung zum IAM Access Analyzer im IAM-Benutzerhandbuch.
- Bedarf einer Multi-Faktor-Authentifizierung (MFA) – Wenn Sie ein Szenario haben, das IAM-Benutzer oder Root-Benutzer in Ihrem AWS-Konto erfordert, aktivieren Sie MFA für zusätzliche Sicherheit. Um MFA beim Aufrufen von API-Vorgängen anzufordern, fügen Sie Ihren Richtlinien

MFA-Bedingungen hinzu. Weitere Informationen finden Sie unter Konfigurieren eines MFA-geschützten API-Zugriffs im IAM-Benutzerhandbuch.

Weitere Informationen zu bewährten Methoden in IAM finden Sie unter Bewährte Methoden für die Sicherheit in IAM im IAM-Benutzerhandbuch.

Mithilfe der Konsole CodeCommit

Um auf die AWS CodeCommit-Konsole zuzugreifen, müssen Sie über einen Mindestsatz von Berechtigungen verfügen. Diese Berechtigungen müssen es Ihnen ermöglichen, Informationen zu den CodeCommit Ressourcen in Ihrem Amazon Web Services Services-Konto aufzulisten und einzusehen. Wenn Sie eine identitätsbasierte Richtlinie erstellen, die strenger ist als die mindestens erforderlichen Berechtigungen, funktioniert die Konsole nicht wie vorgesehen für Entitäten (IAM-Benutzer oder -Rollen) mit dieser Richtlinie.

Um sicherzustellen, dass diese Entitäten die CodeCommit Konsole weiterhin verwenden können, fügen Sie den Entitäten außerdem die folgende AWS verwaltete Richtlinie hinzu. Weitere Informationen finden Sie unter Hinzufügen von Berechtigungen zu einem Benutzer im IAM-Benutzerhandbuch:

Weitere Informationen finden Sie unter Verwendung identitätsbasierter Richtlinien (IAM-Richtlinien) für CodeCommit.

Für Benutzer, die nur Aufrufe an die AWS CLI oder AWS-API durchführen, müssen Sie keine Mindestberechtigungen in der Konsole erteilen. Stattdessen sollten Sie nur Zugriff auf die Aktionen zulassen, die den API-Operation entsprechen, die Sie ausführen möchten.

Gewähren der Berechtigung zur Anzeige der eigenen Berechtigungen für Benutzer

In diesem Beispiel wird gezeigt, wie Sie eine Richtlinie erstellen, die IAM-Benutzern die Berechtigung zum Anzeigen der eingebundenen Richtlinien und verwalteten Richtlinien gewährt, die ihrer Benutzeridentität angefügt sind. Diese Richtlinie enthält Berechtigungen für die Ausführung dieser Aktion auf der Konsole oder für die programmgesteuerte Ausführung über die AWS CLI oder die AWS-API.

```
{
    "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": "*"
        }
    ]
}
```

CodeCommit ***Repositorys*** auf der Grundlage von Tags anzeigen

Sie können in Ihrer identitätsbasierten Richtlinie Bedingungen für die Steuerung des Zugriffs auf CodeCommit -Ressourcen auf der Basis von Tags verwenden. Eine Beispielrichtlinie, die die Vorgehensweise veranschaulicht, finden Sie unter Beispiel 5: Aktionen für Repositorys mit Tags verweigern oder zulassen.

Weitere Informationen finden Sie unter IAM-JSON-Richtlinienelemente: Bedingung im IAM-Benutzerhandbuch.

Fehlerbehebung für AWS CodeCommit-Identität und -Zugriff

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit CodeCommit und IAM auftreten können.

Themen

- Ich bin nicht berechtigt, eine Aktion durchzuführen in CodeCommit
- Ich bin nicht berechtigt, iam auszuführen: PassRole
- Ich möchte meine Zugriffsschlüssel anzeigen
- Ich bin Administrator und möchte anderen Zugriff gewähren CodeCommit
- Ich möchte Personen außerhalb meines Amazon Web Services Services-Kontos den Zugriff auf meine CodeCommit Ressourcen ermöglichen

Ich bin nicht berechtigt, eine Aktion durchzuführen in CodeCommit

Wenn die AWS Management Console Ihnen mitteilt, dass Sie nicht zur Ausführung einer Aktion autorisiert sind, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.

Weitere Informationen finden Sie unter Erforderliche Berechtigungen für die Verwendung der CodeCommit-Konsole

Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie die Fehlermeldung erhalten, dass Sie nicht zum Durchführen der `iam:PassRole`-Aktion autorisiert sind, müssen Ihre Richtlinien aktualisiert werden, um eine Rolle an CodeCommit übergeben zu können.

Einige AWS-Services erlauben die Übergabe einer vorhandenen Rolle an diesen Dienst, sodass keine neue Servicerolle oder serviceverknüpfte Rolle erstellt werden muss. Hierzu benötigen Sie Berechtigungen für die Übergabe der Rolle an den Dienst.

Der folgende Beispielfehler tritt auf, wenn ein IAM-Benutzer mit dem Namen `marymajor` versucht, die Konsole zu verwenden, um eine Aktion in CodeCommit auszuführen. Die Aktion erfordert jedoch, dass der Service über Berechtigungen verfügt, die durch eine Servicerolle gewährt werden. Mary besitzt keine Berechtigungen für die Übergabe der Rolle an den Dienst.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In diesem Fall müssen die Richtlinien von Mary aktualisiert werden, um die Aktion `iam:PassRole` ausführen zu können.

Wenden Sie sich an Ihren AWS-Administrator, falls Sie weitere Unterstützung benötigen. Ihr Administrator hat Ihnen Ihre Anmeldeinformationen odzur Verfügung gestellt.

Ich möchte meine Zugriffsschlüssel anzeigen

Nachdem Sie Ihre IAM-Benutzerzugriffsschlüssel erstellt haben, können Sie Ihre Zugriffsschlüssel-ID jederzeit anzeigen. Sie können Ihren geheimen Zugriffsschlüssel jedoch nicht erneut anzeigen. Wenn Sie den geheimen Zugriffsschlüssel verlieren, müssen Sie ein neues Zugriffsschlüsselpaar erstellen.

Zugriffsschlüssel bestehen aus zwei Teilen: einer Zugriffsschlüssel-ID (z. B. `AKIAIOSFODNN7EXAMPLE`) und einem geheimen Zugriffsschlüssel (z. B. `wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY`). Ähnlich wie bei Benutzernamen und Passwörtern müssen Sie die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel zusammen verwenden, um Ihre Anforderungen zu authentifizieren. Verwalten Sie Ihre Zugriffsschlüssel so sicher wie Ihren Benutzernamen und Ihr Passwort.

> **Important**
>
> Geben Sie Ihre Zugriffsschlüssel nicht an Dritte weiter, auch nicht für die Suche nach Ihrer kanonischen Benutzer-ID. Wenn Sie dies tun, gewähren Sie anderen Personen möglicherweise den permanenten Zugriff auf Ihr AWS-Konto.

Während der Erstellung eines Zugriffsschlüsselpaars werden Sie aufgefordert, die Zugriffsschlüssel-ID und den geheimen Zugriffsschlüssel an einem sicheren Speicherort zu speichern. Der geheime Zugriffsschlüssel ist nur zu dem Zeitpunkt verfügbar, an dem Sie ihn erstellen. Wenn Sie Ihren geheimen Zugriffsschlüssel verlieren, müssen Sie Ihrem IAM-Benutzer neue Zugriffsschlüssel hinzufügen. Sie können maximal zwei Zugriffsschlüssel besitzen. Wenn Sie bereits zwei Zugriffschlüssel besitzen, müssen Sie ein Schlüsselpaar löschen, bevor Sie ein neues erstellen. Anweisungen hierfür finden Sie unter Verwalten von Zugriffsschlüsseln im IAM-Benutzerhandbuch.

Ich bin Administrator und möchte anderen Zugriff gewähren CodeCommit

Um anderen den Zugriff zu ermöglichen CodeCommit, müssen Sie eine IAM-Entität (Benutzer oder Rolle) für die Person oder Anwendung erstellen, die Zugriff benötigt. Sie werden die Anmeldeinformationen für diese Einrichtung verwenden, um auf AWS zuzugreifen. Anschließend müssen Sie der Entität eine Richtlinie anfügen, die dieser die korrekten Berechtigungen in CodeCommit gewährt.

Informationen zum Einstieg finden Sie unter Erstellen Ihrer ersten delegierten IAM-Benutzer und -Gruppen im IAM-Benutzerhandbuch.

Ich möchte Personen außerhalb meines Amazon Web Services Services-Kontos den Zugriff auf meine CodeCommit Ressourcen ermöglichen

Weitere Informationen finden Sie unter Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen.

Ausfallsicherheit in AWS CodeCommit

Die globale AWS-Infrastruktur ist um AWS-Regionen und Availability Zones herum aufgebaut. AWS-Regionen bieten mehrere physisch getrennte und isolierte Availability Zones, die mit einem Netzwerk mit geringer Latenz, hohem Durchsatz und hoher Redundanz verbunden sind. Mithilfe von Availability Zones können Sie Anwendungen und Datenbanken erstellen und ausführen, die automatisch Failover zwischen Zonen ausführen, ohne dass es zu Unterbrechungen kommt. Availability Zones sind besser verfügbar, fehlertoleranter und skalierbarer als herkömmliche Infrastrukturen mit einem oder mehreren Rechenzentren.

EIN CodeCommit Repository oder CodeCommit Genehmigungsregelvorlage ist in der DateiAWS-Regionwo es geschaffen wurde. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte für AWS CodeCommit. Um die Ausfallsicherheit in Repositorys zu gewährleisten, können Sie Ihren Git-Client so konfigurieren, dass er auf zwei Repositorys gleichzeitig übertragen wird. Weitere Informationen finden Sie unter Push Commits in ein zusätzliches Git-Repository.

Weitere Informationen über AWS-Regionen und Availability Zones finden Sie unter Globale AWS-Infrastruktur.

Sicherheit der Infrastruktur in AWS CodeCommit

Als Managed ServiceAWS CodeCommitist geschützt durch dieAWSglobale Verfahren zur Netzwerksicherheit, die imAmazon Web Services: Übersicht über SicherheitsprozesseWhitepaper.

Du benutztAWSveröffentlichte API-Aufrufe, um über das Netzwerk auf zuzugreifen. Kunden müssen Transport Layer Security (TLS) 1.0 oder neuer unterstützen. Wir empfehlen TLS 1.2 oder höher. Clients müssen außerdem Cipher Suites mit PFS (Perfect Forward Secrecy) wie DHE (Ephemeral

Diffie-Hellman) oder ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) unterstützen. Die meisten modernen Systemen wie Java 7 und höher unterstützen diese Modi.

Anforderungen müssen mit einer Zugriffsschlüssel-ID und einem geheimen Zugriffsschlüssel signiert sein, der mit einem IAM-Prinzipal verknüpft ist. Alternativ können Sie mit AWS Security Token Service (AWS STS) temporäre Sicherheitsanmeldeinformationen erstellen, um die Anforderungen zu signieren.

Sie können diese API-Operationen von jedem Netzwerkstandort aus aufrufen, CodeCommit unterstützt jedoch Einschränkungen basierend auf der IP-Quelladresse. Sie können auch CodeCommit-Richtlinien verwenden, um den Zugriff über bestimmte Amazon Virtual Private Cloud (Amazon VPC) -Endpunkte oder bestimmte VPCs zu steuern. Hierdurch wird der Netzwerkzugriff effektiv auf eine bestimmte CodeCommit-Ressource beschränkt, so dass er ausschließlich über eine bestimmte VPC in derAWS-Netzwerk.

Weitere Informationen finden Sie unter:

- Beispiel 1: Erlauben Sie einem Benutzer, CodeCommit Operationen in einem einzigen System auszuführen AWS-Region
- Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine Verbindung herstellt, Zugriff auf ein Repository
- Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle

Überwachung von AWS CodeCommit

Überwachung ist wichtig zur Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Performance von CodeCommit und deine anderenAWSLösungen.AWSbietet die folgenden Überwachungstools zur Beobachtung von CodeCommit, melden Sie, wenn etwas nicht funktioniert, und ergreifen Sie bei Bedarf automatisch Gegenmaßnahmen:

- Amazon EventBridge kann verwendet werden, um IhreAWS-Services und reagieren automatisch auf Systemereignisse, z. B. bei Problemen mit der Anwendungsverfügbarkeit oder Ressourcenänderungen. Veranstaltungen vonAWSDienstleistungen werden geliefert an EventBridge nahezu in Echtzeit. Sie können einfache Regeln schreiben, um anzugeben, welche Ereignisse für Sie interessant sind und welche automatisierten Aktionen ausgeführt werden sollen, wenn ein Ereignis mit einer Regel übereinstimmt. Weitere Informationen finden Sie unterAmazon EventBridge BenutzerhandbuchundÜberwachung CodeCommit Ereignisse bei Amazon EventBridge und Amazon CloudWatch Events.
- Amazon CloudWatch Ereignisse bietet einen Stream von Systemereignissen nahezu in Echtzeit, der Änderungen in denAWSRessourcen schätzen. CloudWatch Ereignisse ermöglicht automatisierte, ereignisgesteuerte Datenverarbeitung, denn Sie können Regeln schreiben, die bestimmte Ereignisse überwachen und automatisierte Aktionen in anderenAWSDienste, wenn diese Ereignisse eintreten. Weitere Informationen finden Sie im .Amazon CloudWatch BenutzerhandbuchundÜberwachung CodeCommit Ereignisse bei Amazon EventBridge und Amazon CloudWatch Events.
- Amazon CloudWatch Mit den Protokollen können Sie Ihre Protokolldateien überwachen, speichern und darauf zugreifen CloudTrail und anderen Quellen. CloudWatch Logs können Informationen in den Protokolldateien überwachen und Sie benachrichtigen, wenn bestimmte Schwellenwerte erreicht werden. Sie können Ihre Protokolldaten auch in einem sehr robusten Speicher archivieren. Weitere Informationen finden Sie im .Amazon CloudWatch Benutzerhandbuch.
- AWS CloudTrailerfasst API-Aufrufe und zugehörige Ereignisse, die von oder im Namen Ihres Amazon Web Services Services-Kontos erfolgten, und übermittelt die Protokolldateien an einen von Ihnen angegebenen Amazon-S3-Bucket. Sie können die Benutzer und Konten, die AWS aufgerufen haben, identifizieren, sowie die Quell-IP-Adresse, von der diese Aufrufe stammen, und den Zeitpunkt der Aufrufe ermitteln. Weitere Informationen finden Sie im .AWS CloudTrailBenutzerhandbuchundProtokollierung von AWS CodeCommit-API-Aufrufen mit AWS CloudTrail.

Überwachung CodeCommit Ereignisse bei Amazon EventBridge und Amazon CloudWatch Events

Sie können überwachenAWS CodeCommitEreignisse in EventBridge, das einen Stream von Echtzeitdaten aus Ihren eigenen Anwendungen liefert, software-as-a-service (SaaS) -Anwendungen undAWS-Services. EventBridgeleitet diese Daten an Ziele wieAWS Lambdaund Amazon Simple Notification Service. Diese Ereignisse sind mit den in Amazon auftretenden Ereignissen identisch, die in Amazon auftreten CloudWatch Ereignisse, die einen Stream von Systemereignissen nahezu in Echtzeit zur Verfügung stellen, der Änderungen in denAWSRessourcen schätzen.

Die folgenden Beispiele zeigen CodeCommit.

> **Note**
>
> CodeCommit unterstützt die Bereitstellung`displayName`und`emailAddress`Informationen, die in Sitzungs-Tags in Ereignissen enthalten sind, sofern diese Informationen verfügbar sind. Weitere Informationen finden Sie unter Sitzungs-Tags und Verwendung von Tags zur Bereitstellung von Identitätsinformationen in CodeCommit.

Themen

- referenceCreated-Ereignis
- referenceUpdated-Ereignis
- referenceDeleted-Ereignis
- unreferencedMergeCommitErstelltes
- commentOnCommitErstelltes
- commentOnCommitErstelltes
- commentOnPullRequestCreated Veranstaltung
- commentOnPullRequestUpdated Veranstaltung
- pullRequestCreated Veranstaltung
- pullRequestSourceBranchUpdated Veranstaltung
- pullRequestStatusGeändertes Ereignis
- pullRequestMergeStatusUpdated Veranstaltung
- approvalRuleTemplateErstelltes

- approvalRuleTemplateErstelltes
- approvalRuleTemplateErstelltes
- approvalRuleTemplateAssociatedWithRepository Veranstaltung
- approvalRuleTemplateDisassociatedWithRepository Veranstaltung
- approvalRuleTemplateBatchAssociatedWithRepositories Veranstaltung
- approvalRuleTemplateBatchDisassociatedFromRepositories Veranstaltung
- pullRequestApprovalRuleCreated Veranstaltung
- pullRequestApprovalRuleDeleted Veranstaltung
- pullRequestApprovalRuleOverridden Veranstaltung
- pullRequestApprovalStateChanged Veranstaltung
- pullRequestApprovalRuleUpdated Veranstaltung
- ReactionCreated
- ReactionUpdat

referenceCreated-Ereignis

In diesem Beispielereignis wurde eine Verzweigung namens `myBranch` in einem Repository mit dem Namen `MyDemoRepo` erstellt.

```
{
   "version": "0",
   "id": "01234567-EXAMPLE",
   "detail-type": "CodeCommit Repository State Change",
   "source": "aws.codecommit",
   "account": "123456789012",
   "time": "2019-06-12T10:23:43Z",
   "region": "us-east-2",
   "resources": [
     "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
   ],
   "detail": {
     "event": "referenceCreated",
     "repositoryName": "MyDemoRepo",
     "repositoryId": "12345678-1234-5678-abcd-12345678abcd",
     "referenceType": "branch",
     "referenceName": "myBranch",
     "referenceFullName": "refs/heads/myBranch",
```

```
      "commitId": "3e5983DESTINATION"
    }
 }
```

referenceUpdated-Ereignis

In diesem Beispielereignis wurde eine Verzweigung mit dem Namen `myBranch` durch eine Zusammenführung in einem Repository mit dem Namen `MyDemoRepo` aktualisiert.

```
{
   "version": "0",
   "id": "01234567-EXAMPLE",
   "detail-type": "CodeCommit Repository State Change",
   "source": "aws.codecommit",
   "account": "123456789012",
   "time": "2019-06-12T10:23:43Z",
   "region": "us-east-2",
   "resources": [
     "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
   ],
   "detail": {
     "event": "referenceUpdated",
     "repositoryName": "MyDemoRepo",
     "repositoryId": "12345678-1234-5678-abcd-12345678abcd",
     "referenceType": "branch",
     "referenceName": "myBranch",
     "referenceFullName": "refs/heads/myBranch",
     "commitId": "7f0103fMERGE",
     "oldCommitId": "3e5983DESTINATION",
     "baseCommitId": "3e5a9bf1BASE",
     "sourceCommitId": "26a8f2SOURCE",
     "destinationCommitId": "3e5983DESTINATION",
     "mergeOption": "THREE_WAY_MERGE",
     "conflictDetailsLevel": "LINE_LEVEL",
     "conflictResolutionStrategy": "AUTOMERGE"
   }
}
```

referenceDeleted-Ereignis

In diesem Beispielereignis wurde eine Verzweigung namens `myBranch` in einem Repository mit dem Namen `MyDemoRepo` gelöscht.

```
{
  "version": "0",
  "id": "01234567-EXAMPLE",
  "detail-type": "CodeCommit Repository State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-06-12T10:23:43Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "event": "referenceDeleted",
    "repositoryName": "MyDemoRepo",
    "repositoryId": "12345678-1234-5678-abcd-12345678abcd",
    "referenceType": "branch",
    "referenceName": "myBranch",
    "referenceFullName": "refs/heads/myBranch",
    "oldCommitId": "26a8f2EXAMPLE"
  }
}
```

unreferencedMergeCommitErstelltes

In diesem Beispielereignis wurde ein nicht referenziertes Merge-Commit in einem Repository namens `MyDemoRepo` erstellt.

```
{
  "version": "0",
  "id": "01234567-EXAMPLE",
  "detail-type": "CodeCommit Repository State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-06-12T10:23:43Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "event": "unreferencedMergeCommitCreated",
    "repositoryName": "MyDemoRepo",
    "repositoryId": "12345678-1234-5678-abcd-12345678abcd",
```

```
    "commitId": "7f0103fMERGE",
    "baseCommitId": "3e5a9bf1BASE",
    "sourceCommitId": "26a8f2SOURCE",
    "destinationCommitId": "3e5983DESTINATION",
    "mergeOption": "SQUASH_MERGE",
    "conflictDetailsLevel": "LINE_LEVEL",
    "conflictResolutionStrategy": "AUTOMERGE"
  }
}
```

commentOnCommitErstelltes

In diesem Beispielereignis hat ein verbundener Benutzer mit dem Namen `Mary_Major` ein Commit kommentiert. In diesem Beispiel hat der Verbundidentitätsanbieter Sitzungs-Tags für `displayName` und `emailAddress` konfiguriert. Diese Informationen sind in der Veranstaltung enthalten.

```
{
  "version": "0",
  "id": "e9dce2e9-EXAMPLE",
  "detail-type": "CodeCommit Comment on Commit",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-09-29T20:20:39Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "beforeCommitId": "3c5dEXAMPLE",
    "repositoryId": "7dd1EXAMPLE...",
    "inReplyTo": "695bEXAMPLE...",
    "notificationBody": "A comment event occurred in the following repository:
 MyDemoRepo. The display name for the user is Mary Major. The email address for
 the user is mary_major@example.com. The user arn:aws:sts::123456789012:federated-
user/Mary_Major made a comment. The comment was made on the following comment ID:
 463bEXAMPLE.... For more information, go to the AWS CodeCommit console at https://us-
east-2.console.aws.amazon.com/codecommit/home?region=us-east-2#/repository/MyDemoRepo/
compare/3c5dEXAMPLE...f4d5EXAMPLE#463bEXAMPLE....",
    "commentId": "463bEXAMPLE...",
    "afterCommitId": "f4d5EXAMPLE",
    "event": "commentOnCommitCreated",
    "repositoryName": "MyDemoRepo",
    "callerUserArn": "arn:aws:sts::123456789012:federated-user/Mary_Major",
```

```
    "displayName": "Mary Major",
    "emailAddress": "mary_major@example.com"
  }
}
```

commentOnCommitErstelltes

In diesem Beispielereignis hat ein Benutzer, der eine Rolle namens `Admin` mit einem Sitzungsnamen namens `Mary_Major` übernommen hat, einen Kommentar zu einem Commit bearbeitet. In diesem Beispiel enthielt die Rolle konfigurierte Sitzungs-Tags für `displayName` und `emailAddress`. Diese Informationen sind in der Veranstaltung enthalten.

```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Comment on Commit",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "afterCommitId": "53812581",
    "beforeCommitId": "03314446",
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "commentId": "a7e5471e-EXAMPLE",
    "event": "commentOnCommitUpdated",
    "inReplyTo": "bdb07d47-EXAMPLE",
    "notificationBody": "A comment event occurred in the following AWS
 CodeCommit repository: MyDemoRepo. The display name for the user is Mary
 Major. The email address for the user is mary_major@example.com. The user
 arn:aws:sts::123456789012:federated-user/Mary_Major updated a comment or
 replied to a comment. The comment was made on the following comment ID:
 bdb07d47-6fe9-47b0-a839-b93cc743b2ac:468cd1cb-2dfb-4f68-9636-8de52431d1d6.
 For more information, go to the AWS CodeCommit console https://us-
east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/
compare/033144464617842958996982309670958225176 8/.../53812581502937833614716802771360172913 82?
region\u003dus-east-2",
    "repositoryId": "12345678-1234-1234-1234-123456789012",
    "repositoryName": "MyDemoRepo",
    "displayName": "Mary Major",
```

```
    "emailAddress": "mary_major@example.com"
  }
}
```

commentOnPullRequestCreated Veranstaltung

In diesem Beispielereignis hat ein verbundener Benutzer mit dem Namen `Saanvi_Sarkar` eine Pull-Anforderung kommentiert. In diesem Beispiel hat der Verbundidentitätsanbieter Sitzungs-Tags für `displayName` und `emailAddress` konfiguriert. Diese Informationen sind in der Veranstaltung enthalten.

```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Comment on Pull Request",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "beforeCommitId": "3c5dEXAMPLE",
    "repositoryId": "7dd1EXAMPLE...",
    "inReplyTo": "695bEXAMPLE...",
    "notificationBody": "A comment event occurred in the following AWS
 CodeCommit repository: MyDemoRepo. The display name for the user is Saanvi
 Sarkar. The email address for the user is saanvi_sarkar@example.com. The user
 arn:aws:sts::123456789012:federated-user/Saanvi_Sarkar made a comment. The comment
 was made on the following Pull Request: 201. For more information, go to the AWS
 CodeCommit console https://us-east-2.console.aws.amazon.com/codecommit/home?region=us-
east-2#/repository/MyDemoRepo/pull-request/201/activity#3276EXAMPLE...",
    "commentId": "463bEXAMPLE...",
    "afterCommitId": "f4d5EXAMPLE",
    "event": "commentOnPullRequestCreated",
    "repositoryName": "MyDemoRepo",
    "callerUserArn": "arn:aws:sts::123456789012:federated-user/Saanvi_Sarkar",
    "pullRequestId": "201",
    "displayName": "Saanvi Sarkar",
    "emailAddress": "saanvi_sarkar@example.com"
  }
```

```
}
```

commentOnPullRequestUpdated Veranstaltung

In diesem Beispielereignis hat ein verbundener Benutzer namens `Saanvi_Sarkar` einen Kommentar zu einer Pull-Anforderung bearbeitet. In diesem Beispiel hat der Verbundidentitätsanbieter Sitzungs-Tags für `displayName` und `emailAddress` konfiguriert. Diese Informationen sind in der Veranstaltung enthalten.

```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Comment on Pull Request",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "afterCommitId": "96814774EXAMPLE",
    "beforeCommitId": "6031971EXAMPLE",
    "callerUserArn": "arn:aws:sts::123456789012:federated-user/Saanvi_Sarkar",
    "commentId": "40cb52f0-EXAMPLE",
    "event": "commentOnPullRequestUpdated",
    "inReplyTo": "1285e713-EXAMPLE",
    "notificationBody": "A comment event occurred in the following AWS
 CodeCommit repository: MyDemoRepo. The display name for the user is Saanvi
 Sarkar. The email address for the user is saanvi_sarkar@example.com. The user
 arn:aws:sts::123456789012:federated-user/Saanvi_Sarkar updated a comment or
 replied to a comment. The comment was made on the following Pull Request:
 1. For more information, go to the AWS CodeCommit console https://us-
east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/1/activity#40cb52f0-aac7-4c43-b771-601eff02EXAMPLE",
    "pullRequestId": "1",
    "repositoryId": "12345678-1234-1234-1234-123456789012",
    "repositoryName": "MyDemoRepo"
  }
}
```

pullRequestCreated Veranstaltung

In diesem Beispielereignis wurde eine Pull-Anforderung in einem Repository namens `MyDemoRepo` von einem Benutzer erstellt, der eine Rolle namens `Admin` mit dem Sitzungsnamen `Mary_Major` angenommen hat. Es wurden keine Sitzungs-Tag-Informationen bereitgestellt, sodass diese Informationen nicht im Ereignis enthalten sind.


```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "author": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "creationDate": "Tue Feb 9 2019 10:18:42 PDT ",
    "description": "An example description.",
    "destinationCommit": "12241970EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestCreated",
    "isMerged": "False",
    "lastModifiedDate": "Tue Feb 9 2019 10:18:42 PDT",
    "notificationBody": "A pull request event occurred in the following AWS CodeCommit
 repository: MyDemoRepo. User: arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major.
 Event: Created. The pull request was created with the following information: Pull
 Request ID as 1 and title as My Example Pull Request. For more information, go to the
 AWS CodeCommit console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/
repositories/MyDemoRepo/pull-requests/1",
    "pullRequestId": "1",
    "pullRequestStatus": "Open",
    "repositoryNames": ["MyDemoRepo"],
    "revisionId": "bdc0cb9bEXAMPLE",
    "sourceCommit": "2774290EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My Example Pull Request"
  }
```

```
}
```

pullRequestSourceBranchUpdated Veranstaltung

In diesem Beispielereignis hat ein Benutzer, der eine Rolle namens `Admin` mit dem Sitzungsnamen `Mary_Major` übernommen hat, die Quellverzweigung namens `test-branch` für eine Pull-Anforderung mit der ID „1“ aktualisiert.

```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "author": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "creationDate": "Tue Feb 9 2019 10:18:42 PDT",
    "description": "An example description.",
    "destinationCommit": "7644990EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestSourceBranchUpdated",
    "isMerged": "False",
    "lastModifiedDate": "Tue Feb 9 2019 10:18:42 PDT",
    "notificationBody": "A pull request event occurred in the following AWS
 CodeCommit repository: MyDemoRepo. User: arn:aws:sts::123456789012:assumed-role/
Admin/Mary_Major. Event: Updated.  The user updated the following pull request:
 1. The pull request was updated with one or more commits to the source branch:
 test-branch. For more information, go to the AWS CodeCommit console https://us-
east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/1?region\u003dus-east-2",
    "pullRequestId": "1",
    "pullRequestStatus": "Open",
    "repositoryNames": ["MyDemoRepo"],
    "revisionId": "bdc0cb9b4EXAMPLE",
    "sourceCommit": "64875001EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My Example Pull Request"
```

```
  }
}
```

pullRequestStatusGeändertes Ereignis

In diesem Beispielereignis hat ein Benutzer, der eine Rolle namens `Admin` mit dem Sitzungsnamen `Mary_Major` angenommen hat, eine Pull-Anforderung mit der ID „1“ geschlossen. Die Pull-Anforderung wurde nicht zusammengeführt.

```
{
  "version": "0",
  "id": "98377d67-EXAMPLE",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-02-09T07:15:16Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "author": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "creationDate": "Tue Jun 18 10:34:20 PDT 2019",
    "description": "An example description.",
    "destinationCommit": "95149731EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestStatusChanged",
    "isMerged": "False",
    "lastModifiedDate": "Tue Jun 18 10:34:20 PDT 2019",
    "notificationBody": "A pull request event occurred in the following AWS CodeCommit
 repository: MyDemoRepo. arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major
 updated the following PullRequest 1. The pull request status has been updated. The
 status is closed. For more information, go to the AWS CodeCommit console https://
us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/1?region\u003dus-east-2",
    "pullRequestId": "1",
    "pullRequestStatus": "Closed",
    "repositoryNames": ["MyDemoRepo"],
    "revisionId": "bdc0cb9bEXAMPLE",
    "sourceCommit": "4409936EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My Example Pull Request"
```

```
  }
}
```

pullRequestMergeStatusUpdated Veranstaltung

In diesem Beispielereignis hat ein Benutzer, der eine Rolle namens `Admin` mit dem Sitzungsnamen `Mary_Major` übernommen hat, eine Pull-Anforderung mit der ID „1“ zusammengeführt.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "CodeCommit Pull Request State Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-06-12T10:23:43Z",
  "region": "us-east-2",
  "resources": [
    "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
  ],
  "detail": {
    "author": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "callerUserArn": "arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major",
    "creationDate": "Mon Mar 11 14:42:31 PDT 2019",
    "description": "An example description.",
    "destinationCommit": "4376719EXAMPLE",
    "destinationReference": "refs/heads/main",
    "event": "pullRequestMergeStatusUpdated",
    "isMerged": "True",
    "lastModifiedDate": "Mon Mar 11 14:42:31 PDT 2019",
    "mergeOption": "FAST_FORWARD_MERGE",
    "notificationBody": "A pull request event occurred in the following AWS CodeCommit
 repository: MyDemoRepo. arn:aws:sts::123456789012:assumed-role/Admin/Mary_Major
 updated the following PullRequest 1. The pull request merge status has been updated.
 The status is merged. For more information, go to the AWS CodeCommit console https://
us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/1?region\u003dus-east-2",
    "pullRequestId": "1",
    "pullRequestStatus": "Closed",
    "repositoryNames": ["MyDemoRepo"],
    "revisionId": "bdc0cb9beEXAMPLE",
    "sourceCommit": "0701696EXAMPLE",
    "sourceReference": "refs/heads/test-branch",
    "title": "My Example Pull Request"
```

```
  }
}
```

approvalRuleTemplateErstelltes

In diesem Beispielereignis hat ein Benutzer mit dem -BenutzernamenMary_Majorhat eine Genehmigungsregelvorlage mit dem dem2-approvers-required-for-main.

```
{
    "version": "0",
    "id": "f7702227-EXAMPLE",
    "detail-type": "CodeCommit Approval Rule Template Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:02:27Z",
    "region": "us-east-2",
    "resources": [],
    "detail": {
        "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
        "approvalRuleTemplateId": "d7385967-EXAMPLE",
        "approvalRuleTemplateName": "2-approvers-required-for-main",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:02:14 UTC 2019",
        "event": "approvalRuleTemplateCreated",
        "lastModifiedDate": "Wed Nov 06 19:02:14 UTC 2019",
        "notificationBody": "A approval rule template event occurred in the following
 AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user/Mary_Major.
  Additional information: An approval rule template with the following name has been
 created: 2-approvers-required-for-main. The ID of the created template is: d7385967-
EXAMPLE. For more information, go to the AWS CodeCommit console.",
        "repositories": {}
    }
}
```

approvalRuleTemplateErstelltes

In diesem Beispielereignis hat ein Benutzer mit dem -BenutzernamenMary_Majorhat eine Genehmigungsregelvorlage mit dem dem2-approvers-required-for-main. Die Genehmigungsregelvorlage ist keinem Repository zugeordnet.

```
{
  "version": "0",
```

```
  "id": "66403118-EXAMPLE",
  "detail-type": "CodeCommit Approval Rule Template Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-11-12T23:03:30Z",
  "region": "us-east-2",
  "resources": [

  ],
  "detail": {
    "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
    "approvalRuleTemplateId": "c9d2b844-EXAMPLE",
    "approvalRuleTemplateName": "2-approvers-required-for-main",
    "callerUserArn": "arn:aws:iam::123456789012:user\Mary_Major",
    "creationDate": "Tue Nov 12 23:03:06 UTC 2019",
    "event": "approvalRuleTemplateDeleted",
    "lastModifiedDate": "Tue Nov 12 23:03:20 UTC 2019",
    "notificationBody": "A approval rule template event occurred in the following AWS
 CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user\Mary_Major.
 Additional information: An approval rule template with the following name has been
 deleted: 2-approvers-required-for-main. The ID of the updated template is: c9d2b844-
EXAMPLE. For more information, go to the AWS CodeCommit console.",
    "repositories": {}
  }
}
```

approvalRuleTemplateErstelltes

In diesem Beispielereignis hat ein Benutzer mit dem -Benutzernamen`Mary_Major`hat eine Genehmigungsregelvorlage mit dem Namen`2-approvers-required-for-main`. Die Genehmigungsregelvorlage ist keinem Repository zugeordnet.

```
{
  "version": "0",
  "id": "66403118-EXAMPLE",
  "detail-type": "CodeCommit Approval Rule Template Change",
  "source": "aws.codecommit",
  "account": "123456789012",
  "time": "2019-11-12T23:03:30Z",
  "region": "us-east-2",
  "resources": [],
  "detail": {
    "approvalRuleTemplateContentSha256": "4f3de6632EXAMPLE",
```

```
    "approvalRuleTemplateId": "c9d2b844-EXAMPLE",
    "approvalRuleTemplateName": "2-approvers-required-for-main",
    "callerUserArn": "arn:aws:iam::123456789012:user\Mary_Major",
    "creationDate": "Tue Nov 12 23:03:06 UTC 2019",
    "event": "approvalRuleTemplateUpdated",
    "lastModifiedDate": "Tue Nov 12 23:03:20 UTC 2019",
    "notificationBody": "A approval rule template event occurred in the following AWS
 CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user\Mary_Major.
 Additional information: An approval rule template with the following name has
 been updated: 2-approvers-required-for-main. The ID of the updated template is:
 c9d2b844-EXAMPLE. The after rule template content SHA256 is 4f3de663EXAMPLE. For more
 information, go to the AWS CodeCommit console.",
    "repositories": {}
  }
}
```

approvalRuleTemplateAssociatedWithRepository Veranstaltung

In diesem Beispielereignis hat ein Benutzer mit dem -BenutzernamenMary_Majoreine Genehmigungsregelvorlage mit dem dem dem2-approvers-required-for-mainmit einem Repository namensMyDemoRepo.

```
{
    "version": "0",
    "id": "ea1c6d73-EXAMPLE",
    "detail-type": "CodeCommit Approval Rule Template Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:02:27Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
        "approvalRuleTemplateId": "d7385967-EXAMPLE",
        "approvalRuleTemplateName": "2-approvers-required-for-main",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:02:14 UTC 2019",
        "event": "approvalRuleTemplateAssociatedWithRepository",
        "lastModifiedDate": "Wed Nov 06 19:02:14 UTC 2019",
        "notificationBody": "A approval rule template event occurred in the following
 AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user/Mary_Major.
```

```
 Additional information: An approval rule template has been associated with the
 following repository: [MyDemoRepo]. For more information, go to the AWS CodeCommit
 console.",
        "repositories": {
            "MyDemoRepo": "92ca7bf2-d878-49ed-a994-336a6cc7c574"
        }
    }
}
```

approvalRuleTemplateDisassociatedWithRepository Veranstaltung

In diesem Beispielereignis hat ein Benutzer mit dem -BenutzernamenMary_Majorhat eine Genehmigungsregelvorlage mit dem dem dem Namen2-approvers-required-for-mainaus einem Repository namensMyDemoRepo.

```
{
    "version": "0",
    "id": "ea1c6d73-EXAMPLE",
    "detail-type": "CodeCommit Approval Rule Template Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:02:27Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
        "approvalRuleTemplateId": "d7385967-EXAMPLE",
        "approvalRuleTemplateName": "2-approvers-required-for-main",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:02:14 UTC 2019",
        "event": "approvalRuleTemplateDisassociatedFromRepository",
        "lastModifiedDate": "Wed Nov 06 19:02:14 UTC 2019",
        "notificationBody": "A approval rule template event occurred in the following
 AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user/Mary_Major.
 Additional information: An approval rule template has been disassociated from the
 following repository: [MyDemoRepo]. For more information, go to the AWS CodeCommit
 console.",
        "repositories": {
            "MyDemoRepo": "92ca7bf2-d878-49ed-a994-336a6cc7c574"
        }
    }
```

```
}
```

approvalRuleTemplateBatchAssociatedWithRepositories Veranstaltung

In diesem Beispielereignis hat ein Benutzer mit dem -BenutzernamenMary_Majorhat eine Genehmigungsregelvorlage mit dem dem dem Namen2-approvers-required-for-mainmit einem Repository namensMyDemoRepound ein Repository namensMyTestRepo.

```
{
    "version": "0",
    "id": "0f861e5b-EXAMPLE",
    "detail-type": "CodeCommit Approval Rule Template Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-12T23:39:09Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
        "approvalRuleTemplateId": "c71c1fe0-EXAMPLE",
        "approvalRuleTemplateName": "2-approvers-required-for-main",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Tue Nov 12 23:38:57 UTC 2019",
        "event": "batchAssociateApprovalRuleTemplateWithRepositories",
        "lastModifiedDate": "Tue Nov 12 23:38:57 UTC 2019",
        "notificationBody": "A approval rule template event occurred in the following
 AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user\Mary_Major.
 Additional information: An approval rule template has been batch associated with the
 following repository names: [MyDemoRepo, MyTestRepo]. For more information, go to the
 AWS CodeCommit console.",
        "repositories": {
            "MyDemoRepo": "MyTestRepo"
        }
    }
}
```

approvalRuleTemplateBatchDisassociatedFromRepositories Veranstaltung

In diesem Beispielereignis hat ein Benutzer mit dem -BenutzernamenMary_Majorhat eine Genehmigungsregelvorlage mit dem dem dem dem Namen2-approvers-required-for-mainaus einem Repository namensMyDemoRepound ein Repository namensMyTestRepo.

```
{
    "version": "0",
    "id": "e08fc996-EXAMPLE",
    "detail-type": "CodeCommit Approval Rule Template Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-12T23:39:09Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalRuleTemplateContentSha256": "f742eebbEXAMPLE",
        "approvalRuleTemplateId": "c71c1fe0-ff91-4db4-9a45-a86a7b6c474f",
        "approvalRuleTemplateName": "2-approvers-required-for-main",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Tue Nov 12 23:38:57 UTC 2019",
        "event": "batchDisassociateApprovalRuleTemplateFromRepositories",
        "lastModifiedDate": "Tue Nov 12 23:38:57 UTC 2019",
        "notificationBody": "A approval rule template event occurred in the following
 AWS CodeCommit account: 123456789012. User: arn:aws:iam::123456789012:user/Mary_Major.
 Additional information: An approval rule template has been batch disassociated from
 the following repository names: [MyDemoRepo, MyTestRepo]. For more information, go to
 the AWS CodeCommit console.",
        "repositories": {
            "MyDemoRepo": "MyTestRepo"
        }
    }
}
```

pullRequestApprovalRuleCreated Veranstaltung

In diesem Beispielereignis hat ein Benutzer mit dem -BenutzernamenMary_Majorhat eine Genehmigungsregel mit dem Namen1-approver-neededfür eine Pull-Anforderung mit der ID227.

```
{
```

```
    "version": "0",
    "id": "ad860f12-EXAMPLE",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalRuleContentSha256": "f742eebbEXAMPLE",
        "approvalRuleId": "0a9b5dfc-EXAMPLE",
        "approvalRuleName": "1-approver-needed",
        "author": "arn:aws:iam::123456789012:user/Mary_Major",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalRuleCreated",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
        "notificationBody": "A pull request event occurred in the following AWS
 CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/Mary_Major.
 Event: Updated. Pull request: 227. Additional information: An approval rule has been
 created with the following name: 1-approver-needed. For more information, go to the
 AWS CodeCommit console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/
repositories/MyDemoRepo/pull-requests/227?region=us-east-2",
        "pullRequestId": "227",
        "pullRequestStatus": "Open",
        "repositoryNames": [
            "MyDemoRepo"
        ],
        "revisionId": "3b8cecab3EXAMPLE",
        "sourceCommit": "29964a17EXAMPLE",
        "sourceReference": "refs/heads/test-branch",
        "title": "My example pull request"
    }
}
```

pullRequestApprovalRuleDeleted Veranstaltung

In diesem Beispielereignis hat ein Benutzer mit dem -Benutzernamen`Mary_Major`hat eine Genehmigungsregel mit dem Namen`1-approver-needed`für eine Pull-Anforderung mit der ID`227`. Ein IAM-Benutzer mit dem Namen`Saanvi_Sarkar`hat die Genehmigungsregel ursprünglich erstellt.


```
{
    "version": "0",
    "id": "c1c3509d-EXAMPLE",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalRuleContentSha256": "f742eebbEXAMPLE",
        "approvalRuleId": "0a9b5dfc-EXAMPLE",
        "approvalRuleName": "1-approver-needed",
        "author": "arn:aws:iam::123456789012:user/Saanvi_Sarkar",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalRuleDeleted",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
        "notificationBody": "A pull request event occurred in the following AWS
 CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/Mary_Major.
 Event: Created. Pull request: 227. Additional information: An approval rule has been
 deleted: 1-approver-needed was deleted. For more information, go to the AWS CodeCommit
 console https://us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/
MyDemoRepo/pull-requests/227?region=us-east-2",
        "pullRequestId": "227",
        "pullRequestStatus": "Open",
        "repositoryNames": [
            "MyDemoRepo"
        ],
        "revisionId": "3b8cecabEXAMPLE",
        "sourceCommit": "29964a17EXAMPLE",
```

```
        "sourceReference": "refs/heads/test-branch",
        "title": "My example pull request"
    }
}
```

pullRequestApprovalRuleOverridden Veranstaltung

In diesem Beispielereignis wurden die Genehmigungsregelanforderungen für eine Pull-Anforderung von einem Benutzer mit dem`Mary_Major`. Die Pull-Anforderung wurde von einem Benutzer mit dem IAM-Benutzernamen`Li_Juan`.

```
{
    "version": "0",
    "id": "52d2cb73-EXAMPLE",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "author": "arn:aws:iam::123456789012:user/Li_Juan",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalRuleOverridden",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
        "notificationBody": "A pull request event occurred in the following AWS
 CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/Mary_Major.
 Event: Updated. Pull request name: 227. Additional information: An override
 event has occurred for the approval rules for this pull request. Override status:
 OVERRIDE. For more information, go to the AWS CodeCommit console https://us-
east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/227?region=us-east-2",
        "overrideStatus": "OVERRIDE",
        "pullRequestId": "227",
        "pullRequestStatus": "Open",
        "repositoryNames": [
```

```
            "MyDemoRepo"
        ],
        "revisionId": "3b8cecabEXAMPLE",
        "sourceCommit": "29964a17EXAMPLE",
        "sourceReference": "refs/heads/test-branch",
        "title": "My example pull request"
    }
}
```

In diesem Beispiel wurden die Genehmigungsregelanforderungen für eine Pull-Anforderung wieder eingeführt („REVOKE“).

```
{
    "version": "0",
    "id": "2895482d-13eb-b783-270d-76588e6029fa",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "author": "arn:aws:iam::123456789012:user/Li_Juan",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalRuleOverridden",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
        "notificationBody": "A pull request event occurred in the following
 AWS CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/
Mary_Major. Event: Updated. Pull request name: 227. Additional information: An
 override event has occurred for the approval rules for this pull request. Override
 status: REVOKE. For more information, go to the AWS CodeCommit console https://
us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/227?region=us-east-2",
        "overrideStatus": "REVOKE",
        "pullRequestId": "227",
        "pullRequestStatus": "Open",
```

```
        "repositoryNames": [
            "MyDemoRepo"
        ],
        "revisionId": "3b8cecabEXAMPLE",
        "sourceCommit": "29964a17EXAMPLE",
        "sourceReference": "refs/heads/test-branch",
        "title": "My example pull request"
    }
}
```

pullRequestApprovalStateChanged Veranstaltung

In diesem Beispielereignis wurde eine Pull-Anforderung von einem Benutzer mit dem IAM-Benutzernamen genehmigt`Mary_Major`.

```
{
    "version": "0",
    "id": "53e5d7e9-986c-1ebf-9d8b-ebef5596da0e",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalStatus": "APPROVE",
        "author": "arn:aws:iam::123456789012:user/Li_Juan",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalStateChanged",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
        "notificationBody": "A pull request event occurred in the following
 AWS CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/
Mary_Major. Event: Updated. Pull request name: 227. Additional information:
 A user has changed their approval state for the pull request. State change:
 APPROVE. For more information, go to the AWS CodeCommit console https://us-
```

```
east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/227?region=us-east-2",
        "pullRequestId": "227",
        "pullRequestStatus": "Open",
        "repositoryNames": [
            "MyDemoRepo"
        ],
        "revisionId": "3b8cecabEXAMPLE",
        "sourceCommit": "29964a17EXAMPLE",
        "sourceReference": "refs/heads/test-branch",
        "title": "My example pull request"
    }
}
```

In diesem Beispielereignis wurde eine Genehmigung für eine Pull-Anforderung von einem Benutzer mit dem IAM-Benutzernamen widerrufen`Mary_Major`.

```
{
    "version": "0",
    "id": "25e183d7-d01a-4e07-2bd9-b2d56ebecc81",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalStatus": "REVOKE",
        "author": "arn:aws:iam::123456789012:user/Li_Juan",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalStateChanged",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
        "notificationBody": "A pull request event occurred in the following AWS
 CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/Mary_Major.
 Event: Updated. Pull request name: 227. Additional information: A user has changed
 their approval state for the pull request. State change: REVOKE. For more information,
```

```
 go to the AWS CodeCommit console https://us-east-2.console.aws.amazon.com/codesuite/
codecommit/repositories/MyDemoRepo/pull-requests/227?region=us-east-2",
        "pullRequestId": "227",
        "pullRequestStatus": "Open",
        "repositoryNames": [
            "MyDemoRepo"
        ],
        "revisionId": "3b8cecabEXAMPLE",
        "sourceCommit": "29964a17EXAMPLE",
        "sourceReference": "refs/heads/test-branch",
        "title": "My example pull request"
    }
}
```

pullRequestApprovalRuleUpdated Veranstaltung

In diesem Beispielereignis wurde eine Genehmigungsregel für eine Pull-Anforderung von einem Benutzer mit dem`Mary_Major`. Sie ist auch die Benutzerin, die die Pull-Anfrage erstellt hat.

```
{
    "version": "0",
    "id": "21b1c819-2889-3528-1cb8-3861aacf9d42",
    "detail-type": "CodeCommit Pull Request State Change",
    "source": "aws.codecommit",
    "account": "123456789012",
    "time": "2019-11-06T19:12:19Z",
    "region": "us-east-2",
    "resources": [
        "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
    ],
    "detail": {
        "approvalRuleContentSha256": "f742eebbEXAMPLE",
        "approvalRuleId": "0a9b5dfc-EXAMPLE",
        "approvalRuleName": "1-approver-needed",
        "author": "arn:aws:iam::123456789012:user/Mary_Major",
        "callerUserArn": "arn:aws:iam::123456789012:user/Mary_Major",
        "creationDate": "Wed Nov 06 19:10:58 UTC 2019",
        "description": "An example description.",
        "destinationCommit": "194fdf00EXAMPLE",
        "destinationReference": "refs/heads/main",
        "event": "pullRequestApprovalRuleUpdated",
        "isMerged": "False",
        "lastModifiedDate": "Wed Nov 06 19:10:58 UTC 2019",
```

```
        "notificationBody": "A pull request event occurred in the following
 AWS CodeCommit repository: MyDemoRepo. User: arn:aws:iam::123456789012:user/
Mary_Major. Event: Updated. Pull request name: 227. The content of an approval
 rule has been updated for the pull request. The name of the updated rule is: 1-
approver-needed. For more information, go to the AWS CodeCommit console https://
us-east-2.console.aws.amazon.com/codesuite/codecommit/repositories/MyDemoRepo/pull-
requests/227?region=us-east-2",
        "pullRequestId": "227",
        "pullRequestStatus": "Open",
        "repositoryNames": [
            "MyDemoRepo"
        ],
        "revisionId": "3b8cecab3EXAMPLE",
        "sourceCommit": "29964a17EXAMPLE",
        "sourceReference": "refs/heads/test-branch",
        "title": "My example pull request"
    }
}
```

ReactionCreated

In diesem Beispielereignis wurde eine Reaktion auf eine -Kommentar von einem Benutzer mit dem IAM-Benutzernamen eine Reaktion auf einen Kommentar von einem Benutzer mit dem -Benutzernamen eine`Mary_Major`.

```
{
   "version":"0",
   "id":"59fcccd8-217a-32ce-2b05-561ed68a1c42",
   "detail-type":"CodeCommit Comment Reaction Change",
   "source":"aws.codecommit",
   "account":"123456789012",
   "time":"2020-04-14T00:49:03Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
   ],
   "detail":{
      "callerUserArn":"arn:aws:iam::123456789012:user/Mary_Major",
      "commentId":"28930161-EXAMPLE",
      "event":"commentReactionCreated",
      "notificationBody":"A comment reaction event occurred in the following AWS
 CodeCommit Repository: MyDemoRepo. The user: arn:aws:iam::123456789012:user/Mary_Major
 made a comment reaction # to the comment with comment ID: 28930161-EXAMPLE",
```

```
      "reactionEmojis":["#"],
      "reactionShortcodes":[":thumbsdown:"],
      "reactionUnicodes":["U+1F44E"],
      "repositoryId":"12345678-1234-5678-abcd-12345678abcd",
      "repositoryName":"MyDemoRepo"
   }
}
```

ReactionUpdat

In diesem Beispielereignis wurde eine Reaktion auf eine -Kommentar von einem Benutzer mit dem IAM-BenutzernamenMary_Major. Benutzer können nur ihre eigenen Reaktionen aktualisieren.

```
{
   "version":"0",
   "id":"0844ed99-a53f-3bdb-6048-4de315516889",
   "detail-type":"CodeCommit Comment Reaction Change",
   "source":"aws.codecommit",
   "account":"123456789012",
   "time":"2020-04-22T23:19:42Z",
   "region":"us-east-2",
   "resources":[
      "arn:aws:codecommit:us-east-2:123456789012:MyDemoRepo"
   ],
   "detail":{
      "callerUserArn":"arn:aws:iam::123456789012:user/Mary_Major",
      "commentId":"28930161-EXAMPLE",
      "event":"commentReactionUpdated",
      "notificationBody":"A comment reaction event occurred in the following AWS
 CodeCommit Repository: MyDemoRepo. The user: arn:aws:iam::123456789012:user/Mary_Major
 updated a reaction :smile: to the comment with comment ID: 28930161-EXAMPLE",
      "reactionEmojis":[
         "#"
      ],
      "reactionShortcodes":[
         ":smile:"
      ],
      "reactionUnicodes":[
         "U+1F604"
      ],
      "repositoryId":"12345678-1234-5678-abcd-12345678abcd",
      "repositoryName":"MyDemoRepo"
   }
```

```
}
```

Protokollierung von AWS CodeCommit-API-Aufrufen mit AWS CloudTrail

CodeCommit ist in integriertAWS CloudTrail, einen Service, der eine Aufzeichnung der von einem Benutzer, einer Rolle oder einemAWS -Service durchgeführten Aktionen erfasst CodeCommit. CloudTrail erfasst alle API-Aufrufe, einschließlich CodeCommit Aufrufen von der CodeCommit Konsole, Ihrem Git-Client und von Code-Aufrufen an die CodeCommit APIs. Wenn Sie einen Trail erstellen, können Sie die kontinuierliche Bereitstellung von CloudTrail Ereignissen an einen Amazon S3-Bucket, einschließlich Ereignisse für Amazon S3-Bucket aktivieren CodeCommit. Wenn Sie keinen Trail konfigurieren, können Sie die neuesten Ereignisse in der CloudTrail Konsole trotzdem in Event history (Ereignisverlauf) anzeigen. Mit den von CloudTrail gesammelten Informationen können Sie die an gestellte Anfrage CodeCommit, die IP-Adresse, von der die Anfrage gestellt wurde, den Initiator der Anfrage, den Zeitpunkt der Anfrage und weitere Angaben bestimmen.

Weitere Informationen CloudTrail finden Sie im AWS CloudTrailBenutzerhandbuch.

CodeCommit Informationen in CloudTrail

CloudTrail wird beim Erstellen Ihres Amazon-Web-Services-Kontos für Sie aktiviert. Die in CodeCommit Ereignisverlauf auftretenden Aktivitäten werden als CloudTrail Ereignis zusammen mit anderenAWS -Serviceereignissen in Event history (Ereignisverlauf) aufgezeichnet. Sie können die neusten Ereignisse in Ihrem Amazon Web Services-Konto anzeigen, suchen und herunterladen. Weitere Informationen finden Sie unter Anzeigen von Ereignissen mit dem CloudTrail - Ereignisverlauf.

Erstellen Sie einen Trail zur laufenden Aufzeichnung der Ereignisse in Ihrem Amazon Web Services Services-Konto, einschließlich Ereignisse für Sie CodeCommit, erstellen Sie einen Trail. Ein Trail ermöglicht es CloudTrail , Protokolldateien in einem Amazon S3 S3-Bucket bereitzustellen. Wenn Sie einen Trail in der Konsole anlegen, gilt dieser standardmäßig für alle Regionen. Der Trail protokolliert Ereignisse aus allen Regionen in der AWS-Partition und stellt die Protokolldateien in dem von Ihnen angegebenen Amazon-S3-Bucket bereit. Darüber hinaus können Sie andereAWS -Services konfigurieren, um die in den CloudTrail -Protokollen erfassten Ereignisdaten weiter zu analysieren und entsprechend zu agieren. Weitere Informationen finden Sie unter:

- Übersicht zum Erstellen eines Trails

- CloudTrail Unterstützte Dienste und Integrationen
- Konfigurieren Amazon SNS-Amazon SNS-Amazon SNS-Amazon SNS- CloudTrail
- Empfangen von CloudTrail Protokolldateien aus mehreren Regionen und Empfangen von CloudTrail Protokolldateien aus mehreren Konten.

Wenn die CloudTrail Protokollierung in Ihrem Amazon Web Services Services-Konto aktiviert ist, werden API-Aufrufe für CodeCommit Aktionen in CloudTrail Protokolldateien aufgezeichnet, wo sie zusammen mit anderenAWS -Servicedatensätzen aufgezeichnet werden. CloudTrail bestimmt anhand eines Zeitraums und der Dateigröße, wann diese Informationen in eine neue erstellte Protokolldatei geschrieben werden sollen.

Alle CodeCommit Aktionen werden protokolliert CloudTrail, einschließlich einiger (z. B.`GetObjectIdentifier`), die derzeit nicht in der AWS CodeCommitAPI-Referenz dokumentiert sind, sondern stattdessen als Zugriffsberechtigungen referenziert und in dokumentiert sindReferenz für CodeCommit-Berechtigungen. Beispielsweise generieren Aufrufe der Aktionen`ListRepositories` (in den AktionenAWS CLI,`aws codecommit list-repositories`),`CreateRepository`Put`RepositoryTriggers` (`aws codecommit create-repository`aws `codecommit put-repository-triggers`) und () Einträge in den CloudTrail Protokolldateien sowie Git-Client-Aufrufe an`GitPull` und`GitPush`. Wenn Sie ein CodeCommit Repository als Quelle für eine Pipeline konfiguriert haben CodePipeline, werden außerdem Aufrufe für CodeCommit Zugriffsberechtigungsaktionen angezeigt, z. B.`UploadArchive` von CodePipeline. Da CodeCommit AWS Key Management Service zur Ver- und Entschlüsselung von Repositorys verwendet, werden auch Aufrufe von CodeCommit an `Encrypt`- und `Decrypt`-Aktionen von AWS KMS in den CloudTrail -Protokollen angezeigt.

Jeder Protokolleintrag enthält Informationen über den Ersteller der Anforderung. Der Benutzeridentität im Protokolleintrag können Sie folgende Informationen entnehmen:

- Ob die Anfrage mit Root- oder IAM-Benutzer-Anmeldeinformationen ausgeführt wurde
- Ob die Anfrage mit temporären Sicherheitsanmeldeinformationen für eine Funktion oder einen verbundenen Benutzer oder von einer angenommenen Rolle getätigt wurde
- Ob die Anforderung von einem anderen AWS-Service getätigt wurde.

Weitere Informationen finden Sie unter CloudTrail userIdentity-Element.

Sie können Ihre Protokolldateien beliebig lange im Amazon S3 S3-Bucket speichern. Sie können aber auch Amazon S3 S3-Lebenszyklus3-Lebenszyklusy-Regeln aufstellen, anhand derer

die Protokolldateien automatisch archiviert oder gelöscht werden. Standardmäßig werden die Protokolldateien mit serverseitiger Amazon S3 S3-Verschlüsselung (SSE) verschlüsselt.

Grundlagen zu CodeCommit -Protokolldateieinträgen

CloudTrail -Protokolldateien können einen oder mehrere Einträge enthalten. In jedem Eintrag werden mehrere Ereignisse im JSON-Format aufgelistet. Ein Protokollereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält unter anderem Informationen über die angeforderte Aktion, das Datum und die Uhrzeit der Aktion sowie über die Anforderungsparameter. Protokolleinträge sind kein geordnetes Stacktrace der öffentlichen API-Aufrufe und erscheinen daher nicht in einer bestimmten Reihenfolge.

> **Note**
>
> Dieses Beispiel wurde zur Verbesserung der Lesbarkeit formatiert. In einer CloudTrail Protokolldatei werden alle Einträge und Ereignisse in einer einzigen Zeile zusammengefügt. Dieses Beispiel ist ebenfalls auf einen einzigen CodeCommit Eintrag beschränkt. In einer echten CloudTrail Protokolldatei sehen Sie Einträge und Ereignisse von mehrerenAWS Diensten.

Inhalt

- Beispiel: Ein Logeintrag zum Auflisten von CodeCommit Repositorys
- Beispiel: Ein Logeintrag für die Erstellung eines CodeCommit Repositorys
- Beispiele: Protokolleinträge zum Abrufen von Git-Aufrufen an ein CodeCommit-Repository
- Beispiel: Ein Logeintrag für einen erfolgreichen Push in ein CodeCommit Repository

Beispiel: Ein Logeintrag zum Auflisten von CodeCommit Repositorys

Das folgende Beispiel zeigt einen CloudTrail -Protokolleintrag, der die`ListRepositories` Aktion veranschaulicht.

> **Note**
>
> `ListRepositories`Gibt zwar eine Liste von Repositorys zurück, unveränderliche Antworten werden jedoch nicht in CloudTrail Protokollen aufgezeichnet, sondern wie`null` in der Protokolldatei angezeigt.`responseElements`

```
{
  "eventVersion":"1.05",
  "userIdentity": {
    "type":"IAMUser",
    "principalId":"AIDACKCEVSQ6C2EXAMPLE",
    "arn":"arn:aws:iam::444455556666:user/Mary_Major",
    "accountId":"444455556666",
    "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
    "userName":"Mary_Major"
    },
  "eventTime":"2016-12-14T17:57:36Z",
  "eventSource":"codecommit.amazonaws.com",
  "eventName":"ListRepositories",
  "awsRegion":"us-east-1",
  "sourceIPAddress":"203.0.113.12",
  "userAgent":"aws-cli/1.10.53 Python/2.7.9 Windows/8 botocore/1.4.43",
  "requestParameters":null,
  "responseElements":null,
  "requestID":"cb8c167e-EXAMPLE",
  "eventID":"e3c6f4ce-EXAMPLE",
  "readOnly":true,
  "eventType":"AwsApiCall",
  "apiVersion":"2015-04-13",
  "recipientAccountId":"444455556666"
}
```

Beispiel: Ein Logeintrag für die Erstellung eines CodeCommit Repositorys

Das folgende Beispiel zeigt einen CloudTrail -Protokolleintrag, der die`CreateRepository` Aktion in der Region USA Ost (Ohio) veranschaulicht.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName":"Mary_Major"
  },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
```

```
  "eventName": "CreateRepository",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.12",
  "userAgent": "aws-cli/1.10.53 Python/2.7.9 Windows/8 botocore/1.4.43",
  "requestParameters": {
    "repositoryDescription": "Creating a demonstration repository.",
    "repositoryName": "MyDemoRepo"
  },
  "responseElements": {
    "repositoryMetadata": {
      "arn": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "creationDate": "Dec 14, 2016 6:19:14 PM",
      "repositoryId": "8afe792d-EXAMPLE",
      "cloneUrlSsh": "ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/
MyDemoRepo",
      "repositoryName": "MyDemoRepo",
      "accountId": "111122223333",
      "cloneUrlHttp": "https://git-codecommit.us-east-2.amazonaws.com/v1/repos/
MyDemoRepo",
      "repositoryDescription": "Creating a demonstration repository.",
      "lastModifiedDate": "Dec 14, 2016 6:19:14 PM"
    }
  },
  "requestID": "d148de46-EXAMPLE",
  "eventID": "740f179d-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "accountId": "111122223333",
      "type": "AWS::CodeCommit::Repository"
    }
  ],
  "eventType": "AwsApiCall",
  "apiVersion": "2015-04-13",
  "recipientAccountId": "111122223333"
}
```

Beispiele: Protokolleinträge zum Abrufen von Git-Aufrufen an ein CodeCommit-Repository

Das folgende Beispiel zeigt einen CloudTrail -Protokolleintrag, der die`GitPull` Aktion veranschaulicht up-to-date.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName":"Mary_Major"
    },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
  "eventName": "GitPull",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.12",
  "userAgent": "git/2.11.0.windows.1",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData": {
    "protocol": "HTTP",
    "dataTransferred": false,
    "repositoryName": "MyDemoRepo",
    "repositoryId": "8afe792d-EXAMPLE",
    },
  "requestID": "d148de46-EXAMPLE",
  "eventID": "740f179d-EXAMPLE",
  "readOnly": true,
  "resources": [
    {
      "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "accountId": "111122223333",
      "type": "AWS::CodeCommit::Repository"
      }
    ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```


Das folgende Beispiel zeigt einen CloudTrail Protokolleintrag, der die`GitPull` Aktion demonstriert, bei der sich das lokale Repo nicht up-to-date befindet, sodass Daten vom CodeCommit Repository in das lokale Repo übertragen werden.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName":"Mary_Major"
    },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
  "eventName": "GitPull",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.12",
  "userAgent": "git/2.10.1",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData": {
    "protocol": "HTTP",
    "capabilities": [
      "multi_ack_detailed",
      "side-band-64k",
      "thin-pack"
      ],
    "dataTransferred": true,
    "repositoryName": "MyDemoRepo",
    "repositoryId": "8afe792d-EXAMPLE",
    "shallow": false
    },
  "requestID": "d148de46-EXAMPLE",
  "eventID": "740f179d-EXAMPLE",
  "readOnly": true,
  "resources": [
    {
      "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "accountId": "111122223333",
      "type": "AWS::CodeCommit::Repository"
      }
    ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

Beispiel: Ein Logeintrag für einen erfolgreichen Push in ein CodeCommit Repository

Das folgende Beispiel zeigt einen CloudTrail -Protokolleintrag, der die erfolgreiche`GitPush` Aktion demonstriert. Die Aktion `GitPush` wird im Protokoll bei einem erfolgreichen Push-Vorgang zweimal angezeigt.


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName":"Mary_Major"
    },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
  "eventName": "GitPush",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.12",
  "userAgent": "git/2.10.1",
  "requestParameters": null,
  "responseElements": null,
  "additionalEventData": {
    "protocol": "HTTP",
    "dataTransferred": false,
    "repositoryName": "MyDemoRepo",
    "repositoryId": "8afe792d-EXAMPLE",
    },
  "requestID": "d148de46-EXAMPLE",
  "eventID": "740f179d-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "accountId": "111122223333",
      "type": "AWS::CodeCommit::Repository"
      }
    ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
},
```

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "type": "IAMUser",
    "principalId": "AIDACKCEVSQ6C2EXAMPLE",
    "arn": "arn:aws:iam::444455556666:user/Mary_Major",
    "accountId": "444455556666",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "userName":"Mary_Major"
    },
  "eventTime": "2016-12-14T18:19:15Z",
  "eventSource": "codecommit.amazonaws.com",
  "eventName": "GitPush",
  "awsRegion": "us-east-2",
  "sourceIPAddress": "203.0.113.12",
  "userAgent": "git/2.10.1",
  "requestParameters": {
    "references": [
      {
        "commit": "100644EXAMPLE",
        "ref": "refs/heads/main"
        }
      ]
    },
  "responseElements": null,
  "additionalEventData": {
    "protocol": "HTTP",
    "capabilities": [
      "report-status",
      "side-band-64k"
      ],
    "dataTransferred": true,
    "repositoryName": "MyDemoRepo",
    "repositoryId": "8afe792d-EXAMPLE",
    },
  "requestID": "d148de46-EXAMPLE",
  "eventID": "740f179d-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:codecommit:us-east-2:111122223333:MyDemoRepo",
      "accountId": "111122223333",
      "type": "AWS::CodeCommit::Repository"
      }
```

```
    ],
  "eventType": "AwsApiCall",
  "recipientAccountId": "111122223333"
}
```

CodeCommit Ressourcen erstellen mit AWS CloudFormation

AWS CodeCommit ist in AWS CloudFormation integriert, ein Service, der Ihnen hilft, Ihre AWS-Ressourcen zu modellieren und einzurichten, damit Sie weniger Zeit mit der Erstellung und Verwaltung Ihrer Ressourcen und Infrastruktur verbringen können. Sie erstellen eine Vorlage, die alle gewünschtenAWS -Ressourcen beschreibt (z. B. Repositories), und stelltAWS CloudFormation diese Ressourcen bereit und konfiguriert sie für Sie.

Wenn Sie AWS CloudFormation verwenden, können Sie Ihre Vorlage wiederverwenden, um Ihre CodeCommit-Ressourcen einheitlich und wiederholt einzurichten. Sie beschreiben Ihre Ressourcen dann einmal und können die gleichen Ressourcen dann in mehreren AWS-Konten-Konten und -Regionen immer wieder bereitstellen.

CodeCommit- und AWS CloudFormation-Vorlagen

Um Ressourcen für CodeCommit und verwandte Dienstleistungen bereitzustellen und zu konfigurieren, müssen Sie AWS CloudFormation-Vorlagen kennen und verstehen. Vorlagen sind formatierte Textdateien in JSON oder YAML. Diese Vorlagen beschreiben die Ressourcen, die Sie in Ihren AWS CloudFormation-Stacks bereitstellen möchten. Wenn Sie noch keine Erfahrungen mit JSON oder YAML haben, können Sie AWS CloudFormation Designer verwenden, der den Einstieg in die Arbeit mit AWS CloudFormation-Vorlagen erleichtert. Weitere Informationen finden Sie unter Was ist AWS CloudFormation-Designer? im AWS CloudFormation-Benutzerhandbuch.

CodeCommitunterstützt das Erstellen von Repositorys inAWS CloudFormation Im Gegensatz zum Erstellen von Repositorys über die Konsole oder die Befehlszeile können SieAWS CloudFormation damit Repositorys erstellen und automatisch Code aus einer angegebenen .zip-Datei in einem Amazon S3 S3-Bucket in das neu erstellte Repository übertragen. Weitere Informationen, einschließlich Beispiele für JSON- und YAML-Vorlagen für -Repository AWS::CodeCommit::Repository

Wenn Sie einCodeCommit Repository mit erstellenAWS CloudFormation, haben Sie die Möglichkeit, Code im Rahmen des Erstellungsprozesses in dieses Repository zu übertragen, sofern das Archiv weniger als 20 MB groß ist, indem Sie Eigenschaften in AWS:CodeCommit: :Repository-Code konfigurieren. Sie können den Amazon S3 S3-Bucket angeben, in dem der Code gespeichert ist, und optional die BranchNameEigenschaft verwenden, um den Namen des Standard-Branches

anzugeben, der beim ersten Commit dieses Codes erstellt wird. Diese Eigenschaften werden nur bei der ersten Erstellung des Repositorys verwendet und bei Stack-Updates ignoriert. Sie können diese Eigenschaften nicht verwenden, um zusätzliche Commits an ein Repository vorzunehmen oder den Namen des Standard-Branches zu ändern, nachdem der erste Commit durchgeführt wurde.

> **Note**
>
> Am 19. Januar 2021AWS wurde der Name des Standard-BranchesCodeCommit von Master zu Main geändert. Diese Namensänderung wirkt sich auf das StandardverhaltenCodeCommit bei der Erstellung des ersten Commits für Repositorys mithilfe derCodeCommit Konsole, derCodeCommit APIs, derAWS SDKs und der ausAWS CLI. Repositorys, die im Rahmen der ErstellungAWS CDK mitAWS CloudFormation oder mit einem ersten Commit von Code erstellt wurden, entsprechen dieser Änderung ab dem 4. März 2021. Diese Änderung wirkt sich nicht auf bestehende Repositorys oder Branches aus. Kunden, die lokale Git-Clients verwenden, um ihre ersten Commits zu erstellen, haben einen standardmäßigen Branchnamen, der der Konfiguration dieser Git-Clients folgt. Weitere Informationen findest du unter Arbeiten mit Branches, Erstellen eines Commits und Ändern der Branch-Einstellungen.

Sie können auch Vorlagen erstellen, mit denen verwandte Ressourcen erstellt werden, z. B. Benachrichtigungsregeln für Repositorys, AWS CodeBuildBuild-Projekte, AWS CodeDeployAnwendungen und AWS CodePipelinePipelines.

Beispiele für ---

In den folgenden Beispielen wird einCodeCommit Repository mit dem Namen erstellt *`MyDemoRepo`*. Das neu erstellte Repository wird mit Code gefüllt, der in einem Amazon S3 S3-Bucket gespeichert *`MySourceCodeBucket`*und in einer Verzweigung mit dem Namen *`development`* platziert ist.

> **Note**
>
> Der Name des Amazon S3 S3-Buckets, der ZIP-Datei mit den Inhalten enthält, die in das neue Repository überragt Amazon Web Services. Der Amazon-S3-Objektschlüssel ist im Amazon-S3-Entwicklerhandbuch definiert.

JSON:

```
{
    "MyRepo": {
        "Type": "AWS::CodeCommit::Repository",
        "Properties": {
            "RepositoryName": "MyDemoRepo",
            "RepositoryDescription": "This is a repository for my project with code
 from MySourceCodeBucket.",
            "Code": {
                "BranchName": "development",
                "S3": {
                    "Bucket": "MySourceCodeBucket",
                    "Key": "MyKey",
                    "ObjectVersion": "1"
                }
            }
        }
    }
}
```

YAML:

```
MyRepo:
  Type: AWS::CodeCommit::Repository
  Properties:
    RepositoryName: MyDemoRepo
    RepositoryDescription: This is a repository for my project with code from
 MySourceCodeBucket.
    Code:
      BranchName: development
      S3:
        Bucket: MySourceCodeBucket,
        Key: MyKey,
        ObjectVersion: 1
```

Weitere Beispiele finden Sie unter AWS::CodeCommit::Repository.

AWS CloudFormation,CodeCommit, und derAWS Cloud Development Kit (AWS CDK)

Repositorys, die mithilfe derAWS CDKAWS CloudFormation Use-Funktionalität bei ihrer Erstellung erstellt wurden. Wenn Sie verstehen, wieAWS CloudFormation Vorlagen mitCodeCommit

Ressourcen funktionieren, können Sie IhrenAWS CDK Code erstellen und verwalten. Weitere Informationen über dieAWS CDK finden Sie im AWS Cloud Development Kit (AWS CDK)Entwicklerhandbuch und in der AWS CDKAPI-Referenz.

Das folgendeAWS CDK Typescript-Beispiel erstellt einCodeCommit Repository namens *MyDemoRepo*. Das neu erstellte Repository wird mit Code gefüllt, der in einem Amazon S3 S3-Bucket gespeichert *MySourceCodeBucket*und in einer Verzweigung mit dem Namen *development* platziert ist.

```
import * as cdk from '@aws-cdk/core';
import codecommit = require('@aws-cdk/aws-codecommit');
export class CdkCodecommitStack extends cdk.Stack {
  constructor(scope: cdk.Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);
    // The code creates a CodeCommit repository with a default branch name development
      new codecommit.CfnRepository(this, 'MyRepoResource', {
            repositoryName: "MyDemoRepo",
            code: {
              "branchName": "development",
              "s3": {
                "bucket": "MySourceCodeBucket",
                "key": "MyKey"
              }
            },
        }
     );
  }
}
```

Weitere Informationen zu AWS CloudFormation

Weitere Informationen zu AWS CloudFormation finden Sie in den folgenden Ressourcen.

- AWS CloudFormation
- AWS CloudFormation-Benutzerhandbuch
- AWS CloudFormation-Benutzerhandbuch für die Befehlszeilenschnittstelle

Fehlerbehebung für AWS CodeCommit

Die folgenden Informationen helfen Ihnen möglicherweise bei der Lösung häufiger Probleme in AWS CodeCommit.

Themen

- Fehlerbehebung bei Git-Anmeldeinformationen und HTTPS-Verbindungen zuAWS CodeCommit
- Fehlerbehebung bei git-remote-codecommit und AWS CodeCommit
- Fehlerbehebung bei SSH-Verbindungen zuAWS CodeCommit
- Fehlerbehebung beim Helfer für Anmeldeinformationen und HTTPS-Verbindungen zuAWS CodeCommit
- Fehlerbehebung bei Git-Clients undAWS CodeCommit
- Fehlerbehebung bei Zugriffsfehlern undAWS CodeCommit
- Beheben von Konfigurationsfehlern undAWS CodeCommit
- Behebung von Konsolenfehlern und AWS CodeCommit
- Fehlerbehebung bei Auslösern undAWS CodeCommit
- Aktivieren des Debuggings

Fehlerbehebung bei Git-Anmeldeinformationen und HTTPS-Verbindungen zuAWS CodeCommit

Die folgenden Informationen können Ihnen dabei helfen, häufige Probleme bei der Verwendung von Git-Anmeldeinformationen und HTTPS zur Verbindung mit AWS CodeCommit-Repositorys zu beheben.

Themen

- Git-Anmeldeinformationen fürAWS CodeCommit: Bei der Verbindung zu meinem CodeCommit -Repository über Terminal oder Befehlszeile wird immer eine Eingabeaufforderung für Anmeldeinformationen angezeigt
- Git-Anmeldeinformationen fürAWS CodeCommit: Ich habe Git-Anmeldeinformationen eingerichtet, aber mein System nutzt diese nicht

Git-Anmeldeinformationen fürAWS CodeCommit: Bei der Verbindung zu meinem CodeCommit -Repository über Terminal oder Befehlszeile wird immer eine Eingabeaufforderung für Anmeldeinformationen angezeigt

Problem: Wenn Sie versuchen, über das Terminal oder die Befehlszeile einen Push, Pull oder eine andere Aktion in Bezug auf ein CodeCommit -Repository durchzuführen, werden Sie zur Eingabe eines Benutzernamens und eines Passworts aufgefordert. Darüber müssen Sie die Git-Anmeldeinformationen für Ihren IAM-Benutzer angeben.

Mögliche Lösungen: Die häufigsten Ursachen für diesen Fehler sind folgende: Auf Ihrem lokalen Computer wird möglicherweise ein Betriebssystem ausgeführt, das die Verwaltung von Anmeldeinformationen nicht unterstützt; oder auf dem Computer ist kein Dienstprogramm zur Verwaltung von Anmeldeinformationen installiert; oder die Git-Anmeldeinformationen für Ihren IAM-Anmeldeinformationen nicht in einem dieser Systeme zur Verwaltung von Anmeldeinformationen gespeichert wurden. Je nach Ihrem Betriebssystem und der lokalen Umgebung müssen Sie möglicherweise einen Anmeldeinformationsmanager installieren, den in Ihr Betriebssystem integrierten Anmeldeinformationsmanager konfigurieren oder Ihre lokale Umgebung so anpassen, dass Anmeldeinformationen gespeichert werden. Wenn auf Ihrem Computer zum Beispiel macOS ausgeführt wird, können Sie Ihre Anmeldeinformationen mithilfe des Schlüsselbund-Dienstprogramms (Keychain Access) speichern. Wenn Sie einen Windows-Computer nutzen, können Sie das Git-Dienstprogramm zur Verwaltung von Anmeldeinformationen (Git Credential Manager) verwenden, das zusammen mit Git für Windows installiert wird. Weitere Informationen finden Sie unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden und Credential Storage in der Git-Dokumentation.

Git-Anmeldeinformationen fürAWS CodeCommit: Ich habe Git-Anmeldeinformationen eingerichtet, aber mein System nutzt diese nicht

Problem: Wenn Sie versuchen, CodeCommit mit einem Git-Client zu verwenden, scheint der Client die Git-Anmeldeinformation nicht für den IAM-Benutzer zu verwenden.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler besteht darin, dass Sie zuvor die Verwendung des Hilfsprogramms für Anmeldeinformationen, das in derAWS CLIaus. Überprüfen Sie Ihre GITCONFIG-Datei auf Konfigurationsabschnitte, die dem folgenden ähneln, und entfernen Sie sie:

```
[credential "https://git-codecommit.*.amazonaws.com"]
```

```
    helper = !aws codecommit credential-helper $@
    UseHttpPath = true
```

Speichern Sie die Datei und öffnen Sie anschließend eine neue Befehlszeilen- oder Terminalsitzung, bevor Sie einen erneuten Verbindungsversuch unternehmen.

Möglicherweise wurden auch mehrere Hilfs- oder Verwaltungsprogramme für Anmeldeinformationen auf Ihrem Computer eingerichtet und Ihr System verwendet standardmäßig eine andere Konfiguration. Um die Einstellung zurückzusetzen, welches Hilfsprogramm für Anmeldeinformationen standardmäßig verwendet wird, können Sie die Option --system anstelle von --global oder --local verwenden, wenn Sie den Befehl git config ausführen.

Weitere Informationen finden Sie unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden und Credential Storage in der Git-Dokumentation.

Fehlerbehebung bei git-remote-codecommit und AWS CodeCommit

Die folgenden Informationen können Ihnen helfen, Probleme mit git-remote-codecommit zu beheben, die bei der Verbindung mit AWS CodeCommit-Repositorys auftreten können.

Themen

- Ich sehe einen Fehler: git: 'remote-codecommit' ist kein Git-Befehl
- Ich sehe einen Fehler: fatal: Remote-Helfer für 'codecommit' konnte nicht gefunden werden
- Klonfehler: Ich kann kein CodeCommit----Repository von einer IDE klonen
- Push oder Pull-Fehler: Ich kann keine Commits per Push oder Pull von einer IDE in ein CodeCommit---Repository verschieben

Ich sehe einen Fehler: git: 'remote-codecommit' ist kein Git-Befehl

Problem: Wenn Sie versuchen, git-remote-codecommit zu verwenden, wird ein Fehler angezeigt, dass git-remote-codecommit kein Git-Befehl ist. Siehe 'git —help'“.

Mögliche Lösungen: Der häufigste Grund für diesen Fehlerist, dass Sie entweder die ausführbare Datei git-remote-codecommit nicht zu Ihrem PATH hinzugefügt haben oder dass die Zeichenfolge enthältein Syntaxfehler. Dies kann passieren, wenn ein Bindestrich zwischen git und remote-codecommit fehlt oder wenn ein zusätzlicher Git vor git-remote-codecommit platziert wird.

Weitere Informationen zum Einrichten und Verwenden von git-remote-codecomfinden Sie unterEinrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommitaus.

Ich sehe einen Fehler: fatal: Remote-Helfer für 'codecommit' konnte nicht gefunden werden

Problem: Wenn Sie versuchen, git-remote-codecommit zu verwenden, wird eine Fehlermeldung angezeigt, die „fatal: Der Remote-Helfer für 'codecommit' konnte nicht gefunden werden.

Mögliche Lösungen: Die häufigsten Gründe für diesen Fehler sind:

- Das Setup ist für git-remote-codecommit nicht abgeschlossen
- Sie haben git-remote-codecommit an einem Speicherort installiert, der sich nicht in Ihrem Pfad befindet oder nicht als Teil des`Path`Umgebungsvariable
- Python ist nicht in Ihrem Pfad oder ist nicht als Teil des`Path`Umgebungsvariable
- Sie verwenden ein Terminal- oder Befehlszeilenfenster, das seit Abschluss der Installation von git-remote-codecommit nicht neu gestartet wurde

Weitere Informationen zum Einrichten und Verwenden von git-remote-codecomfinden Sie unterEinrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommitaus.

Klonfehler: Ich kann kein CodeCommit----Repository von einer IDE klonen

Problem: Wenn Sie versuchen, ein CodeCommit----Repository in einer IDE zu klonen, wird eine Fehlermeldung angezeigt, die besagt, dass der Endpunkt oder die URL ungültig ist.

Mögliche Lösungen: Nicht alle IDEs unterstützen die URL, die vongit-remote-codecommitwährend des Klonens. Klonen Sie das Repository lokal über das Terminal oder die Befehlszeile, und fügen Sie Ihrer IDE das lokale Repo hinzu. Weitere Informationen finden Sie unter Schritt 3: Stellen Sie eine Verbindung zum herCodeCommitkonsole und klone das Repository .

Push oder Pull-Fehler: Ich kann keine Commits per Push oder Pull von einer IDE in ein CodeCommit---Repository verschieben

Problem: Wenn Sie versuchen, Code per Push oder Pull von einer IDE zu verschieben, wird ein Verbindungsfehler angezeigt.

Mögliche Lösungen: Der häufigste Grund für diesen Fehler ist, dass die IDE nicht mit Git-Remote-Helfern wie kompatibel istgit-remote-codecommitaus. Anstatt die IDE-Funktionalität zu verwenden, um Commit, Push und Pull für Code durchzuführen, aktualisieren Sie das lokale Repo manuell über die Befehlszeile oder das Terminal mithilfe von Git-Befehlen.

Weitere Informationen zu Remote-Helfern und Git finden Sie in der Git-Dokumentation.

Fehlerbehebung bei SSH-Verbindungen zuAWS CodeCommit

Die folgenden Informationen können Ihnen dabei helfen, häufige Probleme bei der Verwendung von SSH zur Verbindung mit CodeCommit-Repositorys zu beheben.

Themen

- Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen, aber die Verbindung schlägt auf Linux-, macOS- oder Unix-Systemen fehl
- Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen und SSH wurde erfolgreich getestet, aber die Verbindung schlägt auf Windows-Systemen fehl
- Authentifizierungsproblem: Bei der Verbindung mit einem CodeCommit-Repository kann die Authentizität des Hosts nicht festgestellt werden
- IAM-Fehler: „Ungültiges Format“ beim Versuch, IAM einen öffentlichen Schlüssel hinzuzufügen
- Ich muss darauf zugreifenCodeCommitRepositorys in mehreren Amazon Web Services-Konten mit SSH-Anmeldeinformationen
- Git unter Windows: Bash-Emulator oder Befehlszeilenfenster stürzt bei Verbindungsversuchen mit SSH ab
- Das Public-Key-Format erfordert in einigen Linux-Distributionen eine Spezifikation
- Zugriffsfehler: Der öffentliche SSH-Schlüssel wurde verweigert, wenn eine Verbindung zu einem hergestellt wurdeCodeCommitEndlager

Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen, aber die Verbindung schlägt auf Linux-, macOS- oder Unix-Systemen fehl

Problem:Wenn Sie versuchen, eine Verbindung zu einem SSH-Endpunkt herzustellen, um mit einem zu kommunizierenCodeCommitRepository, entweder beim Testen der Verbindung oder beim Klonen eines Repositorys schlägt die Verbindung fehl oder wird abgelehnt.

Mögliche Korrekturen:Die SSH-Schlüssel-ID, die Ihrem öffentlichen Schlüssel in IAM zugewiesen wurde, ist möglicherweise nicht mit Ihrem Verbindungsversuch verknüpft. Möglicherweise haben Sie keine Konfigurationsdatei konfiguriert, Sie haben möglicherweise keinen Zugriff auf die Konfigurationsdatei, eine andere Einstellung verhindert möglicherweise ein erfolgreiches Lesen der Konfigurationsdatei, Sie haben möglicherweise die falsche Schlüssel-ID angegeben, oder Sie haben anstelle der Schlüssel-ID die ID des IAM-Benutzers angegeben.

Die SSH-Schlüssel-ID finden Sie in der IAM-Konsole im Profil Ihres IAM-Benutzers:

> **Note**
>
> Wenn Sie mehr als eine SSH-Schlüssel-IDs hochgeladen haben, werden die Schlüssel alphabetisch nach Schlüssel-ID und nicht nach dem Datum des Hochladens aufgeführt. Stellen Sie sicher, dass Sie die Schlüssel-ID kopiert haben, die dem richtigen Upload-Datum zugeordnet ist.

Testen Sie versuchsweise die Verbindung mit dem folgenden Befehl:

```
ssh Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com
```

Wenn nach der Bestätigung der Verbindung eine Erfolgsmeldung angezeigt wird, ist Ihre SSH-Schlüssel-ID gültig. Bearbeiten Sie Ihre Konfigurationsdatei, um Ihre Verbindungsversuche mit Ihrem öffentlichen Schlüssel in IAM zu verknüpfen. Wenn Sie die Config-Datei nicht bearbeiten möchten, können Sie allen Verbindungsversuchen mit Ihrem Repository Ihre SSH-Schlüssel-ID voranstellen. Zum Beispiel, wenn Sie ein Repository mit dem Namen klonen möchten*MyDemoRepo*Ohne Ihre Konfigurationsdatei so zu ändern, dass sie Ihre Verbindungsversuche verknüpft, würden Sie den folgenden Befehl ausführen:

```
git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/
repos/MyDemoRepo my-demo-repo
```

Weitere Informationen finden Sie unter Für SSH-Verbindungen unter Linux, macOS oder Unix.

Zugriffsfehler: Der öffentliche Schlüssel wurde erfolgreich in IAM hochgeladen und SSH wurde erfolgreich getestet, aber die Verbindung schlägt auf Windows-Systemen fehl

Problem:Wenn Sie versuchen, einen SSH-Endpunkt zum Klonen oder Kommunizieren mit einem zu verwendenCodeCommitRepository, es erscheint eine Fehlermeldung mit dem folgenden SatzNo `supported authentication methods available`.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist das Vorhandensein einer Windows-Systemumgebungsvariablen, die Windows anweist, ein anderes Programm zu verwenden, wenn Sie SSH zu verwenden versuchen. Möglicherweise verweist auf Ihrem System z. B. die Variable GIT_SSH auf eines der PuTTY-Tools (plink.exe). Dabei kann es sich um eine veraltete Konfiguration handeln oder aber auch um eine notwendige Einstellung für ein oder mehrere Programme auf Ihrem Computer. Wenn Sie sicher sind, dass diese Umgebungsvariable nicht benötigt wird, können Sie sie entfernen. Öffnen Sie dazu die Systemeigenschaften.

Um dieses Problem zu umgehen, öffnen Sie einen Bash-Emulator und versuchen Sie erneut, die SSH-Verbindung herzustellen. Geben Sie dabei jedoch `GIT_SSH_COMMAND="SSH"` als Präfix an. Beispiel zum Klonen eines Repositorys mit SSH:

```
GIT_SSH_COMMAND="ssh" git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/
MyDemoRepo my-demo-repo
```

Ein ähnliches Problem kann auftreten, wenn bei Ihrer Windows-Version bei Verbindungen mithilfe von SSH über die Windows-Befehlszeile die SSH-Schlüssel-ID als Teil der Verbindungszeichenfolge angegeben werden muss. Versuchen Sie erneut, Ihre Verbindung herzustellen, diesmal mit der SSH-Schlüssel-ID, die als Teil des Befehls aus IAM kopiert wurde. Beispiele:

```
git clone ssh://Your-SSH-Key-ID@git-codecommit.us-east-2.amazonaws.com/v1/repos/
MyDemoRepo my-demo-repo
```

Authentifizierungsproblem: Bei der Verbindung mit einem CodeCommit-Repository kann die Authentizität des Hosts nicht festgestellt werden

Problem:Wenn Sie versuchen, einen SSH-Endpunkt für die Kommunikation mit einem zu verwendenCodeCommitRepository wird eine Warnmeldung mit dem folgenden Satz angezeigtThe `authenticity of host 'host-name' can't be established.`

Mögliche Lösungen: Ihre Anmeldeinformationen sind möglicherweise nicht richtig eingerichtet. Befolgen Sie die Anweisungen unter Für SSH-Verbindungen unter Linux, macOS oder Unix oder Für SSH-Verbindungen unter Windows.

Wenn Sie diese Schritte befolgt haben und das Problem weiterhin besteht, versucht möglicherweise jemandman-in-the-middleAngriff. Wenn die folgende Meldung angezeigt wird, geben Sie no ein und drücken Sie die Eingabetaste:

```
Are you sure you want to continue connecting (yes/no)?
```

Stellen Sie sicher, dass der Fingerprint und der öffentliche Schlüssel für CodeCommit-Verbindungen mit denen übereinstimmen, die in den SSH-Einrichtungsthemen dokumentiert sind, bevor Sie mit dem Verbindungsaufbau fortfahren.

Öffentliche Fingerabdrücke fürCodeCommit

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.us-east-2.amazonaws.com	MD5	`a9:6d:03:ed:08:42:21:be:06:e1:e0:2a:d1:75:31:5e`
git-codecommit.us-east-2.amazonaws.com	SHA256	`3lBlW2g5xn/NA2Ck6dyeJIrQOWvn7n8UEs56fG6ZIzQ`
git-codecommit.us-east-1.amazonaws.com	MD5	`a6:9c:7d:bc:35:f5:d4:5f:8b:ba:6f:c8:bc:d4:83:84`
git-codecommit.us-east-1.amazonaws.com	SHA256	`eLMY1j0DKA4uvDZcl/KgtIayZANwX6t8+8isPtotBoY`
git-codecommit.us-west-2.amazonaws.com	MD5	`a8:68:53:e3:99:ac:6e:d7:04:7e:f7:92:95:77:a9:77`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.us-west-2.amazonaws.com	SHA256	`0pJx9SQpkbPUAHwy58UVIq0IHcyo1fwCpOOuVgcAWPo`
git-codecommit.eu-west-1.amazonaws.com	MD5	`93:42:36:ea:22:1f:f1:0f:20:02:4a:79:ff:ea:12:1d`
git-codecommit.eu-west-1.amazonaws.com	SHA256	`tKjRkOL8dmJyTmSbeSdN1S8F/f0iql3RlvqgTOP1UyQ`
git-codecommit.ap-northeast-1.amazonaws.com	MD5	`8e:a3:f0:80:98:48:1c:5c:6f:59:db:a7:8f:6e:c6:cb`
git-codecommit.ap-northeast-1.amazonaws.com	SHA256	`Xk/WeYD/K/bnBybzhiuu4dWpBJtXPf7E30jHU7se4Ow`
git-codecommit.ap-southeast-1.amazonaws.com	MD5	`65:e5:27:c3:09:68:0d:8e:b7:6d:94:25:80:3e:93:cf`
git-codecommit.ap-southeast-1.amazonaws.com	SHA256	`ZIsVa7OVzxrTIf+Rk4UbhPv6Es22mSB3uTBojfPXIno`
git-codecommit.ap-southeast-2.amazonaws.com	MD5	`7b:d2:c1:24:e6:91:a5:7b:fa:c1:0c:35:95:87:da:a0`
git-codecommit.ap-southeast-2.amazonaws.com	SHA256	`nYp+gHas80HY3DqbP4yanCDFhqDVjseefVbHEXqH2Ec`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.ap-southeast-3.amazonaws.com	MD5	`64:d9:e0:53:19:4f:a8:91:9a:c3:53:22:a6:a8:ed:a6`
git-codecommit.ap-southeast-3.amazonaws.com	SHA256	`ATdkGSFhpqIu7RqUVT/1RZo6MLxxxUW9NoDVMbAc/6g`
git-codecommit.me-central-1.amazonaws.com	MD5	`bd:fa:e2:f9:05:84:d6:39:6f:bc:d6:8d:fe:de:61:76`
git-codecommit.me-central-1.amazonaws.com	SHA256	`grceUDWubo4MzG1NoaKZKUfrgPvfN3ijliOnQr1lTZA`
git-codecommit.eu-central-1.amazonaws.com	MD5	`74:5a:e8:02:fc:b2:9c:06:10:b4:78:84:65:94:22:2d`
git-codecommit.eu-central-1.amazonaws.com	SHA256	`MwGrkiEki8QkkBtlAgXbYt0hoZYBnZF62VY5RzGJEUY`
git-codecommit.ap-northeast-2.amazonaws.com	MD5	`9f:68:48:9b:5f:fc:96:69:39:45:58:87:95:b3:69:ed`
git-codecommit.ap-northeast-2.amazonaws.com	SHA256	`eegAPQrWY9YsYo9ZHIKOmxetfXBHzAZd8Eya53Qcwko`
git-codecommit.sa-east-1.amazonaws.com	MD5	`74:99:9d:ff:2b:ef:63:c6:4b:b4:6a:7f:62:c5:4b:51`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.sa-east-1.amazonaws.com	SHA256	kW+VKB0jpRaG/ZbXkgbtMQbKgEDK7JnISV3SVoyCmzU
git-codecommit.us-west-1.amazonaws.com	MD5	3b:76:18:83:13:2c:f8:eb:e9:a3:d0:51:10:32:e7:d1
git-codecommit.us-west-1.amazonaws.com	SHA256	gzauWTWXDK2u5KuMMi5vbKTmfyerdIwgSbzYBODLpzg
git-codecommit.eu-west-2.amazonaws.com	MD5	a5:65:a6:b1:84:02:b1:95:43:f9:0e:de:dd:ed:61:d3
git-codecommit.eu-west-2.amazonaws.com	SHA256	r0Rwz5k/IHp/QyrRnfiM9j02D5UEqMbtFNTuDG2hNbs
git-codecommit.ap-south-1.amazonaws.com	MD5	da:41:1e:07:3b:9e:76:a0:c5:1e:64:88:03:69:86:21
git-codecommit.ap-south-1.amazonaws.com	SHA256	hUKwnTj7+Xpx4Kddb6p45j4RazIJ4IhAMD8k29itOfE
git-codecommit.ap-south-2.amazonaws.com	MD5	bc:cc:9f:15:f8:f3:58:a2:68:65:21:e2:23:71:8d:ce
git-codecommit.ap-south-2.amazonaws.com	SHA256	Xe0CyZEOvgR5Xa2YUGqf+jn8/Ut7l7nX/CmslSFNEig

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.ca-central-1.amazonaws.com	MD5	`9f:7c:a2:2f:8c:b5:74:fd:ab:b7:e1:fd:af:46:ed:23`
git-codecommit.ca-central-1.amazonaws.com	SHA256	`Qz5puafQdANVprLlj6r0Qyh4lCNsF6ob61dGcPtFS7w`
git-codecommit.eu-west-3.amazonaws.com	MD5	`1b:7f:97:dd:d7:76:8a:32:2c:bd:2c:7b:33:74:6a:76`
git-codecommit.eu-west-3.amazonaws.com	SHA256	`uw7c2FL564jVoFgtc+ikzILnKBsZz7t9+CFdSJjKbLI`
Git-Code-Commit.us-gov-west-1.amazonaws.com	MD5	`9f:6c:19:3b:88:cd:e8:88:1b:9c:98:6a:95:31:8a:69`
Git-Code-Commit.us-gov-west-1.amazonaws.com	SHA256	`djXQoSIFcg8vHe0KVH1xW/gOF9X37tWTqu4Hkng75x4`
Git-Code-Commit.us-gov-east-1.amazonaws.com	MD5	`00:8d:b5:55:6f:05:78:05:ed:ea:cb:3f:e6:f0:62:f2`
Git-Code-Commit.us-gov-east-1.amazonaws.com	SHA256	`fVb+R0z7qW7minenW+rUpAABRCRBTCzmETAJEQrg98`
git-codecommit.eu-north-1.amazonaws.com	MD5	`8e:53:d8:59:35:88:82:fd:73:4b:60:8a:50:70:38:f4`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.eu-north-1.amazonaws.com	SHA256	`b6KSK7xKq+V8jl7iuAcjqXsG7zkqoUZZmmhYYFBq1wQ`
git-codecommit.me-south-1.amazonaws.com	MD5	`0e:39:28:56:d5:41:e6:8d:fa:81:45:37:fb:f3:cd:f7`
git-codecommit.me-south-1.amazonaws.com	SHA256	`O+NToCGgjrHekiBuOl0ad7ROGEsz+DBLXOd/c9wc0JU`
git-codecommit.ap-east-1.amazonaws.com	MD5	`a8:00:3d:24:52:9d:61:0e:f6:e3:88:c8:96:01:1c:fe`
git-codecommit.ap-east-1.amazonaws.com	SHA256	`LafadYwUYW8hONoTRpojbjNs9IRnbEwHtezD3aAIBX0`
git-codecommit.cn-north-1.amazonaws.com .cn	MD5	`11:7e:2d:74:9e:3b:94:a2:69:14:75:6f:5e:22:3b:b3`
git-codecommit.cn-north-1.amazonaws.com .cn	SHA256	`IYUXxH2OpTDsyYMLIp+JY8CTLS4UX+ZC5JVZXPRaxc8`
git-codecommit.cn-northwest-1.amazonaws.com .cn	MD5	`2e:a7:fb:4c:33:ac:6c:f9:aa:f2:bc:fb:0a:7b:1e:b6`
git-codecommit.cn-northwest-1.amazonaws.com .cn	SHA256	`wqjd6eHd0+mOBx+dCNuL0omUoCNjaDtZiEpWj5TmCfQ`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.eu-south-1.amazonaws.com	MD5	`b9:f6:5d:e2:48:92:3f:a9:37:1e:c4:d0:32:0e:fb:11`
git-codecommit.eu-south-1.amazonaws.com	SHA256	`lyXrWbCg3uQmJrl1XxB/ASR7ugW1Ysf5yzYOJbudHsI`
git-codecommit.ap-northeast-3.amazonaws.com	MD5	`25:17:40:da:b9:d4:18:c3:b6:b3:fb:ed:1c:20:fe:29`
git-codecommit.ap-northeast-3.amazonaws.com	SHA256	`2B815B9F0AvwLnRxSVxUz4kDYmtEQUGGdQYP8OQLXhA`
git-codecommit.af-south-1.amazonaws.com	MD5	`21:a0:ba:d7:c1:d1:b5:39:98:8d:4d:7c:96:f5:ca:29`
git-codecommit.af-south-1.amazonaws.com	SHA256	`C34ji3x/cnsDZjUpyNGXdE5pjHYimqJrQZ3leTgqJHM`
git-codecommit.il-central-1.amazonaws.com	MD5	`04:74:89:16:98:7a:61:b1:69:46:42:3c:d1:b4:ac:a9`
git-codecommit.il-central-1.amazonaws.com	SHA256	`uFxhp51kUWhleTLeYbxQVYm4RnNLNZ5Dbdm1cgdSl/8`

IAM-Fehler: „Ungültiges Format“ beim Versuch, IAM einen öffentlichen Schlüssel hinzuzufügen

Problem:In IAM, wenn Sie versuchen, die Verwendung von SSH einzurichten mitCodeCommitwird eine Fehlermeldung mit dem folgenden Satz angezeigt`Invalid formatwenn` Sie versuchen, Ihren öffentlichen Schlüssel hinzuzufügen.

Mögliche Korrekturen:IAM erfordert, dass der öffentliche Schlüssel im ssh-rsa-Format oder PEM-Format codiert sein muss. Es akzeptiert nur öffentliche Schlüssel im OpenSSH-Format und hat zusätzliche Anforderungen, wie unter beschriebenVerwenden Sie SSH-Schlüssel mitCodeCommitin derIAM-Benutzerhandbuch. Wenn Sie den öffentlichen Schlüssel in einem anderen Format bereitstellen oder wenn der Schlüssel nicht die erforderliche Bitzahl enthält, wird diese Fehlermeldung angezeigt.

- Beim Kopieren der öffentlichen SSH-Schlüssel wurden durch Ihr Betriebssystem möglicherweise Zeilenumbrüche eingeführt. Stellen Sie sicher, dass der öffentliche Schlüssel, den Sie zu IAM hinzufügen, keine Zeilenumbrüche enthält.
- Einige Windows-Betriebssysteme verwenden das OpenSSH-Format nicht. Informationen zum Generieren eines Schlüsselpaars und zum Kopieren des von IAM benötigten OpenSSH-Formats finden Sie unterthe section called “Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein”.

Weitere Informationen zu den Anforderungen für SSH-Schlüssel in IAM finden Sie unterVerwenden Sie SSH-Schlüssel mitCodeCommitin derIAM-Benutzerhandbuch.

Ich muss darauf zugreifenCodeCommitRepositorys in mehreren Amazon Web Services-Konten mit SSH-Anmeldeinformationen

Problem:Ich möchte den SSH-Zugang einrichten fürCodeCommitRepositorys in mehr als einem Amazon Web Services-Konto.

Mögliche Korrekturen:Sie können eindeutige öffentliche/private SSH-Schlüsselpaare für jedes Amazon Web Services-Konto erstellen und IAM mit jedem Schlüssel konfigurieren. Anschließend können Sie Ihre Datei ~/.ssh/config mit Informationen zu jeder IAM-Benutzer-ID konfigurieren, die dem öffentlichen Schlüssel zugeordnet ist. Beispiele:

```
Host codecommit-1
```

```
    Hostname git-codecommit.us-east-1.amazonaws.com
    User SSH-KEY-ID-1 # This is the SSH Key ID you copied from IAM in Amazon Web
 Services account 1 (for example, APKAEIBAERJR2EXAMPLE1).
    IdentityFile ~/.ssh/codecommit_rsa # This is the path to the associated public key
 file, such as id_rsa.  We advise creating CodeCommit specific _rsa files.

Host codecommit-2
    Hostname git-codecommit.us-east-1.amazonaws.com
    User SSH-KEY-ID-2 # This is the SSH Key ID you copied from IAM in Amazon Web
 Services account 2 (for example, APKAEIBAERJR2EXAMPLE2).
    IdentityFile ~/.ssh/codecommit_2_rsa # This is the path to the other associated
 public key file.  We advise creating CodeCommit specific _rsa files.
```

In dieser Konfiguration können Sie 'git-codecommit.us-east-1.amazonaws.com' durch 'codecommit-2' ersetzen. Um beispielsweise ein Repository in Ihrem zweiten Amazon Web Services-Konto zu klonen:

```
git clone ssh://codecommit-2/v1/repos/YourRepositoryName
```

Um ein Remote für Ihr Repository einzurichten, führen Sie folgenden Befehl ausgit remote add. Beispiele:

```
git remote add origin ssh://codecommit-2/v1/repos/YourRepositoryName
```

Weitere Beispiele finden Sie unterdieser Forumsbeitragunddieser Beitrag amGitHub.

Git unter Windows: Bash-Emulator oder Befehlszeilenfenster stürzt bei Verbindungsversuchen mit SSH ab

Problem: Nachdem Sie SSH-Zugriff für Windows konfiguriert und Konnektivität im Befehlszeilen- oder Terminalfenster überprüft haben, wird eine Meldung angezeigt, dass der Host-Schlüssel des Servers nicht in der Registry zwischengespeichert wurde. Außerdem reagiert die Eingabeaufforderung zum Speichern des Schlüssels im Cache nicht (y/n/Eingabetaste wird nicht akzeptiert), wenn Sie versuchen, Befehle wie zum Beispiel git pull, git push oder git clone an der Eingabeauffoderung oder im Bash-Emulator zu verwenden.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass Ihre Git-Umgebung zur Verwendung einer anderen Authentifizierungsmethode als OpenSSH konfiguriert ist (möglicherweise PuTTY). Es ist bekannt, dass dies in einigen Konfigurationen Probleme mit der Zwischenspeicherung

von Schlüsseln verursachen kann. Um dieses Problem zu beheben, versuchen Sie es mit einer der folgenden Methoden:

- Öffnen Sie einen Bash-Emulator und fügen Sie den Parameter `GIT_SSH_COMMAND="ssh"` vor dem Git-Befehl hinzu. Wenn Sie zum Beispiel versuchen, ein Repository per Push zu überragen, geben Sie anstelle von git push den folgenden Befehl ein:

  ```
  GIT_SSH_COMMAND="ssh" git push
  ```

- Wenn Sie PuTTY installiert haben, öffnen Sie PuTTY und inHostname (oder IP-Adresse), geben Sie denCodeCommitEndpunkt, den Sie erreichen möchten (z. B. git-codecommit.us-east-2.amazonaws.com). Klicken Sie auf Open. Wenn Sie eine Eingabeaufforderung von PuTTY Security Alert erhalten, wählen Sie Yes (Ja), um den Schlüssel dauerhaft zwischenzuspeichern.
- Löschen Sie die Umgebungsvariable `GIT_SSH` oder benennen Sie sie um, wenn Sie sie nicht mehr verwenden. Öffnen Sie anschließend ein neues Befehlszeilenfenster oder eine neue Bash-Emulator-Sitzung und versuchen Sie erneut, den Befehl einzugeben.

Weitere Lösungen finden Sie unter Git clone/pull continually freezing at Store key in cache auf Stack Overflow.

Das Public-Key-Format erfordert in einigen Linux-Distributionen eine Spezifikation

Problem:Wenn Sie versuchen, ein öffentlich-privates Schlüsselpaar zu konfigurieren, erhalten Sie eine Fehlermeldung.

Mögliche Korrekturen:Einige Linux-Distributionen erfordern eine zusätzliche Konfigurationszeile im`~/.ssh/config`Datei, die die akzeptierten Typen von öffentlichen Schlüsseln spezifiziert. Weitere Informationen finden Sie in der Dokumentation zu Ihrer Distribution über`PubkeyAcceptedKeyTypes`.

Zugriffsfehler: Der öffentliche SSH-Schlüssel wurde verweigert, wenn eine Verbindung zu einem hergestellt wurdeCodeCommitEndlager

Problem:Wenn Sie versuchen, einen SSH-Endpunkt für die Kommunikation mit einem zu verwendenCodeCommitRepository, es erscheint eine Fehlermeldung mit dem folgenden Satz`Error: public key denied`.

Mögliche Fehlerbehebungen: Der häufigste Grund für diesen Fehler ist, dass Sie die Einrichtung für die SSH-Verbindungen nicht abgeschlossen haben. Konfigurieren Sie ein öffentliches und ein privates SSH-Schlüsselpaar und ordnen Sie den öffentlichen Schlüssel dann Ihrem IAM-Benutzer zu. Weitere Informationen zur Konfiguration von SSH finden Sie unter Für SSH-Verbindungen unter Linux, macOS oder Unix und Für SSH-Verbindungen unter Windows.

Fehlerbehebung beim Helfer für Anmeldeinformationen und HTTPS-Verbindungen zuAWS CodeCommit

Die folgenden Informationen können Ihnen dabei helfen, häufige Probleme bei der Verwendung des Hilfsprogramms für Anmeldeinformationen, das in der enthalten istAWS CLIund HTTPS, um sich mit CodeCommit-Repositorys zu verbinden.

> **Note**
>
> Obwohl das Hilfsprogramm für Anmeldeinformationen eine unterstützte Methode für die Verbindung mit CodeCommit über den Verbundzugriff, über einen Identitätsanbieter oder mit temporären Anmeldeinformationen ist, empfiehlt sich die Installation und Verwendung desgit-remote-codecommitDienstprogramm. Weitere Informationen finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit .

Themen

- Ich erhalte eine Fehlermeldung beim Ausführen desgit configBefehl zum Konfigurieren des Helfers für Anmeldeinformationen
- Zurückgegebene Fehlermeldung "Befehl nicht gefunden" unter Windows bei Verwendung des Hilfsprogramms für Anmeldeinformationen
- Wenn ich eine Verbindung mit einem CodeCommit-Repository verbinde, werde ich zur Eingabe eines Benutzernamens aufgefordert
- Git für macOS: Ich habe den Hilfsprogramm für Anmeldeinformationen erfolgreich konfiguriert, aber jetzt wird mir der Zugriff auf mein Repository verweigert (403)
- Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert (403)

Ich erhalte eine Fehlermeldung beim Ausführen des`git config`Befehl zum Konfigurieren des Helfers für Anmeldeinformationen

Problem: Wenn Sie versuchen, den Befehl git config auszuführen, um den Helfer für Anmeldeinformationen für die Kommunikation mit einem CodeCommit-Repository zu konfigurieren, wird ein Fehler angezeigt, dass zu wenige Argumente vorliegen, oder eine Eingabeaufforderung, die Git-Konfigurationsbefehle und Syntax vorschlägt.

Mögliche Lösungen: Der häufigste Grund für diesen Fehler ist, dass entweder einfache Anführungszeichen für den Befehl unter einem Windows-Betriebssystem verwendet werden oder doppelte Anführungszeichen für den Befehl in einem Linux-, macOS- oder Unix-Betriebssystem verwendet werden. Die richtige Syntax ist wie folgt:

- Windows: `git config --global credential.helper "!aws codecommit credential-helper $@"`
- Linux, macOS oder Unix:`git config --global credential.helper '!aws codecommit credential-helper $@'`

Zurückgegebene Fehlermeldung "Befehl nicht gefunden" unter Windows bei Verwendung des Hilfsprogramms für Anmeldeinformationen

Problem: Nach dem Aktualisieren desAWSCLI, Hilfsverbindungen für Anmeldeinformationsprogramme mit CodeCommit-Repositorys schlagen mit fehl`aws codecommit credential-helper $@ get: aws: command not found`aus.

Ursache: Der häufigste Grund für diesen Fehler ist, dass SieAWSDie CLI-Version wurde auf eine Version aktualisiert, die Python 3 verwendet. Dies ist ein bekanntes Problem bei dem MSI-Paket. Um festzustellen, ob Sie über eine der betroffenen Versionen verfügen, öffnen Sie eine Befehlszeile und führen Sie den folgenden Befehl aus: `aws --version`

Wenn die Python-Version in der Ausgabe mit 3 beginnt, ist Ihre Version betroffen. Beispiel:

```
aws-cli/1.16.62 Python/3.6.2 Darwin/16.7.0 botocore/1.12.52
```

Mögliche Lösungen: Sie können dieses Problem mit einer der folgenden Möglichkeiten umgehen:

- Installieren und konfigurieren Sie die AWS CLI auf Windows unter Verwendung von Python und pip anstelle des MSI. Weitere Informationen finden Sie unter Installieren von Python, pip und der AWS CLI unter Windows.
- Bearbeiten Sie Ihre `.gitconfig`-Datei manuell, um den Abschnitt `[credential]` so zu ändern, dass er explizit auf `aws.cmd` auf Ihrem lokalen Computer verweist. Beispiel:

```
[credential]
    helper = !"\C:\\Program Files\\Amazon\\AWSCLI\\bin\\aws.cmd\" codecommit
 credential-helper $@
    UseHttpPath = true
```

- Führen Sie den Befehl git config aus, um Ihre `.gitconfig`-Datei so zu konfigurieren, dass sie explizit auf `aws.cmd` verweist. Aktualisieren Sie dann manuell die PATH-Umgebungsvariable nach Bedarf mit dem Pfad zu dem Befehl. Beispiel:

```
git config --global credential.helper "!aws.cmd codecommit credential-helper $@"
git config --global credential.UseHttpPath true
```

Wenn ich eine Verbindung mit einem CodeCommit-Repository verbinde, werde ich zur Eingabe eines Benutzernamens aufgefordert

Problem: Wenn Sie versuchen, mithilfe des Hilfsprogramms für Anmeldeinformationen mit einem CodeCommit-Repository zu kommunizieren, werden Sie in einer Meldung aufgefordert, Ihren Benutzernamen einzugeben.

Mögliche Lösungen: KonfigurierenAWSProfil oder stellen Sie sicher, dass Sie das Profil verwenden, das Sie für die Arbeit mit CodeCommit konfiguriert haben. Weitere Informationen zur Einrichtung erhalten Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitRepositorys auf Linux, macOS oder Unix mit demAWS CLIHelfer für Anmeldeinformationen oder Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitRepositorien unter Windows mit demAWS CLIHelfer für Anmeldeinformationen. Weitere Informationen über IAM, Zugriffsschlüssel und geheime Schlüssel finden Sie unterVerwalten der Zugriffsschlüssel für IAM-BenutzerundWie erhalte ich Anmeldeinformationen?

Git für macOS: Ich habe den Hilfsprogramm für Anmeldeinformationen erfolgreich konfiguriert, aber jetzt wird mir der Zugriff auf mein Repository verweigert (403)

Problem: Für macOS greift das Hilfsprogramm für Anmeldeinformationen nicht wie erwartet auf Ihre Anmeldeinformationen zu bzw. verwendet diese nicht wie erwartet. Dies kann durch zwei verschiedene Probleme verursacht werden:

- DieAWS CLIist konfiguriert für einAWS-Regionanders als die, in der sich das Repository befindet.
- Das Schlüsselbund-Dienstprogramm (Keychain Access) hat Anmeldeinformationen gespeichert, die jetzt nicht mehr gültig sind.

Mögliche Lösungen: So überprüfen Sie, ob derAWS CLIist für die richtige Region konfiguriert, führen Sie dieaws configureund überprüfen Sie die angezeigten Informationen. Wenn sich das CodeCommit-Repository in einemAWS-Regionanders als der für dieAWS CLI, musst du das führen Sie das aus:aws configurebefehlen und ändern Sie die Werte in diejenigen, die für diese Region geeignet sind. Weitere Informationen finden Sie unter Schritt 1: Erstkonfiguration fürCodeCommit .

Die Standardversion von Git, die unter OS X und macOS veröffentlicht wurde, speichert generierte Anmeldeinformationen mithilfe des Schlüsselbund-Dienstprogramms (Keychain Access). Aus Sicherheitsgründen ist das für den Zugriff auf das -CodeCommit-Repository generierte Passwort vorübergehend, sodass die im Schlüsselbund gespeicherten Anmeldeinformationen nach ca. 15 Minuten nicht mehr gültig sind. Wenn Sie nur mit CodeCommit auf Git zugreifen, versuchen Sie folgende Methode:

1. Geben Sie am Terminal den Befehl git config ein, um die Git-Konfigurationsdatei (`gitconfig`) zu finden, in der das Schlüsselbund-Dienstprogramm definiert ist. Abhängig von Ihrem lokalen System und Ihren Einstellungen haben Sie möglicherweise mehr als eine `gitconfig`-Datei.

   ```
   git config -l --show-origin | grep credential
   ```

   In der Ausgabe dieses Befehls wirdSuche nach ähnlichen Ergebnissen:

   ```
   file:/path/to/gitconfig  credential.helper=osxkeychain
   ```

   Die am Anfang dieser Zeile aufgelistete Datei ist die Git-Konfigurationsdatei, die Sie bearbeiten müssen.

2. Verwenden Sie zum Bearbeiten der Git-Konfigurationsdatei einen Texteditor, oder führen Sie den folgenden Befehl aus:

```
nano /usr/local/git/etc/gitconfig
```

3. Ändern Sie die Konfiguration mit einer der folgenden Strategien:

   - Kommentieren oder löschen Sie den Abschnitt „Anmeldeinformationen“, der enthält`helper = osxkeychain`aus. Beispiel:

```
# helper = osxkeychain
```

   - Aktualisieren Sie beide das`aws credential helper`und`osxkeychain`Helferabschnitte für Anmeldeinformationen, um Kontext zu haben. Beispiel, wenn`osxkeychain`wird verwendet, um sich bei GitHub zu authentifizieren:

```
[credential "https://git-codecommit.us-east-1.amazonaws\.com"]
  helper = !aws --profile CodeCommitProfile codecommit credential-helper $@
  UseHttpPath = true
[credential "https://github.com"]
  helper = osxkeychain
```

     In dieser Konfiguration verwendet Git das`osxkeychain`Helfer, wenn der Remote-Host übereinstimmt“`https://github.com`„und der Helfer für Anmeldeinformationen, wenn der Remote-Host übereinstimmt“`https://git-codecommit\.us-east-1\.amazonaws.com`„.

   - Fügen Sie vor dem Helfer für Anmeldeinformationen einen leeren String-Helfer ein. Beispiel:

```
[credential]
  helper =
  helper = !aws --profile CodeCommitProfile codecommit credential-helper $@
  UseHttpPath = true
```

   Wenn Sie andernfalls das Schlüsselbund-Dienstprogramm weiterhin verwenden wollen, um Anmeldeinformationen für weitere Git-Repositorys zu cachen, modifizieren Sie die Kopfzeile, anstatt die Zeile auszukommentieren. Um beispielsweise zwischengespeicherte Anmeldeinformationen für GitHub zuzulassen, könnten Sie die Kopfzeile wie folgt modifizieren:

```
[credential "https://github.com"]
   helper = osxkeychain
```

Wenn Sie mit Git auf andere Repositorys zugreifen, können Sie das Schlüsselbund-Dienstprogramm so konfigurieren, dass es keine Anmeldeinformationen für Ihre CodeCommit-Repositorys bereitstellt. So konfigurieren Sie das Schlüsselbund-Dienstprogramm (Keychain Access):

1. Öffnen Sie das Schlüsselbund-Dienstprogramm. (Sie finden das Schlüsselbund-Dienstprogramm mit dem Finder.)
2. Search By`git-codecommit.`*`us-east-2`*`.amazonaws.com`und ersetzen*`us-east-2`*mit demAWS-Regionwo das Repository existiert. Markieren Sie die Zeile, öffnen Sie das Kontextmenü (klicken Sie mit der rechten Maustaste) und wählen Sie dann Get Info.
3. Wählen Sie die Registerkarte Access Control aus.
4. Wählen Sie unter Confirm before allowing access den Eintrag `git-credential-osxkeychain` aus und klicken Sie auf das Minuszeichen, um den Eintrag aus der Liste zu entfernen.

> **Note**
>
> Nachdem Sie `git-credential-osxkeychain` aus der Liste entfernt haben, wird bei jeder Ausführung eines Git-Befehls ein Dialogfeld eingeblendet. Klicken Sie zum Fortfahren auf Deny. Falls Sie diese Dialogfelder umgehen möchten, finden Sie hier einige Alternativen:
>
> - Connect Sie sich mit CodeCommit über SSHoder Git-AnmeldeinformationenINSTEAD OFHilfsprogramm für Anmeldeinformationen mitHTTPS. Weitere Informationen finden Sie unter Für SSH-Verbindungen unter Linux, macOS oder Unix und Einrichtung für HTTPS-Benutzer mit Git-Anmeldeinformationen.
> - Wählen Sie im Schlüsselbund-Dienstprogramm auf der Registerkarte Access Control für `git-codecommit.us-east-2.amazonaws.com` die Option Allow all applications to access this item (access to this item is not restricted) aus. Auf diese Weise werden keine Popups angezeigt, aber die Anmeldeinformationen laufen ab (nach durchschnittlich etwa 15 Minuten) und die Fehlermeldung 403 wird angezeigt. In diesem Fall müssen Sie das Schlüsselbundobjekt löschen, um die Funktionalität wiederherzustellen.

- Sie installieren eine Git-Version, in der nicht standardmäßig der Schlüsselbund verwendet wird.
- Ziehen Sie eine Skriptlösung zum Löschen des Schlüsselbundelements in Betracht. Ein von der Community generiertes Beispiel einer Skriptlösung finden Sie unter Mac OS X Script to Periodically Delete Cached Credentials in the OS X Certificate Store in Produkt- und Service-Integrationen.

Wenn Sie nicht möchten, dass Git das Schlüsselbund-Dienstprogramm verwendet, können Sie Git so konfigurieren, dass "osxkeychain" nicht mehr als Hilfsprogramm für die Anmeldeinformationen verwendet wird. Wenn Sie beispielsweise ein Terminal öffnen und den Befehl `git config --system credential.helper` ausführen und `osxkeychain` zurückgegeben wird, ist Git zur Verwendung des Schlüsselbund-Dienstprogramms eingestellt. Sie können dies ändern, indem Sie den folgenden Befehl ausführen:

```
git config --system --unset credential.helper
```

Beachten Sie, dass Sie diesen Befehl mit dem`--system`-Option ändert systemweit das Git-Verhalten für alle Benutzer und dies kann unerwartete Konsequenzen für andere Benutzer oder Repositorys haben, wenn Sie neben CodeCommit weitere Repository-Services verwenden. Außerdem ist für diesen Ansatz möglicherweise `sudo` erforderlich und Sie benötigen ausreichende Systemberechtigungen zum Anwenden dieser Änderung. Führen Sie den Befehl `git config --system credential.helper` erneut aus, um sicherzustellen, dass er korrekt ausgeführt wurde. Weitere Informationen finden Sie unter Customizing Git - Git Configuration und in diesem Artikel zu Stack Overflow.

Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert (403)

Problem: Unter Windows greift das Hilfsprogramm für Anmeldeinformationen nicht wie erwartet auf Ihre Anmeldeinformationen zu bzw. verwendet diese nicht wie erwartet. Dies kann durch verschiedene Probleme verursacht werden:

- DieAWS CLIist konfiguriert für einAWS-Regionanders als die, in der sich das Repository befindet.
- Standardmäßig installiert Git für Windows ein Git-Dienstprogramm für Anmeldeinformationen, das nicht mit CodeCommit-Verbindungen kompatibel ist, die dieAWSHilfsprogramm für

Anmeldeinformationen. Wenn diese Installation installiert ist, schlagen Verbindungen mit dem Repository fehl, obwohl das Hilfsprogramm für Anmeldeinformationen mit derAWS CLIund für Verbindungen mit CodeCommit konfiguriert.

- Einige Versionen von Git für Windows erfüllen möglicherweise nicht vollständig die in RFC 2617 und RFC 4559 festgeschriebenen Bedingungen. Dies kann zu Problemen mit Git-Anmeldeinformationen und dem in die AWS CLI integrierten Hilfsprogramm für Anmeldeinformationen führen. Weitere Informationen finden Sie unter Version 2.11.0(3) does not ask for username/password.

Mögliche Lösungen:

- Wenn Sie versuchen, das Hilfsprogramm für Anmeldeinformationen der AWS CLI zu verwenden, sollten Sie die Verbindung unter Umständen mit Git-Anmeldeinformationen über HTTPS und nicht mit dem Hilfsprogramm aufbauen. Die Git-Anmeldeinformationen, die für Ihren IAM-Benutzer konfiguriert sind, sind im Gegensatz zum Hilfsprogramm für Anmeldeinformationen mit dem Git-Anmeldeinformationsverwaltung für kompatibelAWS CodeCommitaus. Weitere Informationen finden Sie unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden .

  Wenn Sie das Hilfsprogramm für Anmeldeinformationen verwenden möchten, um zu überprüfen, ob dieAWS CLIist für das Richtige konfiguriertAWS-Region, führen Sie das ausaws configureund überprüfen Sie die angezeigten Informationen. Wenn sich das CodeCommit-Repository in einemAWS-Regionanders als der für dieAWS CLI, musst du das führen Sie das aus:aws configurebefehlen und ändern Sie die Werte in diejenigen, die für diese Region geeignet sind. Weitere Informationen finden Sie unter Schritt 1: Erstkonfiguration fürCodeCommit .
- Wenn möglich, deinstallieren Sie Git für Windows und installieren Sie es dann erneut. Achten Sie bei der Installation von Git für Windows darauf, dass das Kontrollkästchen zur Installation des Git-Dienstprogramms zur Verwaltung von Anmeldeinformationen (Git Credential Manager) nicht aktiviert ist. Dieses Programm zur Verwaltung von Anmeldeinformationen ist nicht mit dem AWS CodeCommit-Hilfsprogramm für Anmeldeinformationen kompatibel. Wenn Sie den Git Credential Manager oder ein anderes Dienstprogramm zur Verwaltung von Anmeldeinformationen installiert haben und es nicht deinstallieren möchten, können Sie Ihren `.gitconfig`Datei und fügen Sie die Verwaltung von Anmeldeinformationen für CodeCommit hinzu:

  1. ÖffnenSystemsteuerung, wählenAnmeldeinformationsverwaltungund entfernen Sie alle gespeicherten Anmeldeinformationen für CodeCommit.
  2. Öffnen Sie Ihre `.gitconfig`-Datei in einem Texteditor wie zum Beispiel dem Windows-eigenen Editor.

> **Note**
>
> Wenn Sie mit mehreren Git-Profilen arbeiten, haben Sie möglicherweise lokale und globale `.gitconfig`-Dateien. Stellen Sie sicher, dass Sie die richtige Datei bearbeiten.

3. Fügen Sie nun der `.gitconfig`-Datei den folgenden Abschnitt hinzu:

```
[credential "https://git-codecommit.*.amazonaws.com"]
    helper = !aws codecommit credential-helper $@
    UseHttpPath = true
```

4. Speichern Sie die Datei und öffnen Sie anschließend eine neue Befehlszeilensitzung, bevor Sie einen erneuten Verbindungsversuch unternehmen.

Sie können diesen Ansatz auch anwenden, wenn Sie das Hilfsprogramm für Anmeldeinformationen für verwenden möchtenAWS CodeCommitWenn Sie eine Verbindung mit CodeCommit-Repositorys und bei der Verbindung mit anderen gehosteten Repositorys (wie zum Beispiel GitHub-Repositorys) eine Verbindung mit anderen gehosteten Repositorys (wie zum Beispiel Git

Um die Einstellung zurückzusetzen, welches Hilfsprogramm für Anmeldeinformationen standardmäßig verwendet wird, können Sie die Option --system anstelle von --global oder --local verwenden, wenn Sie den Befehl git config ausführen.

- Wenn Sie Git-Anmeldeinformationen auf einem Windows-Computer verwenden, können Sie versuchen, Probleme aufgrund von mangelnder RFC-Compliance zu umgehen, indem Sie Ihren Git-Benutzernamen als Teil der Verbindungszeichenfolge eingeben. So können Sie beispielsweise das Problem umgehen und ein Repository mit dem Namen klonen*MyDemoRepo*In der Region USA Ost (Ohio):

```
git clone https://Your-Git-Credential-Username@git-codecommit.us-
east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo
```

> **Note**
>
> Dieser Ansatz funktioniert nicht, wenn in dem Benutzernamen Ihrer Git-Anmeldeinformationen ein @-Zeichen vorkommt. Sie müssen für das Zeichen eine URL-Codierung (auch bekannt als URL-Escaping oder Prozentcodierung) durchführen.

Fehlerbehebung bei Git-Clients undAWS CodeCommit

Die folgenden Informationen können Ihnen dabei helfen, häufige Probleme bei der Verwendung von Git mit AWS CodeCommit-Repositorys zu beheben. Weitere Informationen zur Fehlerbehebung bei Git-Clients bei der Verwendung von HTTPS oder SSH finden Sie unter Fehlerbehebung bei Git-Anmeldeinformationen (HTTPS), Fehlerbehebung bei SSH-Verbindungen und Fehlerbehebung beim Hilfsprogramm für Anmeldeinformationen (HTTPS).

Themen

- Git-Fehler: Fehler: RPC fehlgeschlagen; result=56, HTTP code = 200 fatal: Das Remote-Ende hing unerwartet auf
- Git-Fehler: Zu viele Referenz-Update-Befehle
- Git-Fehler: Push-Übertragung über HTTPS funktioniert in einigen Git-Versionen nicht
- Git-Fehler: "gnutls_handshake() failed"
- Git-Fehler: Git kann das CodeCommit--Repository nicht finden oder hat keine Zugriffsberechtigung für das Repository
- Git unter Windows: Keine unterstützten Authentifizierungsmethoden verfügbar (publickey)

Git-Fehler: Fehler: RPC fehlgeschlagen; result=56, HTTP code = 200 fatal: Das Remote-Ende hing unerwartet auf

Problem: Wird eine umfassende Änderung, eine hohe Anzahl an Änderungen oder ein großes Repository per Push übertragen, werden HTTPS-Langzeitverbindungen häufig aufgrund von Problemen mit der Netzwerkverbindung oder den Firewalleinstellungen vorzeitig beendet.

Mögliche Lösungen: Führen Sie die Push-Übertragung stattdessen mit SSH durch. Zur Migration eines großen Repositorys können Sie auch die Anweisungen unter befolgenMigrieren Sie ein

Repository schrittweiseaus. Stellen Sie darüber hinaus sicher, dass Sie die Größenbeschränkung für einzelne Dateien nicht überschreiten. Weitere Informationen finden Sie unter Kontingente .

Git-Fehler: Zu viele Referenz-Update-Befehle

Problem: Die maximale Anzahl für Referenz-Updates pro Push ist 4 000. Dieser Fehler tritt auf, wenn der Push mehr als 4 000 Referenz-Updates enthält.

Mögliche Lösungen: Versuchen Sie, Branches und Tags mit einzeln zu verschiebengit push --allundgit push --tagsaus. Wenn Sie zu viele Tags haben, teilen Sie die Tags auf mehrere Push-Übertragungen auf. Weitere Informationen finden Sie unter Kontingente .

Git-Fehler: Push-Übertragung über HTTPS funktioniert in einigen Git-Versionen nicht

Problem: Ein Problem mit dem Curl-Update auf Version 7.41.0 verursacht Fehlschläge bei der SSPI-basierten Digest-Authentifizierung. Betroffen ist unter anderem die Git-Version 1.9.5.msysgit.1. Einige Versionen von Git für Windows erfüllen möglicherweise nicht vollständig die in RFC 2617 und RFC 4559 festgeschriebenen Bedingungen. Dies kann zu Problemen mit HTTPS-Verbindungen führen, wenn Git-Anmeldeinformationen und das in der AWS CLI integrierte Hilfsprogramm für Anmeldeinformationen verwendet werden.

Mögliche Lösungen: Überprüfen Sie Ihre Git-Version auf bekannte Probleme oder verwenden Sie eine ältere oder neuere Version. Weitere Informationen über mysysgit finden Sie unter Push to HTTPS Is Broken in den GitHub-Foren. Weitere Informationen zu Problemen mit Git-Versionen für Windows finden Sie unter Version 2.11.0(3) fragt nicht nach Benutzernamen/Passwort.

Git-Fehler: "gnutls_handshake() failed"

Problem: Wenn Sie unter Linux versuchen, mithilfe von Git mit einem CodeCommit--Repository zu kommunizieren, wird eine Fehlermeldung mit folgendem Inhalt angezeigt:`error: gnutls_handshake() failed`aus.

Mögliche Lösungen: Kompilieren Sie Git für OpenSSL. Einen Ansatz finden Sie unter "Error: gnutls_handshake() failed" When Connecting to HTTPS Servers in den Ask Ubuntu-Foren.

Alternativ dazu können Sie SSH anstelle von HTTPS verwenden, um mit CodeCommit-Repositorys zu kommunizieren.

Git-Fehler: Git kann das CodeCommit--Repository nicht finden oder hat keine Zugriffsberechtigung für das Repository

Problem: Ein nachgestellter Schrägstrich in der Verbindungszeichenfolge kann dazu führen, dass Verbindungsversuche fehlschlagen.

Mögliche Lösungen: Stellen Sie sicher, dass Sie den richtigen Namen und die richtige Verbindungszeichenfolge für das Repository angegeben haben und dass keine nachgestellten Schrägstriche angegeben sind. Weitere Informationen finden Sie unter Herstellen einer Verbindung mit einem Repository .

Git unter Windows: Keine unterstützten Authentifizierungsmethoden verfügbar (publickey)

Problem: Nachdem Sie SSH-Zugriff für Windows konfiguriert haben, wird ein Fehler beim Zugriff verweigert angezeigt, wenn Sie versuchen, Befehle wiegit pull,git push, odergit cloneaus.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass eine GIT_SSH-Umgebungsvariable auf Ihrem Computer vorhanden ist, die zur Unterstützung eines anderen Verbindungsdienstprogramms wie zum Beispiel PuTTY konfiguriert ist. Um dieses Problem zu beheben, versuchen Sie es mit einer der folgenden Methoden:

- Öffnen Sie einen Bash-Emulator und fügen Sie den Parameter `GIT_SSH_COMMAND="ssh"` vor dem Git-Befehl hinzu. Wenn Sie zum Beispiel versuchen, ein Repository zu klonen, führen Sie anstelle von git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/MyDemoRepo my-demo-repo den folgenden Befehl aus:

  ```
  GIT_SSH_COMMAND="ssh" git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/
  repos/MyDemoRepo my-demo-repo
  ```

- Löschen Sie die Umgebungsvariable `GIT_SSH` oder benennen Sie sie um, wenn Sie sie nicht mehr verwenden. Öffnen Sie anschließend ein neues Befehlszeilenfenster oder eine neue Bash-Emulator-Sitzung und versuchen Sie erneut, den Befehl einzugeben.

Weitere Informationen zur Behebung von Git-Problemen unter Windows bei Verwendung von SSH finden Sie unter Fehlerbehebung bei SSH-Verbindungen.

Fehlerbehebung bei Zugriffsfehlern undAWS CodeCommit

Die folgenden Informationen können Ihnen helfen, Zugriffsfehler zu beheben, die bei der Verbindung mit AWS CodeCommit-Repositorys auftreten können.

Themen

- Zugriffsfehler: Ich werde zur Eingabe eines Benutzernamens und eines Kennworts aufgefordert, über Windows eine Verbindung mit einem CodeCommit-Repository herzustellen
- Zugriffsfehler: Verweigerung des öffentlichen Schlüssels bei der Verbindung mit einem CodeCommit-Repository
- Zugriffsfehler: Meldung „Quote überschritten“ oder „429" bei der Verbindung mit einem CodeCommit-Repository

Zugriffsfehler: Ich werde zur Eingabe eines Benutzernamens und eines Kennworts aufgefordert, über Windows eine Verbindung mit einem CodeCommit-Repository herzustellen

Problem: Wenn Sie versuchen, mithilfe von Git mit einem CodeCommit-Repository zu kommunizieren, werden Sie in einem Dialogfeld aufgefordert, Ihren Benutzernamen und Ihr Passwort einzugeben.

Mögliche Lösungen: Dies könnte das integrierte Verwaltungssystem für Anmeldeinformationen für Windows sein. Abhängig von Ihrer Konfiguration führen Sie eine der folgenden Maßnahmen durch:

- Wenn Sie HTTPS mit Git-Anmeldeinformationen verwenden, sind Ihre Git-Anmeldeinformationen noch nicht im System gespeichert. Stellen Sie die Git-Anmeldeinformationen bereit und fahren Sie fort. Sie sollten nicht mehr zur Eingabe aufgefordert werden. Weitere Informationen finden Sie unter Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden .
- Wenn Sie HTTPS mit dem Anmeldeinformationen-Hilfsprogramm für AWS CodeCommit verwenden, ist dieses nicht kompatibel mit dem System für die Anmeldeinformationsverwaltung von Windows Klicken Sie auf Abbrechen.

  Dieses Problem kann auch bedeuten, dass Sie das Git-Dienstprogramm zur Verwaltung von Anmeldeinformationen (Git Credential Manager) bei der Installation von Git für Windows installiert haben. Der Git Credential Manager ist CodeCommit demAWS CLIaus. Erwägen Sie, den Git-Anmeldeinformations-Manager zu deinstallieren. Sie können auch installieren und konfigurierengit-

remote-codecommitAls Alternative zur Verwendung des Hilfsprogramms für Anmeldeinformationen für CodeCommit.

Weitere Informationen finden Sie unter Einrichtungsschritte für HTTPS-Verbindungen zuAWS CodeCommitmitgit-remote-codecommit, Für HTTPS-Verbindungen unter Windows mit demAWS CLICredential-Helper und Git für Windows: Ich habe Git für Windows installiert, aber mir wird der Zugriff auf mein Repository verweigert (403).

Zugriffsfehler: Verweigerung des öffentlichen Schlüssels bei der Verbindung mit einem CodeCommit-Repository

Problem: Wenn Sie versuchen, mithilfe eines SSH-Endpunkts mit einem CodeCommit-Repository zu kommunizieren, wird eine Fehlermeldung mit folgendem Inhalt angezeigt:.`Error: public key deniedaus`.

Mögliche Lösungen: Der häufigste Grund für diesen Fehler ist, dass Sie die Einrichtung für SSH-Verbindungen nicht abgeschlossen haben. Konfigurieren Sie ein SSH-Schlüsselpaar (öffentlicher und privater Schlüssel) und verknüpfen Sie dann den öffentlichen Schlüssel mit Ihrem IAM-Benutzer. Weitere Informationen zur Konfiguration von SSH finden Sie unter Für SSH-Verbindungen unter Linux, macOS oder Unix und Für SSH-Verbindungen unter Windows.

Zugriffsfehler: Meldung „Quote überschritten“ oder „429" bei der Verbindung mit einem CodeCommit-Repository

Problem: Wenn Sie versuchen, mit einem CodeCommit-Repository zu kommunizieren, erscheint eine Meldung mit „Quote überschritten“ oder dem Fehlercode „429". Die Kommunikation wird entweder erheblich verlangsamt oder schlägt fehl.

Ursache: Alle Aufrufe von CodeCommit, sei es aus einer Anwendung, derAWS CLI, einem Git-Client oder derAWS Management Consoleunterliegen einer maximalen Anzahl von Anfragen pro Sekunde und insgesamt aktiven Anforderungen. Sie dürfen die maximal zulässige Anforderungsrate für ein Amazon Web Services Services-Konto in keinemAWS-Regionaus. Wenn Anforderungen den Höchstsatz überschreiten, erhalten Sie eine Fehlermeldung und weitere Anrufe werden für Ihr Amazon Web Services Services-Konto vorübergehend gedrosselt. Während des Drosselungszeitraums werden Ihre Verbindungen mit CodeCommit verlangsamt und schlagen möglicherweise fehl.

Mögliche Lösungen: Ergreifen Sie Schritte, um die Anzahl der Verbindungen oder Aufrufe von CodeCommit zu reduzieren oder Anforderungen zu verteilen. Beachten Sie folgende Ansätze:

- Implementieren von Jitter in Anforderungen, insbesondere in regelmäßige Abfrageanforderungen

  Wenn eine Anwendung regelmäßig abfragt und diese Anwendung auf mehreren Amazon EC2 EC2-Instances ausgeführt wird, implementieren Sie Jitter (eine zufällige Verzögerung), sodass verschiedene Amazon EC2 EC2-Instances nicht zur gleichen Sekunde abfragen. Wir empfehlen eine Zufallszahl von 0 bis 59 Sekunden, um Abfragemechanismen gleichmäßig über einen einminütigen Zeitraum zu verteilen.
- Verwenden einer ereignisbasierten Architektur anstelle von Abfragen

  Verwenden Sie anstelle von Abfragen eine ereignisbasierte Architektur, sodass Aufrufe nur gemacht werden, wenn ein Ereignis eintritt. Erwägen Sie, CloudWatch Events -Benachrichtigungen fürAWS CodeCommitVeranstaltungenum Ihren Workflow auszulösen.
- Implementieren Sie Wiederholversuche bei Fehlern und exponentielles Backoff für APIs und automatisierte Git-Aktionen

  Durch Wiederholversuche bei Fehlern und exponentielle Backoffs lässt sich die Rate der Aufrufe beschränken. Jedes AWS-SDK implementiert Logik für automatische Wiederholungsversuche und Algorithmen für exponentielles Backoff. Für automatisierte Git-Push- und Git-Pull-Aktionen müssen Sie möglicherweise Ihre eigene Logik für Wiederholversuche implementieren. Weitere Informationen finden Sie unterWiederholversuche bei Fehlern und exponentielles Backoff inAWSaus.
- Fordern Sie eine Erhöhung des CodeCommit-DienstkontingentsAWSSupportcenter

  Um eine Erhöhung des Service Limits zu erhalten, müssen Sie bestätigen, dass Sie die hier gemachten Vorschläge bereits befolgt haben, einschließlich der Methoden für Wiederholversuche bei Fehlern und exponentielles Backoff. Darüber hinaus müssen Sie in Ihrer AnfrageAWS-Region, Amazon Web Services Services-Konto und Zeitrahmen, der von den Drosselungsproblemen betroffen ist.

Beheben von Konfigurationsfehlern undAWS CodeCommit

Die folgenden Informationen können Ihnen helfen, Konfigurationsfehler zu beheben, die bei der Verbindung mit AWS CodeCommit-Repositorys auftreten können.

Themen

- Konfigurationsfehler: Kann nicht konfiguriert werdenAWS CLIAnmeldeinformationen unter macOS

Konfigurationsfehler: Kann nicht konfiguriert werdenAWS CLIAnmeldeinformationen unter macOS

Problem: Wenn du rennstaws configureum dasAWS CLI, siehst du einConfigParseErrormessage.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass bereits eine Datei mit Anmeldeinformationen existiert. Navigieren Sie zu ~/.aws und suchen Sie eine Datei mit dem Namen `credentials`. Löschen Sie diese Datei oder benennen Sie sie um und führen Sie anschließend aws configure erneut aus.

Behebung von Konsolenfehlern und AWS CodeCommit

Die folgenden Informationen können Ihnen helfen, Konsolenfehler zu beheben, die bei der Verbindung mit AWS CodeCommit-Repositorys auftreten können.

Themen

- Zugriffsfehler: Zugriff auf Verschlüsselungsschlüssel für CodeCommit-Repository verweigert (Konsole oder AWS CLI)
- Verschlüsselungsfehler: Das Repository kann nicht entschlüsselt werden
- Konsolenfehler: Der Code in einem CodeCommit Repository kann nicht von der Konsole aus durchsucht werden
- Anzeigefehler: Eine Datei oder ein Vergleich zwischen Dateien kann nicht angezeigt werden

Zugriffsfehler: Zugriff auf Verschlüsselungsschlüssel für CodeCommit-Repository verweigert (Konsole oder AWS CLI)

Problem: Wenn Sie versuchen, CodeCommit von der Konsole oder der aus darauf zuzugreifenAWS CLI, wird eine Fehlermeldung angezeigt, die den Ausdruck `EncryptionKeyAccessDeniedException` oder enthält`User is not authorized for the KMS default key for CodeCommit 'aws/codecommit' in your account`.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass Ihr Amazon Web Services Services-Konto nicht abonniert istAWS Key Management Service, was für CodeCommit erforderlich ist. Öffnen Sie die AWS KMS Konsole, wählen Sie AWSManaged Keys und dann Get Started Now. Wenn eine Meldung angezeigt wird, dass Sie derzeit kein Abonnement für den AWS Key Management Service-Service haben, befolgen Sie die Anweisungen auf dieser Seite, um den Service zu abonnieren. Weitere Informationen zu CodeCommit und finden AWS Key Management Service Sie unterAWS KMS und Verschlüsselung.

Verschlüsselungsfehler: Das Repository kann nicht entschlüsselt werden

Problem: Wenn Sie versuchen, über die Konsole oder die auf ein CodeCommit Repository zuzugreifenAWS CLI, erscheint eine Fehlermeldung, die den Ausdruck `Repository can't be decrypted` enthält.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass der AWS KMS Schlüssel, der zum Verschlüsseln und Entschlüsseln von Daten für dieses Repository verwendet wird, nicht aktiv ist oder noch gelöscht werden muss. Für ist ein aktiver Von AWS verwalteter Schlüssel oder vom Kunden verwalteter Schlüssel AWS Key Management Service erforderlich. CodeCommit Öffnen Sie die AWS KMS Konsole, wählen Sie Von AWS verwaltete Schlüsseloder Vom Kunden verwaltete Schlüssel und stellen Sie sicher, dass der für das Repository verwendete Schlüssel dort vorhanden ist, AWS-Region wo das Repository existiert, und dass sein Status Aktiv ist. Weitere Informationen zu CodeCommit und finden AWS Key Management Service Sie unterAWS KMS und Verschlüsselung.

> **Important**
>
> Wenn der Schlüssel, der zum Verschlüsseln und Entschlüsseln der Daten für das Repository verwendet wurde, dauerhaft gelöscht wurde oder auf andere Weise nicht zugänglich ist, kann auf Daten in den Repositorys, die mit diesem Schlüssel verschlüsselt wurden, nicht zugegriffen werden.

Konsolenfehler: Der Code in einem CodeCommit Repository kann nicht von der Konsole aus durchsucht werden

Problem: Wenn Sie versuchen, den Inhalt eines Repositorys von der Konsole aus anzuzeigen, wird eine Fehlermeldung angezeigt und der Zugriff wird verweigert.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist, dass eine auf Ihr Amazon Web Services Services-Konto angewendete IAM-Richtlinie eine oder mehrere der Berechtigungen verweigert, die für das Durchsuchen von Code von der CodeCommit Konsole aus erforderlich sind. Weitere Informationen zu CodeCommit Zugriffsberechtigungen und zum Surfen finden Sie unter. Authentifizierung und Zugriffskontrolle für AWS CodeCommit

Anzeigefehler: Eine Datei oder ein Vergleich zwischen Dateien kann nicht angezeigt werden

Problem: Wenn Sie versuchen, eine Datei oder einen Vergleich zwischen zwei Versionen einer Datei in der CodeCommit Konsole anzuzeigen, wird ein Fehler angezeigt, der besagt, dass die Datei oder der Unterschied zu groß ist, um angezeigt zu werden.

Mögliche Lösungen: Die häufigste Ursache für diesen Fehler ist entweder, dass die Datei zu groß für die Anzeige ist, eine oder mehrere Zeilen enthält, die die Zeichenbeschränkung für eine einzelne Zeile in einer Datei überschreiten, oder dass der Unterschied zwischen den beiden Versionen der Datei die Zeilenbegrenzung überschreitet. Weitere Informationen finden Sie unter Kontingente. Um die Datei oder die Unterschiede zwischen den Versionen der Datei anzuzeigen, können Sie die Datei lokal in Ihrer bevorzugten IDE öffnen, ein Git-Diff-Tool verwenden oder den git diff Befehl ausführen.

Fehlerbehebung bei Auslösern undAWS CodeCommit

Die folgenden Informationen können Ihnen helfen, Probleme mit Auslösern in AWS CodeCommit zu beheben.

Themen

- Fehler auslösen: Ein Repository-Auslöser wird nicht zum erwarteten Zeitpunkt ausgeführt

Fehler auslösen: Ein Repository-Auslöser wird nicht zum erwarteten Zeitpunkt ausgeführt

Problem: Ein oder mehrere der für ein Repository konfigurierten Auslöser werden offenbar gar nicht oder nicht wie erwartet ausgeführt.

Mögliche Lösungen: Wenn das Ziel des Triggers einAWS Lambdastellen Sie sicher, dass Sie die Ressourcenrichtlinie der Funktion für den Zugriff durch CodeCommit konfiguriert haben.

Weitere Informationen finden Sie unter Beispiel 3: Erstellen Sie eine Richtlinie für die AWS Lambda Integration mit einem Trigger CodeCommit .

Alternativ dazu können Sie den Auslöser bearbeiten. Stellen Sie sicher, dass die Ereignisse, für die Sie Aktionen auslösen möchten, ausgewählt wurden und dass die Branches für den Auslöser den Branch umfassen, in dem Sie Reaktionen auf Aktionen sehen möchten. Ändern Sie versuchsweise die Einstellungen für den Auslöser in All repository events und All branches und testen Sie dann den Auslöser. Weitere Informationen finden Sie unter Trigger für ein Repository bearbeiten.

Aktivieren des Debuggings

Problem: Ich möchte Debugging aktivieren, um mehr Informationen zu meinem Repository zu erhalten und dazu, wie Git Befehle ausführt.

Mögliche Lösungen: Versuchen Sie die folgenden Maßnahmen:

1. Führen Sie in einem Terminal oder einer Eingabeaufforderung auf Ihrem lokalen Computer vor der Ausführung von Git-Befehlen die folgenden Befehle aus:

   Unter Linux, macOS oder Unix:

   ```
   export GIT_TRACE_PACKET=1
   export GIT_TRACE=1
   export GIT_CURL_VERBOSE=1
   ```

   Unter Windows:

   ```
   set GIT_TRACE_PACKET=1
   set GIT_TRACE=1
   set GIT_CURL_VERBOSE=1
   ```

   > **Note**
   >
   > Die Einstellung von `GIT_CURL_VERBOSE` ist nur für HTTPS-Verbindungen hilfreich. SSH verwendet die `libcurl`-Bibliothek nicht.

2. Um weitere Informationen zu Ihrem Git-Repository zu erhalten, empfehlen wir, die neueste Version von Git-Sizer zu installieren. Folgen Sie den Anweisungen, um das für Ihr Betriebssystem und Ihre Umgebung geeignete Dienstprogramm zu ermitteln. Ändern Sie nach

der Installation in der Befehlszeile oder im Terminal die Verzeichnisse in Ihr lokales Repository und führen Sie dann den folgenden Befehl aus:

```
git-sizer --verbose
```

> **Tip**
>
> Erwägen Sie, die Ausgabe des Befehls in einer Datei zu speichern, damit Sie sie bei der Behebung von Problemen, insbesondere im Laufe der Zeit, problemlos mit anderen teilen können.

AWS CodeCommit Referenz

Die folgenden Referenzthemen können Ihnen helfen CodeCommit, Git AWS-Regionen, Service Limits und mehr besser zu verstehen.

Themen

- Regionen und Git-Verbindungsendpunkte für AWS CodeCommit
- Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle
- Kontingente in AWS CodeCommit
- AWS CodeCommit Befehlszeilenreferenz
- Grundlegende Git-Befehle

Regionen und Git-Verbindungsendpunkte für AWS CodeCommit

Jedes CodeCommit Repository ist mit einem verknüpft. AWS-Region CodeCommit bietet regionale Endpunkte, über die Sie Ihre Anfragen an den Dienst stellen können. Darüber hinaus CodeCommit bietet es in jeder Region, in der es verfügbar CodeCommit ist, Git-Verbindungsendpunkte für SSH- und HTTPS-Protokolle.

Alle Beispiele in diesem Handbuch verwenden dieselbe Endpunkt-URL für Git in US East (Ohio):`git-codecommit.us-east-2.amazonaws.com`. Wenn Sie Git verwenden und Ihre Verbindungen konfigurieren, stellen Sie jedoch sicher, dass Sie den Git-Verbindungsendpunkt wählen, der dem entspricht AWS-Region , der Ihr CodeCommit Repository hostet. Wenn du beispielsweise eine Verbindung zu einem Repository im Osten der USA (Nord-Virginia) herstellen möchtest, verwende die Endpunkt-URL von`git-codecommit.us-east-1.amazonaws.com`. Dies gilt auch für API-Aufrufe. Wenn Sie mit den AWS CLI oder den SDKs Verbindungen zu einem CodeCommit Repository herstellen, stellen Sie sicher, dass Sie den richtigen regionalen Endpunkt für das Repository verwenden.

Themen

- Unterstützt AWS-Regionen für CodeCommit
- Endpunkte für Git-Verbindungen
- Server-Fingerabdrücke für CodeCommit

Unterstützt AWS-Regionen für CodeCommit

Sie können CodeCommit Repositorys wie folgt AWS-Regionen erstellen und verwenden:

- US East (Ohio)
- USA Ost (Nord-Virginia)
- USA West (Nordkalifornien)
- US West (Oregon)
- Europa (Irland)
- Europe (London)
- Europa (Paris)
- Europa (Frankfurt)
- Europa (Stockholm)
- Europa (Milan)
- Afrika (Kapstadt)
- Israel (Tel Aviv)
- Asien-Pazifik (Tokio)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Jakarta)
- Naher Osten (VAE)
- Asien-Pazifik (Seoul)
- Asien-Pazifik (Osaka)
- Asien-Pazifik (Mumbai)
- Asien-Pazifik (Hyderabad)
- Asien-Pazifik (Hongkong)
- Südamerika (São Paulo)
- Naher Osten (Bahrain)
- Kanada (Zentral)
- China (Peking)
- China (Ningxia)
- AWS GovCloud (US-West)

- AWS GovCloud (US-Ost)

CodeCommit hat in einigen Regionen Unterstützung für den Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2 hinzugefügt. Weitere Informationen über FIPS und FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-2 – Übersicht. Informationen zu Git-Verbindungsendpunkten, die FIPS unterstützen, finden Sie unter Endpunkte für Git-Verbindungen.

Weitere Informationen zu regionalen Endpunkten AWS CLI, Service- und API-Aufrufen finden Sie unter AWS CodeCommit Endpunkte und CodeCommit Kontingente.

Endpunkte für Git-Verbindungen

Verwende die folgenden URLs, wenn du Git-Verbindungen zu CodeCommit Repositorys konfigurierst:

Git-Verbindungsendpunkte für AWS CodeCommit

Name der Region	Region	Endpunkt-URL	Protokoll
USA Ost (Ohio)	us-east-2	https://git-codecommit.us-east-2.amazonaws.com	HTTPS
USA Ost (Ohio)	us-east-2	ssh://git-codecommit.us-east-2.amazonaws.com	SSH
USA Ost (Ohio)	us-east-2	git-codecommit-fipshttps://.us-east-2.amazonaws.com	HTTPS
USA Ost (Nord-Virginia)	us-east-1	https://git-codecommit.us-east-1.amazonaws.com	HTTPS
USA Ost (Nord-Virginia)	us-east-1	ssh://git-codecommit.us-east-1.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
USA Ost (Nord-Virginia)	us-east-1	git-codecommit-fipshttps://.us-east-1.amazonaws.com	HTTPS
USA West (Oregon)	us-west-2	https://git-codecommit.us-west-2.amazonaws.com	HTTPS
USA West (Oregon)	us-west-2	ssh://git-codecommit.us-west-2.amazonaws.com	SSH
USA West (Oregon)	us-west-2	git-codecommit-fipshttps://.us-west-2.amazonaws.com	HTTPS
USA West (Nordkalifornien)	us-west-1	https://git-codecommit.us-west-1.amazonaws.com	HTTPS
USA West (Nordkalifornien)	us-west-1	ssh://git-codecommit.us-west-1.amazonaws.com	SSH
USA West (Nordkalifornien)	us-west-1	git-codecommit-fipshttps://.us-west-1.amazonaws.com	HTTPS
Europa (Irland)	eu-west-1	https://git-codecommit.eu-west-1.amazonaws.com	HTTPS
Europa (Irland)	eu-west-1	ssh://git-codecommit.eu-west-1.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
Asien-Pazifik (Tokio)	ap-northeast-1	https://git-codecommit.ap-northeast-1.amazonaws.com	HTTPS
Asien-Pazifik (Tokio)	ap-northeast-1	ssh://git-codecommit.ap-northeast-1.amazonaws.com	SSH
Asien-Pazifik (Singapur)	ap-southeast-1	https://git-codecommit.ap-southeast-1.amazonaws.com	HTTPS
Asien-Pazifik (Singapur)	ap-southeast-1	ssh://git-codecommit.ap-southeast-1.amazonaws.com	SSH
Asien-Pazifik (Sydney)	ap-southeast-2	https://git-codecommit.ap-southeast-2.amazonaws.com	HTTPS
Asien-Pazifik (Sydney)	ap-southeast-2	ssh://git-codecommit.ap-southeast-2.amazonaws.com	SSH
Asien-Pazifik (Jakarta)	ap-southeast-3	https://git-codecommit.ap-southeast-3.amazonaws.com	HTTPS
Asien-Pazifik (Jakarta)	ap-southeast-3	ssh://git-codecommit.ap-southeast-3.amazonaws.com	SSH
Naher Osten (VAE)	me-central-1	https://git-codecommit.me-central-1.amazonaws.com	HTTPS

Name der Region	Region	Endpunkt-URL	Protokoll
Naher Osten (VAE)	me-central-1	ssh://git-codecommit.me-central-1.amazonaws.com	SSH
Europa (Frankfurt)	eu-central-1	https://git-codecommit.eu-central-1.amazonaws.com	HTTPS
Europa (Frankfurt)	eu-central-1	ssh://git-codecommit.eu-central-1.amazonaws.com	SSH
Asien-Pazifik (Seoul)	ap-northeast-2	https://git-codecommit.ap-northeast-2.amazonaws.com	HTTPS
Asien-Pazifik (Seoul)	ap-northeast-2	ssh://git-codecommit.ap-northeast-2.amazonaws.com	SSH
Südamerika (São Paulo)	sa-east-1	https://git-codecommit.sa-east-1.amazonaws.com	HTTPS
Südamerika (São Paulo)	sa-east-1	ssh://git-codecommit.sa-east-1.amazonaws.com	SSH
Europa (London)	eu-west-2	https://git-codecommit.eu-west-2.amazonaws.com	HTTPS
Europa (London)	eu-west-2	ssh://git-codecommit.eu-west-2.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
Asien-Pazifik (Mumbai)	ap-south-1	https://git-codecommit.ap-south-1.amazonaws.com	HTTPS
Asien-Pazifik (Mumbai)	ap-south-1	ssh://git-codecommit.ap-south-1.amazonaws.com	SSH
Asien-Pazifik (Hyderabad)	ap-south-2	https://git-codecommit.ap-south-2.amazonaws.com	HTTPS
Asien-Pazifik (Hyderabad)	ap-south-2	ssh://git-codecommit.ap-south-2.amazonaws.com	SSH
Kanada (Zentral)	ca-central-1	https://git-codecommit.ca-central-1.amazonaws.com	HTTPS
Kanada (Zentral)	ca-central-1	ssh://git-codecommit.ca-central-1.amazonaws.com	SSH
Kanada (Zentral)	ca-central-1	git-codecommit-fipshttps://.ca-central-1.amazonaws.com	HTTPS
Europa (Paris)	eu-west-3	https://git-codecommit.eu-west-3.amazonaws.com	HTTPS
Europa (Paris)	eu-west-3	ssh://git-codecommit.eu-west-3.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
AWS GovCloud (US-West)	us-gov-west-1	https://git-codecommit.us-gov-west-1.amazonaws.com	HTTPS
AWS GovCloud (US-West)	us-gov-west-1	ssh: //git-codecommit.us-gov-west-1.amazonaws.com	SSH
AWS GovCloud (US-West)	us-gov-west-1	https://git-codecommit-fips. us-gov-west-1.amazonaws.com	HTTPS
AWS GovCloud (US-Ost)	us-gov-east-1	https://git-codecommit.us-gov-east-1.amazonaws.com	HTTPS
AWS GovCloud (US-Ost)	us-gov-east-1	ssh: //git-codecommit.us-gov-east-1.amazonaws.com	SSH
AWS GovCloud (US-Ost)	us-gov-east-1	https://git-codecommit-fips. us-gov-east-1.amazonaws.com	HTTPS
Europa (Stockholm)	eu-north-1	https://git-codecommit.eu-north-1.amazonaws.com	HTTPS
Europa (Stockholm)	eu-north-1	ssh://git-codecommit.eu-north-1.amazonaws.com	SSH
Naher Osten (Bahrain)	me-south-1	https://git-codecommit.me-south-1.amazonaws.com	HTTPS

Name der Region	Region	Endpunkt-URL	Protokoll
Naher Osten (Bahrain)	me-south-1	ssh://git-codecommit.me-south-1.amazonaws.com	SSH
Asien-Pazifik (Hongkong)	ap-east-1	https://git-codecommit.ap-east-1.amazonaws.com	HTTPS
Asien-Pazifik (Hongkong)	ap-east-1	ssh://git-codecommit.ap-east-1.amazonaws.com	SSH
China (Beijing)	cn-north-1	https://git-codecommit.cn-north-1.amazonaws.com.cn	HTTPS
China (Beijing)	cn-north-1	ssh://git-codecommit.cn-north-1.amazonaws.com .cn	SSH
China (Ningxia)	cn-northwest-1	https://git-codecommit.cn-northwest-1.amazonaws.com.cn	HTTPS
China (Ningxia)	cn-northwest-1	ssh://git-codecommit.cn-northwest-1.amazonaws.com .cn	SSH
Europa (Mailand)	eu-south-1	https://git-codecommit.eu-south-1.amazonaws.com	HTTPS
Europa (Mailand)	eu-south-1	ssh://git-codecommit.eu-south-1.amazonaws.com	SSH

Name der Region	Region	Endpunkt-URL	Protokoll
Asien-Pazifik (Osaka)	ap-northeast-3	https://git-codecommit.ap-northeast-3.amazonaws.com	HTTPS
Asien-Pazifik (Osaka)	ap-northeast-3	ssh://git-codecommit.ap-northeast-3.amazonaws.com	SSH
Afrika (Kapstadt)	af-south-1	https://git-codecommit.af-south-1.amazonaws.com	HTTPS
Afrika (Kapstadt)	af-south-1	ssh://git-codecommit.af-south-1.amazonaws.com	SSH
Israel (Tel Aviv)	il-central-1	https://git-codecommit.il-central-1.amazonaws.com	HTTPS
Israel (Tel Aviv)	il-central-1	ssh://git-codecommit.il-central-1.amazonaws.com	SSH

Server-Fingerabdrücke für CodeCommit

In der folgenden Tabelle sind die öffentlichen Fingerabdrücke für Git-Verbindungsendpunkte in aufgeführt. CodeCommit Diese Server-Fingerprints werden als Teil des Verifizierungsvorgangs beim Hinzufügen eines Endpunkts zu Ihrer bekannten Hosts-Datei angezeigt.

Öffentliche Fingerabdrücke für CodeCommit

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.us-east-2.amazonaws.com	MD5	`a9:6d:03:ed:08:42:21:be:06:e1:e0:2a:d1:75:31:5e`
git-codecommit.us-east-2.amazonaws.com	SHA256	`3lBlW2g5xn/NA2Ck6dyeJIrQOWvn7n8UEs56fG6ZIzQ`
git-codecommit.us-east-1.amazonaws.com	MD5	`a6:9c:7d:bc:35:f5:d4:5f:8b:ba:6f:c8:bc:d4:83:84`
git-codecommit.us-east-1.amazonaws.com	SHA256	`eLMY1j0DKA4uvDZcl/KgtIayZANwX6t8+8isPtotBoY`
git-codecommit.us-west-2.amazonaws.com	MD5	`a8:68:53:e3:99:ac:6e:d7:04:7e:f7:92:95:77:a9:77`
git-codecommit.us-west-2.amazonaws.com	SHA256	`0pJx9SQpkbPUAHwy58UVIq0IHcyo1fwCpOOuVgcAWPo`
git-codecommit.eu-west-1.amazonaws.com	MD5	`93:42:36:ea:22:1f:f1:0f:20:02:4a:79:ff:ea:12:1d`
git-codecommit.eu-west-1.amazonaws.com	SHA256	`tKjRkOL8dmJyTmSbeSdN1S8F/f0iql3RlvqgTOP1UyQ`
git-codecommit.ap-northeast-1.amazonaws.com	MD5	`8e:a3:f0:80:98:48:1c:5c:6f:59:db:a7:8f:6e:c6:cb`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.ap-northeast-1.amazonaws.com	SHA256	`Xk/WeYD/K/bnBybzhiuu4dWpBJtXPf7E30jHU7se4Ow`
git-codecommit.ap-southeast-1.amazonaws.com	MD5	`65:e5:27:c3:09:68:0d:8e:b7:6d:94:25:80:3e:93:cf`
git-codecommit.ap-southeast-1.amazonaws.com	SHA256	`ZIsVa7OVzxrTIf+Rk4UbhPv6Es22mSB3uTBojfPXIno`
git-codecommit.ap-southeast-2.amazonaws.com	MD5	`7b:d2:c1:24:e6:91:a5:7b:fa:c1:0c:35:95:87:da:a0`
git-codecommit.ap-southeast-2.amazonaws.com	SHA256	`nYp+gHas80HY3DqbP4yanCDFhqDVjseefVbHEXqH2Ec`
git-codecommit.ap-southeast-3.amazonaws.com	MD5	`64:d9:e0:53:19:4f:a8:91:9a:c3:53:22:a6:a8:ed:a6`
git-codecommit.ap-southeast-3.amazonaws.com	SHA256	`ATdkGSFhpqIu7RqUVT/1RZo6MLxxxUW9NoDVMbAc/6g`
git-codecommit.me-central-1.amazonaws.com	MD5	`bd:fa:e2:f9:05:84:d6:39:6f:bc:d6:8d:fe:de:61:76`
git-codecommit.me-central-1.amazonaws.com	SHA256	`grceUDWubo4MzG1NoaKZKUfrgPvfN3ijliOnQr1lTZA`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.eu-central-1.amazonaws.com	MD5	`74:5a:e8:02:fc:b2:9c:06:10:b4:78:84:65:94:22:2d`
git-codecommit.eu-central-1.amazonaws.com	SHA256	`MwGrkiEki8QkkBtlAgXbYt0hoZYBnZF62VY5RzGJEUY`
git-codecommit.ap-northeast-2.amazonaws.com	MD5	`9f:68:48:9b:5f:fc:96:69:39:45:58:87:95:b3:69:ed`
git-codecommit.ap-northeast-2.amazonaws.com	SHA256	`eegAPQrWY9YsYo9ZHIKOmxetfXBHzAZd8Eya53Qcwko`
git-codecommit.sa-east-1.amazonaws.com	MD5	`74:99:9d:ff:2b:ef:63:c6:4b:b4:6a:7f:62:c5:4b:51`
git-codecommit.sa-east-1.amazonaws.com	SHA256	`kW+VKB0jpRaG/ZbXkgbtMQbKgEDK7JnISV3SVoyCmzU`
git-codecommit.us-west-1.amazonaws.com	MD5	`3b:76:18:83:13:2c:f8:eb:e9:a3:d0:51:10:32:e7:d1`
git-codecommit.us-west-1.amazonaws.com	SHA256	`gzauWTWXDK2u5KuMMi5vbKTmfyerdIwgSbzYBODLpzg`
git-codecommit.eu-west-2.amazonaws.com	MD5	`a5:65:a6:b1:84:02:b1:95:43:f9:0e:de:dd:ed:61:d3`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.eu-west-2.amazonaws.com	SHA256	`r0Rwz5k/IHp/QyrRnfiM9j02D5UEqMbtFNTuDG2hNbs`
git-codecommit.ap-south-1.amazonaws.com	MD5	`da:41:1e:07:3b:9e:76:a0:c5:1e:64:88:03:69:86:21`
git-codecommit.ap-south-1.amazonaws.com	SHA256	`hUKwnTj7+Xpx4Kddb6p45j4RazIJ4IhAMD8k29itOfE`
git-codecommit.ap-south-2.amazonaws.com	MD5	`bc:cc:9f:15:f8:f3:58:a2:68:65:21:e2:23:71:8d:ce`
git-codecommit.ap-south-2.amazonaws.com	SHA256	`Xe0CyZEOvgR5Xa2YUGqf+jn8/Ut7l7nX/CmslSFNEig`
git-codecommit.ca-central-1.amazonaws.com	MD5	`9f:7c:a2:2f:8c:b5:74:fd:ab:b7:e1:fd:af:46:ed:23`
git-codecommit.ca-central-1.amazonaws.com	SHA256	`Qz5puafQdANVprLlj6r0Qyh4lCNsF6ob61dGcPtFS7w`
git-codecommit.eu-west-3.amazonaws.com	MD5	`1b:7f:97:dd:d7:76:8a:32:2c:bd:2c:7b:33:74:6a:76`
git-codecommit.eu-west-3.amazonaws.com	SHA256	`uw7c2FL564jVoFgtc+ikzILnKBsZz7t9+CFdSJjKbLI`

Server	Kryptografischer Hash-Typ	Fingerabdruck
Git-Code-Commit. us-gov-west-1.amazonaws.com	MD5	`9f:6c:19:3b:88:cd:e8:88:1b:9c:98:6a:95:31:8a:69`
Git-Code-Commit. us-gov-west-1.amazonaws.com	SHA256	`djXQoSIFcg8vHe0KVH1xW/gOF9X37tWTqu4Hkng75x4`
Git-Code-Commit. us-gov-east-1.amazonaws.com	MD5	`00:8d:b5:55:6f:05:78:05:ed:ea:cb:3f:e6:f0:62:f2`
Git-Code-Commit. us-gov-east-1.amazonaws.com	SHA256	`fVb+R0z7qW7minenW+rUpAABRCRBTCzmETAJEQrg98`
git-codecommit.eu-north-1.amazonaws.com	MD5	`8e:53:d8:59:35:88:82:fd:73:4b:60:8a:50:70:38:f4`
git-codecommit.eu-north-1.amazonaws.com	SHA256	`b6KSK7xKq+V8jl7iuAcjqXsG7zkqoUZZmmhYYFBq1wQ`
git-codecommit.me-south-1.amazonaws.com	MD5	`0e:39:28:56:d5:41:e6:8d:fa:81:45:37:fb:f3:cd:f7`
git-codecommit.me-south-1.amazonaws.com	SHA256	`O+NToCGgjrHekiBuOl0ad7ROGEsz+DBLXOd/c9wc0JU`
git-codecommit.ap-east-1.amazonaws.com	MD5	`a8:00:3d:24:52:9d:61:0e:f6:e3:88:c8:96:01:1c:fe`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.ap-east-1.amazonaws.com	SHA256	`LafadYwUYW8hONoTRpojbjNs9IRnbEwHtezD3aAIBX0`
git-codecommit.cn-north-1.amazonaws.com .cn	MD5	`11:7e:2d:74:9e:3b:94:a2:69:14:75:6f:5e:22:3b:b3`
git-codecommit.cn-north-1.amazonaws.com .cn	SHA256	`IYUXxH2OpTDsyYMLIp+JY8CTLS4UX+ZC5JVZXPRaxc8`
git-codecommit.cn-northwest-1.amazonaws.com .cn	MD5	`2e:a7:fb:4c:33:ac:6c:f9:aa:f2:bc:fb:0a:7b:1e:b6`
git-codecommit.cn-northwest-1.amazonaws.com .cn	SHA256	`wqjd6eHd0+mOBx+dCNuL0omUoCNjaDtZiEpWj5TmCfQ`
git-codecommit.eu-south-1.amazonaws.com	MD5	`b9:f6:5d:e2:48:92:3f:a9:37:1e:c4:d0:32:0e:fb:11`
git-codecommit.eu-south-1.amazonaws.com	SHA256	`lyXrWbCg3uQmJrl1XxB/ASR7ugW1Ysf5yzYOJbudHsI`
git-codecommit.ap-northeast-3.amazonaws.com	MD5	`25:17:40:da:b9:d4:18:c3:b6:b3:fb:ed:1c:20:fe:29`
git-codecommit.ap-northeast-3.amazonaws.com	SHA256	`2B815B9F0AvwLnRxSVxUz4kDYmtEQUGGdQYP8OQLXhA`

Server	Kryptografischer Hash-Typ	Fingerabdruck
git-codecommit.af-south-1.amazonaws.com	MD5	`21:a0:ba:d7:c1:d1:b5:39:98:8d:4d:7c:96:f5:ca:29`
git-codecommit.af-south-1.amazonaws.com	SHA256	`C34ji3x/cnsDZjUpyNGXdE5pjHYimqJrQZ3leTgqJHM`
git-codecommit.il-central-1.amazonaws.com	MD5	`04:74:89:16:98:7a:61:b1:69:46:42:3c:d1:b4:ac:a9`
git-codecommit.il-central-1.amazonaws.com	SHA256	`uFxhp51kUWhleTLeYbxQVYm4RnNLNZ5Dbdm1cgdSl/8`

Verwendung AWS CodeCommit mit VPC-Endpunkten mit Schnittstelle

Wenn Sie Amazon Virtual Private Cloud (Amazon VPC) zum Hosten Ihrer AWS Ressourcen verwenden, können Sie eine private Verbindung zwischen Ihrer VPC und herstellen. CodeCommit Sie können diese Verbindung verwenden, CodeCommit um mit Ihren Ressourcen auf Ihrer VPC zu kommunizieren, ohne das öffentliche Internet nutzen zu müssen.

Amazon VPC ist ein AWS Service, mit dem Sie AWS Ressourcen in einem von Ihnen definierten virtuellen Netzwerk starten können. Mit einer VPC haben Sie die Kontrolle über Ihre Netzwerkeinstellungen, wie IP-Adressbereich, Subnetze, Routing-Tabellen und Netzwerk-Gateways. Bei VPC-Endpunkten wird das Routing zwischen der VPC und den AWS Diensten vom AWS Netzwerk abgewickelt, und Sie können IAM-Richtlinien verwenden, um den Zugriff auf Dienstressourcen zu steuern.

Um Ihre VPC zu verbinden CodeCommit, definieren Sie einen VPC-Schnittstellen-Endpunkt für. CodeCommit Ein Schnittstellenendpunkt ist eine elastic network interface mit einer privaten IP-Adresse, die als Einstiegspunkt für Datenverkehr dient, der für einen unterstützten AWS Dienst bestimmt ist. Der Endpunkt bietet zuverlässige, skalierbare Konnektivität, CodeCommit ohne dass

ein Internet-Gateway, eine NAT-Instanz (Network Address Translation) oder eine VPN-Verbindung erforderlich sind. Weitere Informationen finden Sie unter Was ist Amazon VPC im Benutzerhandbuch zu Amazon VPC.

> **Note**
>
> Andere AWS Services, die VPC-Unterstützung bieten und sich integrieren lassen CodeCommit, wie z. B. AWS CodePipeline, unterstützen möglicherweise nicht die Verwendung von Amazon VPC-Endpunkten für diese Integration. Beispielsweise kann der Verkehr zwischen CodePipeline und CodeCommit nicht auf den VPC-Subnetzbereich beschränkt werden. Dienste, die Integration unterstützen, wie z. B. AWS Cloud9, benötigen möglicherweise zusätzliche Dienste wie. AWS Systems Manager

Schnittstelle, auf der VPC-Endpunkte basieren AWS PrivateLink, eine AWS Technologie, die private Kommunikation zwischen AWS Diensten über eine elastic network interface mit privaten IP-Adressen ermöglicht. Weitere Informationen finden Sie unter. AWS PrivateLink

Die folgenden Schritte sind für Benutzer von Amazon VPC vorgesehen. Weitere Informationen finden Sie unter Erste Schritte im Amazon VPC Benutzerhandbuch.

Verfügbarkeit

CodeCommit unterstützt derzeit VPC-Endpunkte in den folgenden Bereichen: AWS-Regionen

- US East (Ohio)
- USA Ost (Nord-Virginia)
- USA West (Nordkalifornien)
- US West (Oregon)
- Europa (Irland)
- Europe (London)
- Europa (Paris)
- Europa (Frankfurt)
- Europa (Stockholm)
- Europa (Milan)
- Afrika (Kapstadt)

- Israel (Tel Aviv)
- Asien-Pazifik (Tokio)
- Asien-Pazifik (Singapur)
- Asien-Pazifik (Sydney)
- Asien-Pazifik (Jakarta)
- Naher Osten (VAE)
- Asien-Pazifik (Seoul)
- Asien-Pazifik (Osaka)
- Asien-Pazifik (Mumbai)
- Asien-Pazifik (Hyderabad)
- Asien-Pazifik (Hongkong)
- Südamerika (São Paulo)
- Naher Osten (Bahrain)
- Kanada (Zentral)
- China (Peking)
- China (Ningxia)
- AWS GovCloud (US-West)
- AWS GovCloud (US-Ost)

VPC-Endpoints erstellen für CodeCommit

Um mit der Verwendung CodeCommit mit Ihrer VPC zu beginnen, erstellen Sie einen VPC-Schnittstellen-Endpunkt für. CodeCommit CodeCommiterfordert separate Endpunkte für Git-Operationen und CodeCommit API-Operationen. Je nach Ihren geschäftlichen Anforderungen müssen Sie möglicherweise mehr als einen VPC-Endpunkt erstellen. Wenn Sie einen VPC-Endpunkt für erstellen CodeCommit, wählen Sie AWS Services und unter Service Name eine der folgenden Optionen aus:

- com.amazonaws. ***region*** .git-codecommit: Wählen Sie diese Option, wenn Sie einen VPC-Endpunkt für Git-Operationen mit Repositorys erstellen möchten. CodeCommit Wählen Sie diese Option beispielsweise, wenn Ihre Benutzer einen Git-Client und Befehle wie`git pull`,`git commit`, verwenden und `git push` wenn sie mit CodeCommit Repositorys interagieren.

- com.amazonaws. ***Region.*** git-codecommit-fips: Wählen Sie diese Option, wenn Sie einen VPC-Endpunkt für Git-Operationen mit CodeCommit Repositorys erstellen möchten, der dem US-Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2 entspricht.

  > **Note**
  >
  > FIPS-Endpunkte für Git sind nicht in allen AWS Regionen verfügbar. Weitere Informationen finden Sie unter Endpunkte für Git-Verbindungen.

- com.amazonaws. ***region*** .codecommit: Wählen Sie diese Option, wenn Sie einen VPC-Endpunkt für CodeCommit API-Operationen erstellen möchten. Wählen Sie diese Option beispielsweise, wenn Ihre Benutzer die AWS CLI CodeCommit API oder die AWS SDKs verwenden, um mit CodeCommit Vorgängen wie`CreateRepository`, und zu interagieren. `ListRepositories` `PutFile`
- com.amazonaws. ***region*** .codecommit-fips: Wählen Sie diese Option, wenn Sie einen VPC-Endpunkt für CodeCommit API-Operationen erstellen möchten, der dem US-Regierungsstandard Federal Information Processing Standard (FIPS) Publication 140-2 entspricht.

  > **Note**
  >
  > FIPS-Endpunkte sind nicht in allen Regionen verfügbar. AWS Weitere Informationen finden Sie unter dem Eintrag AWS CodeCommit in Federal Information Processing Standard (FIPS) 140-2 Overview.

Erstellen Sie eine VPC-Endpunktrichtlinie für CodeCommit

Sie können eine Richtlinie für Amazon VPC-Endpunkte erstellen, für die Sie CodeCommit Folgendes angeben können:

- Prinzipal, der die Aktionen ausführen kann.
- Aktionen, die ausgeführt werden können
- Ressourcen, für die Aktionen ausgeführt werden können

Beispiel: Ein Unternehmen möchte möglicherweise den Zugriff auf Repositorys auf den Netzwerkadressbereich einer VPC einschränken. Ein Beispiel für diese Art von Richtlinie finden Sie hier: Beispiel 3: Erlaubt einem Benutzer, der von einem bestimmten IP-Adressbereich aus eine

Verbindung herstellt, Zugriff auf ein Repository . Das Unternehmen konfigurierte zwei Git-VPC-Endpunkte für die Region USA Ost (Ohio): `com.amazonaws.us-east-2.codecommit` und. `com-amazonaws.us-east-2.git-codecommit-fips` Sie möchten Code-Pushs an ein CodeCommit Repository zulassen, das *MyDemoRepo*nur auf dem FIPS-konformen Endpunkt benannt ist. Um dies zu erzwingen, würden die Benutzer auf dem Endpunkt `com.amazonaws.us-east-2.codecommit` eine Richtlinie ähnlich der folgenden konfigurieren, die speziell Git-Push-Aktionen verweigert:

```
{
    "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "codecommit:GitPush",
            "Effect": "Deny",
            "Resource": "arn:aws:codecommit:us-west-2:123456789012:MyDemoRepo",
            "Principal": "*"
        }
    ]
}
```

Weitere Informationen finden Sie unter Erstellen eines Schnittstellenendpunkts im Amazon VPC Leitfaden.

Kontingente in AWS CodeCommit

In der folgenden Tabelle werden Kontingente in beschrieben CodeCommit. Informationen zu Kontingenten, die geändert werden können, finden Sie unter AWS CodeCommit -Endpunkte und Kontingente. Informationen zur Beantragung einer Erhöhung der Service Quotas finden Sie unter AWS Servicekontingenten. Hinweise zu den erforderlichen Versionen von Git und anderer Software finden Sie unter Kompatibilität für CodeCommit, Git und anderen Komponenten.

Namen für Genehmigungsregeln und Genehmigungsregelvorlagen	Beliebige Kombination aus Buchstaben, Ziffern, Punkten, Leerzeichen, Unterstrichen und Bindestrichen und mit einer Länge zwischen 1 und 100 Zeichen. Bei den Namen muss die

	Groß- und Kleinschreibung beachtet werden. Namen dürfen nicht auf .git enden und keines folgenden Zeichen enthalten: ! ? @ # $ % ^ & * ( ) + = { } [ ] | \ / > < ~ ` ' " ; :
Länge des Inhalts von Genehmigungsregeln	3 000 Zeichen
Länge der Beschreibung einer Genehmigungsregelvorlage	1 000 Zeichen
Zielverweise für Genehmigungsregelvorlagen	100
Genehmigungsregelvorlagen	1000 in einem AWS-Region
Genehmigungsregeln für eine Pull-Anforderung	Maximal 30. 25 davon können aus Genehmigungsregelvorlagen stammen.
Genehmigungsregeln für eine Pull-Anforderung, die aus einer Genehmigungsregelvorlage erstellt werden	25
Genehmigungen für eine Pull-Anforderung	200
Genehmiger in einem Genehmigungs-Pool	50

Branch-Namen	Jede Kombination von zulässigen Zeichen mit einer Länge zwischen 1 und 256 Zeichen, mit der Ausnahme, dass Zweignamen mit genau 40 Hexadezimalzeichen nicht zulässig sind. Branch-Namen dürfen nicht: • mit einem Schrägstrich (`/`) oder Punkt (`.`) anfangen oder enden. • nur das einzelne Zeichen @ aufweisen. • zwei oder mehr aufeinanderfolgende Punkte (`..`), Schrägstriche (`//`) oder die folgende Zeichenkombination enthalten: `@{` • Leerzeichen oder eines der folgenden Zeichen enthalten: `? ^ * [ \ ~ :` Branch-Namen sind Referenzen. Viele der Einschränkungen für Branch-Namen basieren auf dem Git-Referenzstandard. Weitere Informationen finden Sie unter Git Internals und git-check-ref-format.
Länge des Kommentars	Maximal 10.240 Zeichen.
Benutzerdefinierte Daten für Auslöser	Dieses Zeichenfolgenfeld darf maximal 1 000 Zeichen enthalten. Es kann nicht zur Übergabe dynamischer Parameter verwendet werden.
Anzeige in der Konsole	Eine Datei oder ein Vergleich zwischen Dateien kann in der Konsole möglicherweise nicht angezeigt werden, wenn: • Die Datei ist größer als 2 MB • Die Datei enthält mehr als 25.000 Zeichen in einer einzigen Zeile • Ein Vergleich enthält insgesamt mehr als 6.500 Zeilen mit Unterschieden

E-Mail-Adressen in Commits in der Konsole	Beliebige Kombination aus zulässigen Zeichen und mit einer Länge zwischen 1 und 256 Zeichen. E-Mail-Adressen werden nicht validiert.

Dateipfade	Beliebige Kombination aus zulässigen Zeichen und mit einer Länge zwischen 1 und 4,096 Zeichen. Dateipfade müssen einen eindeutigen Namen haben, in dem die Datei und der genaue Speicherort der Datei angegeben sind. Dateipfade dürfen nicht mehr als 20 Verzeichnisse umfassen. Außerdem gelten für Dateipfade folgende Regeln:
	• Sie dürfen keine leeren Zeichenfolgen enthalten.
	• Sie dürfen kein relativer Dateipfad sein.
	• Sie dürfen keine der folgenden Zeichenkombinationen enthalten:
	`/./`
	`/../`
	`//`
	• Sie dürfen nicht mit einem Schrägstrich oder umgekehrten Schrägstrich (Backslash) enden.
	
	Dateinamen und -pfade müssen vollständig qualifiziert sein. Der Dateiname und der Pfad zu einer Datei auf Ihrem lokalen Computer müssen den Standards für das Betriebssystem entsprechen. Verwenden Sie bei der Angabe des Pfads zu einer Datei in einem CodeCommit Repository die Standards für Amazon Linux.

Dateigröße	Maximal 6 MB für jede einzelne Datei bei Verwendung der CodeCommit Konsole, APIs oder der AWS CLI.
Größe des Git-Blobs	Maximal 2 GB **Note** Die Anzahl oder die Gesamtgröße aller Dateien in einem einzelnen Commit ist nicht begrenzt, sofern die Metadaten den Maximalwert von 6 MB nicht übersteigen und ein einzelner Blob nicht größer ist als 2 GB.
Grafische Branch-Darstellung im Commit Visualizer	35 pro Seite. Falls sich mehr als 35 Branches auf einer einzelnen Seite befinden, wird keine Grafik angezeigt.
Metadaten für Commit	Maximal 20 MB für die kombinierten Metadaten für einen Commit (z. B. die Kombination aus Autoreninformationen, Datum, übergeordneter Commit-Liste und Commit-Nachrichten) bei Verwendung der CodeCommit Konsole, der APIs oder der AWS CLI. **Note** Es gibt keine Begrenzung für die Anzahl oder die Gesamtgröße aller Dateien in einem einzigen Commit, solange die Metadaten 6 MB nicht überschreiten, eine einzelne Datei 6 MB nicht überschreitet und ein einzelner Blob 2 GB nicht überschreitet.

Anzahl der Dateien in einem Commit	Maximal 100.
Anzahl der offenen Pull-Requests	Maximal 1.000.
Anzahl Referenzen für einzelnen Push	Maximal 4 000 (einschließlich Erstellen, Löschen und Aktualisieren). Für die Gesamtanzahl der Referenzen im Repository ist keine Einschränkung vorhanden.
Anzahl der Repositorys	Maximal 5.000 pro Amazon Web Services Services-Konto. Dieses Limit kann geändert werden. Weitere Informationen finden Sie unter AWS CodeCommit Endpunkte und Kontingente und AWS Service Quotas.
Anzahl Auslöser im Repository	Maximal 10

Regionen	CodeCommit ist in den folgenden AWS-Regionen Sprachen verfügbar: • US East (Ohio) • USA Ost (Nord-Virginia) • USA West (Nordkalifornien) • US West (Oregon) • Europa (Irland) • Europe (London) • Europa (Paris) • Europa (Frankfurt) • Europa (Stockholm) • Europa (Milan) • Afrika (Kapstadt) • Israel (Tel Aviv) • Asien-Pazifik (Tokio) • Asien-Pazifik (Singapur) • Asien-Pazifik (Sydney) • Asien-Pazifik (Jakarta) • Naher Osten (VAE) • Asien-Pazifik (Seoul) • Asien-Pazifik (Osaka) • Asien-Pazifik (Mumbai) • Asien-Pazifik (Hyderabad) • Asien-Pazifik (Hongkong) • Südamerika (São Paulo) • Naher Osten (Bahrain) • Kanada (Zentral) • China (Peking) • China (Ningxia) • AWS GovCloud (US-West)

	• AWS GovCloud (US-Ost) Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.
Repository-Beschreibungen	Beliebige Zeichenkombination und mit einer Länge zwischen 0 und 1 000 Zeichen. Repository-Beschreibungen sind optional.
Repository-Namen	Beliebige Kombination aus Buchstaben, Ziffern, Punkten, Unterstrichen und Bindestrichen und mit einer Länge zwischen 1 und 100 Zeichen. Bei den Namen muss die Groß- und Kleinschreibung beachtet werden. Repository-Namen dürfen nicht auf .git enden und keines folgenden Zeichen enthalten: `! ? @ # $ % ^ & * ( ) + = { } [ ] | \ / > < ~ ` ' " ; :`
Namen von Repository-Tag-Schlüsseln	Beliebige Kombination aus Unicode-Buchstaben, Zahlen, Leerstellen und zulässigen UTF-8-Zeichen mit einer Länge zwischen 1 und 128 Zeichen. Zulässige Zeichen sind `+ - = . _ : / @` Tag-Schlüsselnamen müssen eindeutig sein. Jeder Schlüssel darf nur einen Wert haben. Ein Tag darf nicht: • mit `aws:` beginnen • nur aus Leerstellen bestehen • mit einem Leerzeichen enden • Emojis oder eines der folgenden Zeichen enthalten: `? ^ * [ \ ~ ! # $ % & * ( ) > < | " ' ` [ ] { } ;`

Repository-Tag-Werte	Beliebige Kombination aus Unicode-Buchstaben, Zahlen, Leerstellen und zulässigen UTF-8-Zeichen mit einer Länge zwischen 1 und 256 Zeichen. Zulässige Zeichen sind `+ - = . _ : / @` Ein Schlüssel darf nur einen Wert haben, viele Schlüssel können aber dasselbe Tag aufweisen . Ein Tag darf nicht: • nur aus Leerstellen bestehen • mit einem Leerzeichen enden • Emojis oder eines der folgenden Zeichen enthalten: `? ^ * [ \ ~ ! # $ % & * ( ) > < | " ' ` [ ] { } ;`
Repository-Tags	Bei Tags muss die Groß- und Kleinschreibung beachtet werden. Maximal 50 pro Ressource. Tagnamen mit genau 40 Hexadezimalzeichen sind nicht zulässig.
Auslösernamen	Beliebige Kombination aus Buchstaben, Ziffern, Punkten, Unterstrichen und Bindestrichen und mit einer Länge zwischen 1 und 100 Zeichen. Auslösernamen dürfen weder Leerzeichen noch Kommata enthalten.
Benutzernamen in Commits in der Konsole	Beliebige Kombination aus zulässigen Zeichen und mit einer Länge zwischen 1 und 1,024 Zeichen.

AWS CodeCommit Befehlszeilenreferenz

Anhand dieser Referenz können Sie leichter die Verwendung der AWS CLI erlernen.

Um das zu installieren und zu konfigurieren AWS CLI

1. Laden Sie auf Ihrem lokalen Computer den herunter und installieren Sie ihn AWS CLI. Dies ist eine Voraussetzung für die Interaktion mit CodeCommit über die Befehlszeile. Wir empfehlen, AWS CLI Version 2 zu installieren. Es ist die neueste Hauptversion von AWS CLI und unterstützt alle aktuellen Funktionen. Es ist die einzige Version von AWS CLI , die die Verwendung eines Root-Kontos, Verbundzugriff oder temporärer Anmeldeinformationen mit git-remote-codecommit unterstützt.

   Weitere Informationen finden Sie unter Einrichtung der AWS Befehlszeilenschnittstelle.

   > **Note**
   >
   > CodeCommit funktioniert nur mit den AWS CLI Versionen 1.7.38 und höher. Es hat sich bewährt, die neueste verfügbare Version zu installieren oder AWS CLI auf sie zu aktualisieren. Führen AWS CLI Sie den aws --version Befehl aus, um festzustellen, welche Version von Sie installiert haben.
   > Informationen zum Upgrade einer älteren Version von AWS CLI auf die neueste Version finden Sie unter Installation von AWS Command Line Interface.

2. Führen Sie diesen Befehl aus, um zu überprüfen, ob die CodeCommit Befehle für installiert AWS CLI sind.

   ```
   aws codecommit help
   ```

   Dieser Befehl gibt eine Liste von CodeCommit Befehlen zurück.

3. Konfigurieren Sie das AWS CLI mit einem Profil, indem Sie den configure Befehl verwenden, wie folgt:.

   ```
   aws configure
   ```

   Wenn Sie dazu aufgefordert werden, geben Sie den AWS Zugriffsschlüssel und den AWS geheimen Zugriffsschlüssel des IAM-Benutzers an, mit CodeCommit dem Sie ihn verwenden möchten. Stellen Sie außerdem sicher, dass Sie angeben, AWS-Region wo sich das Repository befindet, z. B. `us-east-2` Wenn Sie nach dem standardmäßigen Ausgabeformat gefragt werden, geben Sie `json` an. Wenn Sie beispielsweise ein Profil für einen IAM-Benutzer konfigurieren:

```
AWS Access Key ID [None]: Type your IAM user AWS access key ID here, and then press
 Enter
AWS Secret Access Key [None]: Type your IAM user AWS secret access key here, and
 then press Enter
Default region name [None]: Type a supported region for CodeCommit here, and then
 press Enter
Default output format [None]: Type json here, and then press Enter
```

Weitere Informationen zum Erstellen und Konfigurieren von Profilen zur Verwendung mit dem AWS CLI finden Sie im Folgenden:

- Benannte Profile
- Verwenden einer IAM-Rolle in der AWS CLI
- Befehl „Set“
- Verbindung zuAWS CodeCommit Repositorys mit wechselnden Anmeldeinformationen herstellen

Um eine Verbindung zu einem Repository oder einer Ressource in einem anderen Repository herzustellen AWS-Region, müssen Sie das AWS CLI mit dem Standardregionsnamen neu konfigurieren. Zu den unterstützten Standardregionsnamen für CodeCommit gehören:

- us-east-2
- us-east-1
- eu-west-1
- us-west-2
- ap-northeast-1
- ap-southeast-1
- ap-southeast-2
- ap-southeast-3
- me-central-1
- eu-central-1
- ap-northeast-2
- sa-east-1
- us-west-1

- eu-west-2
- ap-south-1
- ap-south-1
- ca-central-1
- us-gov-west-1
- us-gov-east-1
- eu-north-1
- ap-east-1
- me-south-1
- cn-north-1
- cn-northwest-1
- eu-south-1
- ap-northeast-3
- af-south-1
- il-central-1

Weitere Informationen zu CodeCommit und finden AWS-Region Sie unterRegionen und Git-Verbindungsendpunkte. Weitere Informationen zu IAM, Zugriffsschlüsseln und geheimen Schlüsseln finden Sie unter Wie erhalte ich Anmeldeinformationen? und Zugriffsschlüssel für IAM-Benutzer verwalten. Weitere Informationen zu den Profilen AWS CLI und finden Sie unter Benannte Profile.

Führen Sie den folgenden Befehl aus, um eine Liste aller verfügbaren CodeCommit Befehle anzuzeigen:

```
aws codecommit help
```

Um Informationen zu einem CodeCommit Befehl anzuzeigen, führen Sie den folgenden Befehl aus, wobei *Befehlsname* der Name des Befehls ist (z. B.create-repository):

```
aws codecommit command-name help
```

Im Folgenden finden Sie Beschreibungen und beispielhafte Anwendungen der Befehle in der AWS CLI:

- associate-approval-rule-template-mit-Repository
- batch-associate-approval-rule-template-with-repositories
- batch-disassociate-approval-rule-template-from-repositories
- batch-describe-merge-conflicts
- batch-get-commits
- batch-get-repositories
- create-approval-rule-template
- Zweig erstellen
- create-commit
- create-pull-request
- create-pull-request-approval-Regel
- Projektarchiv erstellen
- create-unreferenced-merge-commit
- delete-approval-rule-template
- delete-branch
- delete-comment-content
- delete-file
- Projektarchiv löschen
- describe-merge-conflicts
- delete-pull-request-approval-Regel
- describe-pull-request-events
- disassociate-pull-request-approval-rule-template-from-repository
- evaluate-pull-request-approval-Regeln
- get-approval-rule-template
- get-blob
- Abzweigung abrufen
- get-comment
- get-comment-reactions

- get-comments-for-compared-verpflichten
- get-comments-for-pull-Anfrage
- get-commit
- get-differences
- get-merge-commit
- get-merge-conflicts
- get-merge-options
- get-pull-request
- get-pull-request-approval-Staaten
- get-pull-request-override-Staat
- Repository abrufen
- get-repository-triggers
- list-approval-rule-templates
- list-associated-approval-rule-templates-for-repository
- Zweige auflisten
- list-pull-requests
- Listen-Repositorien
- list-repositories-for-approval-Regelvorlage
- list-tags-for-resource
- merge-branches-by-fast-vorwärts
- merge-branches-by-squash
- merge-branches-by-three-Weg
- merge-pull-request-by-schnell vorwärts
- merge-pull-request-by-Kürbis
- merge-pull-request-by-Dreiweg
- override-pull-request-approval-Regeln
- post-comment-for-compared-verpflichten
- post-comment-for-pull-Anfrage
- post-comment-reply
- put-comment-reaction

- put-file
- put-repository-triggers
- tag-resource
- test-repository-triggers
- untag-resource
- update-approval-rule-template-Inhalt
- update-approval-rule-template-Beschreibung
- update-approval-rule-template-name
- update-comment
- update-default-branch
- update-pull-request-approval-Regelinhalt
- update-pull-request-approval-Staat
- update-pull-request-description
- update-pull-request-status
- update-pull-request-title
- update-repository-description
- update-repository-name

Grundlegende Git-Befehle

Du kannst Git verwenden, um mit einem lokalen Repo und dem CodeCommit Repository zu arbeiten, mit dem du das lokale Repo verbunden hast.

Es folgen einige grundlegende Beispiele für häufig verwendete Git-Befehle.

Weitere Optionen findest du in deiner Git-Dokumentation.

Themen

- Konfigurationsvariablen
- Remote-Repositorys
- Commits
- Branches
- Tags

Konfigurationsvariablen

Listet alle Konfigurationsvariablen auf.	`git config --list`
Listet nur lokale Konfigurationsvariablen auf.	`git config --local -l`
Listet nur Systemkonfigurationsvariablen auf.	`git config --system -l`
Listet nur globale Konfigurationsvariablen auf.	`git config --global -l`
Legt eine Konfigurationsvariable in der angegebenen Konfigurationsdatei fest.	`git config [--local | --global | --system] variable-name variable-value`
Setzt den Standard-Branch-Namen für alle lokalen Repositorys auf main, wenn ein erster Commit in ein Repository erfolgt, das noch keinen Standard-Branch hat	git config --global init.defaultBranch main
Bearbeitet eine Konfigurationsdatei direkt. Kann auch verwendet werden, um den Speicherort einer bestimmten Konfigurationsdatei zu ermitteln. Zum Verlassen des Bearbeitungsmodus geben Sie normalerweise `:q` (Beenden ohne Änderungen zu speichern) oder `:wq` (Änderungen speichern und dann beenden) ein und drücken die Eingabetaste.	`git config [--local | --global | --system] --edit`

Remote-Repositorys

Initialisiert ein lokales Repo als Vorbereitung für die Verbindung mit einem Repository. CodeCommit	`git init`
Kann verwendet werden, um eine Verbindung zwischen einem lokalen Repo und einem Remote-Repository (z. B. einem CodeCommit	`git remote add remote-name remote-url`

Repository) unter Verwendung des angegeben en Nicknamens, den das lokale Repo für das Repository hat, und der angegebenen URL zum CodeCommit Repository einzurichten. CodeCommit	
Erstellt ein lokales Repo, indem eine Kopie eines CodeCommit Repositorys unter der angegebenen URL im angegebenen Unterordn er des aktuellen Ordners auf dem lokalen Computer erstellt wird. Dieser Befehl erstellt außerdem einen Remote-Tracking-Zweig für jeden Branch im geklonten CodeCommit Repository und erstellt einen ersten Branch, der vom aktuellen Standardbranch im geklonten Repository abgezweigt ist, und checkt ihn aus. CodeCommit	`git clone` *`remote-url`* *`local-sub folder-name`*
Zeigt den Spitznamen an, den das lokale Repository für das Repository verwendet. CodeCommit	`git remote`
Zeigt den Spitznamen und die URL an, die das lokale Repo für Abrufe und Pushs in das Repository verwendet. CodeCommit	`git remote -v`
Leitet abgeschlossene Commits vom lokalen Repo in das Repository weiter und verwendet dabei den angegebenen Spitznamen, den das lokale Repo für das CodeCommit Repository und den angegebenen Branch hat. CodeCommit Außerdem werden Upstream-Tracking-Informationen für das lokale Repositor y während des Push-Vorgangs eingerichtet.	`git push -u` *`remote-name`* *`branch-na me`*

Überträgt abgeschlossene Commits aus dem lokalen Repository in das CodeCommit Repository, nachdem die Upstream-Tracking-Informationen festgelegt wurden.	`git push`
Ruft abgeschlossene Commits an das lokale Repository aus dem Repository ab und verwendet dabei den angegebenen Spitzname n, den das lokale CodeCommit Repository für das Repository und den angegebenen Branch hat CodeCommit	`git pull` *`remote-name`* *`branch-name`*
Ruft abgeschlossene Commits an das lokale Repository aus dem Repository ab, nachdem die Upstream-Tracking-Informationen festgeleg t wurden. CodeCommit	`git pull`
Trennt das lokale Repo vom CodeCommi t Repository und verwendet dabei den angegebenen Spitznamen, den das lokale Repo für das Repository hat. CodeCommit	`git remote rm` *`remote-name`*

Commits

Zeigt an, was dem schwebenden Commit im lokalen Repository hinzugefügt wurde oder was nicht.	`git status`
Zeigt in einem übersichtlichen Format an, was zum ausstehenden Commit im lokalen Repo hinzugefügt wurde oder nicht. (`M` = geändert, `A` = hinzugefügt, `D` = gelöscht usw.)	`git status -sb`

Zeigt Änderungen zwischen dem ausstehenden Commit und dem letzten Commit im lokalen Repository an.	`git diff HEAD`
Fügt dem ausstehenden Commit im lokalen Repository bestimmte Dateien hinzu.	`git add [file-name-1 file-name-2 file-name-N | file-pattern ]`
Fügt dem schwebenden Commit im lokalen Repository alle neuen, geänderten und gelöschten Dateien hinzu.	`git add`
Beginnt mit dem Finalisieren des ausstehen den Commits im lokalen Repository, das einen Editor mit einer Commit-Nachricht anzeigt. Nachdem die Mitteilung eingegeben wurde, wird der schwebende Commit finalisiert.	`git commit`
Finalisiert den ausstehenden Commit im lokalen Repository, einschließlich der gleichzei tigen Angabe einer Commit-Nachricht.	`git commit -m "Some meaningful commit comment"`
Listet die letzten Commits im lokalen Repo auf.	`git log`
Listet die letzten Commits im lokalen Repositor y in einem Grafikformat auf.	`git log --graph`
Listet die letzten Commits im lokalen Repositor y in einem vordefinierten komprimierten Format auf.	`git log --pretty=oneline`
Listet die letzten Commits im lokalen Repositor y in einem vordefinierten komprimierten Format mit einem Diagramm auf.	`git log --graph --pretty=oneline`

Listet die letzten Commits im lokalen Repository in einem benutzerdefinierten Format mit einem Diagramm auf. (Weitere Optionen siehe Git-Grundlagen – Anzeigen des Commit-Verlaufs)	`git log --graph --pretty=format:"%H (%h) : %cn : %ar : %s"`

Branches

Listet alle Branches im lokalen Repository auf, wobei neben Ihrem aktuellen Branch ein Sternchen (*) angezeigt wird.	`git branch`
Ruft Informationen über alle vorhandenen Branches im Repository in das CodeCommit lokale Repository ab.	`git fetch`
Listet alle Zweige im lokalen Repository und alle Remote-Tracking-Zweige im lokalen Repo auf.	`git branch -a`
Listet nur Remote-Tracking-Zweige im lokalen Repo auf.	`git branch -r`
Erstellt einen neuen Zweig im lokalen Repository unter Verwendung des angegebenen Zweignamens.	`git branch new-branch-name`
Wechselt unter Verwendung des angegebenen Zweignamens zu einem anderen Zweig im lokalen Repository.	`git checkout other-branch-name`
Erstellt einen neuen Zweig im lokalen Repository unter Verwendung des angegebenen Zweignamens und wechselt dann zu diesem.	`git checkout -b new-branch-name`

Verschiebt eine neue Verzweigung vom lokalen Repository in das Repository, wobei der angegebene Nickname, den das lokale CodeCommit Repository für das Repository hat, und der CodeCommit angegebene Branch-Name verwendet wird. Außerdem werden während des Push-Vorgangs Upstream-Tracking-Informationen für den Branch im lokalen Repository eingerichtet.	`git push -u remote-name new-branch-name`
Erstellt einen neuen Zweig im lokalen Repository unter Verwendung des angegebenen Zweignamens. Verbindet dann den neuen Branch im lokalen Repo mit einem vorhandenen Branch im CodeCommit Repository und verwendet dabei den angegebenen Spitznamen, den das lokale Repository für das CodeCommit Repository hat, und den angegebenen Branch-Namen.	`git branch --track new-branch-name remote-name /remote-branch-name`
Führt Änderungen von einem anderen Zweig im lokalen Repository mit dem aktuellen Zweig im lokalen Repository zusammen.	`git merge from-other-branch-name`
Löscht einen Zweig im lokalen Repository, es sei denn, er enthält Werke, die noch nicht zusammengeführt wurden.	`git branch -d branch-name`
Löscht einen Branch im CodeCommit Repository unter Verwendung des angegebenen Nicknamens, den das lokale Repository für das CodeCommit Repository hat, und des angegebenen Branch-Namens. (Beachten Sie die Verwendung des Doppelpunkts (`:`).)	`git push remote-name :branch-name`

Tags

Listet alle Tags im lokalen Repo auf.	`git tag`
Zieht alle Tags aus dem CodeCommit Repository in das lokale Repo.	`git fetch --tags`
Zeigt Informationen zu einem bestimmten Tag im lokalen Repo an.	`git show` *`tag-name`*
Erzeugt ein „Lightweight“ -Tag im lokalen Repo.	`git tag` *`tag-name commit-id-to-point-tag-at`*
Verschiebt ein bestimmtes Tag aus dem lokalen Repository in das CodeCommi t Repository und verwendet dabei den angegebenen Spitznamen, den das lokale Repo für das CodeCommit Repository hat, und den angegebenen Tagnamen.	`git push` *`remote-name tag-name`*
Verschiebt alle Tags aus dem lokalen Repo in das CodeCommit Repository und verwendet dabei den angegebenen Spitznamen, den das lokale Repo für das Repository hat. CodeCommit	`git push` *`remote-name`* `--tags`
Löscht ein Tag im lokalen Repo.	`git tag -d` *`tag-name`*
Löscht ein Tag im CodeCommit Repositor y unter Verwendung des angegebenen Nicknamens, den das lokale Repo für das CodeCommit Repository hat, und des angegebenen Tag-Namens. (Beachten Sie die Verwendung des Doppelpunkts (`:`).)	`git push` *`remote-name`* `:`*`tag-name`*

Dokumentverlauf für das AWS CodeCommit-Benutzerhandbuch

In der folgenden Tabelle werden wichtige Änderungen an der Dokumentation für CodeCommit beschrieben. Um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten, können Sie einen RSS-Feed abonnieren.

- API-Version: 2015-04-13

Änderung	Beschreibung	Datum
CodeCommit unterstützt jetzt die Verwendung von vom Kunden verwalteten Schlüsseln	Sie können jetzt einen vom Kunden verwalteten Schlüssel oder einen Von AWS verwalteter Schlüssel zum Verschlüsseln und Entschlüsseln von Daten in einem Repository verwenden. Weitere Informationen finden Sie unter Verschlüsselung, Repository erstellen AWS KMS und Repository-Einstellungen ändern.	21. Dezember 2023
CodeCommit ist jetzt in Israel (Tel Aviv) erhältlich;	CodeCommit ist jetzt in Israel (Tel Aviv) erhältlich. Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.	28. August 2023
Änderungen an verwalteten Richtlinien für CodeCommit	Die AWSCodeCommitFullAccess Richtlinien AWSCodeCommitPowerUser und wurden mit einer zusätzlichen Genehmigung aktualisiert. Weitere Informationen finden	16. Mai 2023

	Sie unter CodeCommit - Aktualisierungen von AWS-verwalteten Richtlinien.	
CodeCommit ist jetzt in drei weiteren Versionen erhältlich AWS-Regionen	CodeCommit ist jetzt in drei weiteren Ländern erhältlic hAWS-Regionen: Asien-Pazifik (Jakarta), Naher Osten (VAE) und Asien-Paz ifik (Hyderabad). Weitere Informationen finden Sie unter Regionen und Git-Verbi ndungsendpunkte.	28. Februar 2023
CodeCommit ist jetzt in Afrika (Kapstadt) erhältlich	CodeCommit ist jetzt in einer weiteren Version erhältlichAWS-Region: Afrika (Kapstadt). Weitere Informationen finden Sie unter Regionen und Git-Verbi ndungsendpunkte.	15. September 2021
Änderungen an verwalteten Richtlinien für CodeCommit	Einzelheiten zu Aktualisi erungen der AWS verwaltet en Richtlinien für CodeCommi t sind jetzt verfügbar. Weitere Informationen finden Sie unter CodeCommit -Aktualis ierungen von AWS-verwa lteten Richtlinien.	18. August 2021

CodeCommit ist jetzt im asiatisch-pazifischen Raum (Osaka) erhältlich	CodeCommit ist jetzt in einer weiteren Region erhältlic hAWS-Region: Asien-Pazifik (Osaka). Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.	14. April 2021
AWS CloudFormationund AWS Cloud Development Kit (AWS CDK) ändert das Benennungsverhalten für Standardzweige in CodeCommit	Repositorys, die mit AWS CloudFormation oder AWS CDK mit einem ersten Commit von Code erstellt wurden, verwenden jetzt den Standard-Branch-Namen main. Diese Änderung hat keine Auswirkungen auf bestehende Repositorys oder Branches. Kunden, die lokale Git-Clients verwenden, um ihre ersten Commits zu erstellen, haben einen Standard-Branch-Namen, der der Konfiguration dieser Git-Clients folgt. Weitere Informationen finden Sie unter CodeCommit Ressourcen erstellen mit AWS CloudFormation.	4. März 2021

CodeCommit ändert das Benennungsverhalten für Standardzweige	Seit dem 19. Januar 2021 lautet der Standard-Branch-Name, der durch einen ersten Commit in ein CodeCommit Repository erstellt wurde, main. Diese Änderung hat keine Auswirkungen auf bestehende Repositorys oder Branches. Kunden, die lokale Git-Clients verwenden, um ihre ersten Commits zu erstellen, haben einen Standard-Branch-Namen, der der Konfiguration dieser Git-Clients folgt. Weitere Informationen findest du unter Mit Branches arbeiten, Einen Commit erstellen und Branch-Einstellungen ändern.	19. Januar 2021
CodeCommit ist jetzt in Europa (Mailand) verfügbar	CodeCommit ist jetzt in einer weiteren Version erhältlichAWS-Region: Europa (Mailand). Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.	16. September 2020
CodeCommit fügt Unterstützung für Emoji-Reaktionen auf Kommentare hinzu	CodeCommit unterstützt jetzt das Reagieren auf Kommentare von anderen Benutzern mit Emojis. Weitere Informationen findest du unter Einen Commit kommentieren und einen Pull Request überprüfen.	24. Juni 2020

CodeCommit jetzt in China (Peking) und China (Ningxia) erhältlich	CodeCommit ist jetzt in zwei weiteren Ländern erhältlic hAWS-Regionen: China (Peking) und China (Ningxia) . Weitere Informationen finden Sie unter Regionen und Git-Verbindungsendpunkte.	23. April 2020
CodeCommit fügt Unterstüt zung hinzu für git-remote-codecommit	CodeCommit unterstützt Verbindungen zu CodeCommi t Repositorys über HTTPS mitgit-remote-codecom mit, einem Hilfsprogramm, das Git modifiziert. Dies ist der empfohlene Ansatz für Verbundverbindungen oder temporären Zugriff auf Repositorys. CodeCommit Sie können es auch git-remote-codecommit mit einem IAM-Benutzer verwenden. git-remot e-codecommiterfordert nicht, dass Sie Git-Anmeldeinforma tionen für den Benutzer einrichten. Weitere Informati onen finden Sie unter Einrichtu ngsschritte für HTTPS-Ver bindungen AWS CodeCommit mit git-remote-codecommit.	4. März 2020

CodeCommit unterstützt Sitzungs-Tags	CodeCommit unterstützt die Verwendung von Sitzungs-Tags, bei denen es sich um Schlüssel-Wert-Paa rattribute handelt, die Sie übergeben, wenn Sie eine IAM-Rolle übernehmen, temporäre Anmeldeinformation en verwenden oder einen Benutzer in () verbinden. AWS Security Token Service AWS STS Sie können die Informationen in diesen Tags verwenden, um die Identifiz ierung zu erleichtern, wer eine Änderung vorgenommen hat oder ein Ereignis verursach t hat. Weitere Informationen finden Sie unter CodeCommit-Überwachung und Verwenden von Tags zum Bereitstellen von Identitätsinformationen in CodeCommit.	19. Dezember 2019
CodeCommit ist im asiatisch-pazifischen Raum (Hongkong) verfügbar	Sie können es jetzt CodeCommit im asiatisch-pazifischen Raum (Hongkong) verwenden. Weitere Informati onen über Git-Verbindungsend punkte finden Sie unter Regionen.	11. Dezember 2019

CodeCommit unterstützt Amazon CodeGuru Reviewer	CodeCommit unterstützt Amazon CodeGuru Reviewer, einen automatisierten Code-Review-Service, der Programmanalyse und maschinelles Lernen nutzt, um häufig auftretende Probleme zu erkennen und Korrekturen in Ihrem Java- oder Python-Code zu empfehlen. Weitere Informationen finden Sie unter Ein Repository mit Amazon CodeGuru Reviewer verknüpfen oder die Zuordnung aufheben und Mit Pull Requests arbeiten.	3. Dezember 2019
CodeCommit unterstützt Genehmigungsregeln	Sie können jetzt Genehmigungsregeln verwenden, um Entwicklungs-Workflows über Repositorys hinweg so anzupassen, dass verschiedene Branches über geeignete Genehmigungen und Kontrollen für Pull-Anforderungen verfügen. Weitere Informationen finden Sie unter Working with Approval Rule Templates und Working with Pull Requests.	20. November 2019

CodeCommit unterstützt Benachrichtigungsregeln	Sie können jetzt Benachrichtigungsregeln verwenden, um Benutzer über wichtige Änderungen in Repositorys zu informieren. Diese Funktion ersetzt Benachrichtigungen, die vor dem 5. November 2019 erstellt wurden. Weitere Informationen finden Sie unter Erstellen einer Benachrichtigungsregel.	5. November 2019
CodeCommit ist im Mittleren Osten (Bahrain) verfügbar	Sie können es jetzt CodeCommit im Nahen Osten (Bahrain) verwenden. Weitere Informationen über Git-Verbindungsendpunkte finden Sie unter Regionen.	30. Oktober 2019
CodeCommit fügt Unterstützung für das Abrufen von Informationen über mehrere Commits hinzu	Sie können Informationen zu mehreren Commits abrufen, indem Sie den batch-get-commits Befehl in der AWS CLI verwenden. Weitere Informationen finden Sie unter Anzeigen von Commit-Details.	15. August 2019
CodeCommit ist in Europa (Stockholm) verfügbar	Sie können es jetzt CodeCommit in Europa (Stockholm) verwenden. Weitere Informationen über Git-Verbindungsendpunkte finden Sie unter Regionen.	31. Juli 2019

CodeCommit fügt Unterstützung für das Taggen von Repositorys in der Konsole hinzu CodeCommit	Sie können jetzt Tags für ein Repository hinzufügen, verwalten und entfernen, um Ihre AWS Ressourcen von der CodeCommit Konsole aus zu verwalten. Weitere Informationen finden Sie unter Markieren eines Repositorys.	2. Juli 2019
CodeCommit fügt Unterstützung für zusätzliche Git-Merge-Strategien hinzu	Sie können jetzt zwischen Git-Mergestrategien wählen, wenn Pull-Anforderungen in CodeCommit zusammengeführt werden. Sie können Zusammenführungskonflikte auch in der CodeCommit Konsole lösen. Weitere Informationen finden Sie unter Arbeiten mit Pull-Anforderungen.	10. Juni 2019
CodeCommit ist in AWS GovCloud (USA-Ost) verfügbar	Sie können es jetzt CodeCommit in AWS GovCloud (US-Ost) verwenden. Weitere Informationen über Git-Verbindungsendpunkte finden Sie unter Regionen.	31. Mai 2019
CodeCommit fügt Unterstützung für das Taggen von Repositorys hinzu	Sie können jetzt Tags für ein Repository hinzufügen, verwalten und entfernen, um Ihnen die Verwaltung Ihrer AWS-Ressourcen zu erleichtern. Weitere Informationen finden Sie unter Markieren eines Repositorys.	30. Mai 2019

Suchen Sie in der Konsole nach Ressourcen	Sie können jetzt Schnellsuchen nach Ressourcen, z. B. Repositorys, Build-Projekten, Bereitstellungsanwendungen und Pipelines, ausführen. Wählen Sie Go to Ressource (Zur Ressource) oder drücken Sie die Taste / und geben Sie dann den Namen der Ressource ein. Weitere Informationen finden Sie unter CodeCommit Tutorial.	14. Mai 2019
CodeCommit ist in AWS GovCloud (US-West) verfügbar	Sie können es jetzt CodeCommit in AWS GovCloud (US-West) verwenden. Weitere Informationen über Git-Verbindungsendpunkte finden Sie unter Regionen.	18. April 2019
CodeCommit fügt Unterstützung für Amazon VPC-Endpunkte hinzu	Sie können jetzt eine private Verbindung zwischen Ihrer VPC und CodeCommit herstellen. Weitere Informationen finden Sie unter CodeCommit Mit Interface VPC-Endpoints verwenden.	7. März 2019
CodeCommit fügt eine neue API hinzu	CodeCommit hat eine API zum Erstellen von Commits hinzugefügt. Weitere Informationen finden Sie unter Erstellen eines Commits.	20. Februar 2019

Aktualisierung des Inhalts	Der Inhalt in diesem Handbuch wurde aktualisiert und enthält jetzt kleinere Fehlerbeh ebungen und zusätzliche Anleitungen zur Fehlerbeh ebung.	2. Januar 2019
Aktualisierung des Inhalts	Der Inhalt dieses Handbuchs wurde aktualisiert, um das neue CodeCommit Konsolene rlebnis zu unterstützen.	30. Oktober 2018
CodeCommit und der Federal Information Processing Standard (FIPS)	CodeCommit hat in einigen Regionen Unterstützung für den Regierungsstandard Federal Information Processin g Standard (FIPS) Publicati on 140-2 hinzugefügt. Weitere Informationen über FIPS und FIPS-Endpunkte finden Sie unter Federal Information Processing Standard (FIPS) 140-2 – Übersicht. Weitere Informationen über Git-Verbi ndungsendpunkte finden Sie unter Regionen.	25. Oktober 2018
CodeCommit fügt drei APIs hinzu	CodeCommit hat drei APIs hinzugefügt, um die Arbeit mit Dateien zu unterstüt zen. Weitere Informationen zu Git-Verbindungsend punkten finden Sie unter Berechtigungen für Aktionen für einzelne Dateien und AWS CodeCommitAPI-Referenz.	27. September 2018

CodeCommit Benachrichtigung über den Dokumentationsverlauf als RSS-Feed verfügbar	Sie können jetzt Benachrichtigungen über Aktualisierungen der CodeCommit Dokumentation erhalten, indem Sie einen RSS-Feed abonnieren.	29. Juni 2018

Frühere Aktualisierungen

In der folgenden Tabelle werden die wichtigen Änderungen an der Dokumentation vor dem 29.06.2018 beschrieben.

Änderung	Beschreibung	Datum geändert
Neues Thema	Das Thema Beschränken Sie Pushs und Merges auf Branches wurde hinzugefügt. Das Thema Referenz für CodeCommit-Berechtigungen wurde aktualisiert.	16. Mai 2018
Neuer -Abschnitt	Der Abschnitt Arbeiten mit Dateien inAWS CodeCommitRepositories wurde hinzugefügt. Die Themen Referenz für CodeCommit-Berechtigungen und Erste Schritte mit AWS CodeCommit wurden aktualisiert.	21. Februar 2018
Neues Thema	Das Thema Konfiguriere den kontoübergreifenden Zugriff auf ein AWS CodeCommit Repository mithilfe von Rollen wurde hinzugefügt.	21. Februar 2018
Neues Thema	Das Thema Integration von AWS Cloud9 in AWS CodeCommit wurde hinzugefügt. Das Produkt- und Service-Integrationen Thema wurde mit Informationen über AWS Cloud9 aktualisiert.	1. Dezember 2017
Neuer -Abschnitt	Der Abschnitt Verwenden von Pull-Anforderungen inAWS CodeCommitRepositories wurde hinzugefügt. Der Abschnitt Authentifizierung und Zugriffskontrolle für AWS CodeCommit wurde mit Informationen zu Berechtig	20. November 2017

Änderung	Beschreibung	Datum geändert
	ungen für Pull-Anforderungen und die Kommentarfunktion aktualisiert. Außerdem enthält er aktualisierte verwaltete Richtlinienanweisungen.	
Aktualisierte Themen	Das Produkt- und Service-Integrationen Thema wurde aktualisiert und enthält nun Links für Kunden, die ihre bestehenden Pipelines aktualisieren möchten, um Amazon CloudWatch Events zum Starten von Pipelines als Reaktion auf Änderungen in einem CodeCommit Repository zu verwenden.	11. Oktober 2017
Neue Themen	Der Abschnitt Authentifizierung und Zugriffskontrolle für AWS CodeCommit wurde hinzugefügt. Er ersetzt das Referenz-Thema zu den Zugriffsberechtigungen.	11. September 2017
Aktualisierte Themen	Der Abschnitt Verwalten von Auslösern für ein Repository wurde aktualisiert, um die Änderung der Auslöser-Konfiguration zu berücksichtigen. Themen und Bilder wurden im gesamten Handbuch aktualisiert, um Änderungen in der Navigationsleiste zu berücksichtigen.	29. August 2017
Neues Thema	Das Thema Arbeiten mit Benutzereinstellungen wurde hinzugefügt. Das Thema Tag-Details anzeigen wurde aktualisiert. Die Produkt- und Service-Integrationen Themen wurden mit Informationen zur Integration mit Amazon CloudWatch Events aktualisiert.	3. August 2017
Neue Themen	Die Themen Integration von Eclipse in AWS CodeCommit und Integrieren Sie Visual Studio mitAWS CodeCommit wurden hinzugefügt.	29. Juni 2017
Aktualisiertes Thema	CodeCommit ist jetzt in zwei weiteren Regionen verfügbar: Asien-Pazifik (Mumbai) und Kanada (Zentral). Das Thema Regionen und Git-Verbindungsendpunkte wurde aktualisiert.	29. Juni 2017

Änderung	Beschreibung	Datum geändert
Aktualisiertes Thema	CodeCommit ist jetzt in vier weiteren Regionen verfügbar : Asien-Pazifik (Seoul), Südamerika (São Paulo), USA West (Nordkalifornien) und Europa (London). Das Thema Regionen und Git-Verbindungsendpunkte wurde aktualisi ert.	6. Juni 2017
Aktualisiertes Thema	CodeCommit ist jetzt in vier weiteren Regionen verfügbar: Asien-Pazifik (Tokio), Asien-Pazifik (Singapur), Asien-Paz ifik (Sydney) und Europa (Frankfurt). Das Regionen und Git-Verbindungsendpunkte Thema wurde aktualisiert und enthält nun Informationen zu Git-Verbindungsendpunkten und unterstützten Regionen für CodeCommit.	25. Mai 2017
Neues Thema	Das Thema Zweige vergleichen und zusammenführen wurde hinzugefügt. Der Inhalt des Abschnitts Arbeiten mit Zweigen wurde aktualisiert mit Informationen zur Verwendung der CodeCommit -Konsole für die Arbeit mit Branches in einem Repository.	18. Mai 2017
Neues Thema	Das Thema Vergleichen von Commits mit Informationen über das Vergleichen von Commits wurde hinzugefügt. Die Struktur des Benutzerhandbuchs wurde mit Informationen zum Verwenden von Repositorys, Commits und Branches aktualisiert.	28. März 2017
Aktualisiertes Thema	Das Thema Anzeigen von Commit-Details wurde mit Informationen zum Anzeigen des Unterschieds zwischen einem Commit und dem übergeordneten Commit in der Konsole und dem Anzeigen der Unterschiede zwischen Commits mit der AWS CLI und dem Befehl get-differences aktualisiert.	24. Januar 2017

Änderung	Beschreibung	Datum geändert
Neues Thema	Das Protokollierung von AWS CodeCommit-API-Aufrufen mit AWS CloudTrail Thema wurde mit Informationen zur Protokollierung von Verbindungen zur CodeCommit Verwendung AWS CloudFormation hinzugefügt.	11. Januar 2017
Neues Thema	Das Für HTTPS-Benutzer, die Git-Anmeldeinformationen verwenden Thema wurde mit Informationen zum Einrichte n von Verbindungen zur CodeCommit Verwendung von Git-Anmeldeinformationen über HTTPS hinzugefügt.	22. Dezember 2016
Aktualisiertes Thema	Das Thema Produkt- und Service-Integrationen wurde mit Informationen zur AWS CodeBuild-Integration aktualisiert.	5. Dezember 2016
Aktualisiertes Thema	CodeCommit ist jetzt in einer anderen Region, Europa (Irland), verfügbar. Das Regionen und Git-Verbindungsend punkte Thema wurde aktualisiert und enthält nun Informati onen zu Git-Verbindungsendpunkten und unterstützten Regionen für CodeCommit.	16. November 2016
Aktualisiertes Thema	CodeCommit ist jetzt in einer anderen Region, USA West (Oregon), verfügbar. Das Regionen und Git-Verbi ndungsendpunkte Thema wurde aktualisiert und enthält nun Informationen zu Git-Verbindungsendpunkten und unterstützten Regionen für CodeCommit.	14. November 2016

Änderung	Beschreibung	Datum geändert
Neues Thema	Das Erstellen Sie einen Trigger für eine Lambda-Funktion Thema wurde aktualisiert, um der Möglichkeit Rechnung zu tragen, CodeCommit Trigger im Rahmen der Erstellung der Lambda-Funktion zu erstellen. Dieser vereinfachte Prozess optimiert die Triggererstellung und konfiguriert den Trigger automatisch mit den Berechtigungen, die CodeCommit zum Aufrufen der Lambda-Funktion erforderlich sind. Das Erstellen Sie einen Trigger für eine bestehende Lambda-Funktion Thema wurde hinzugefügt und enthält nun Informationen zum Erstellen von Triggern für bestehende Lambda-Funktionen in der CodeCommit Konsole.	19. Oktober 2016
Neues Thema	CodeCommit ist jetzt in einer anderen Region, USA Ost (Ohio), verfügbar. Das Regionen und Git-Verbindungsendpunkte Thema wurde hinzugefügt, um Informationen zu Git-Verbindungsendpunkten und unterstützten Regionen für CodeCommit bereitzustellen.	17. Oktober 2016
Aktualisiertes Thema	Das Thema Produkt- und Service-Integrationen wurde mit Informationen zur AWS Elastic Beanstalk-Integration aktualisiert.	13. Oktober 2016
Aktualisiertes Thema	Das Thema Produkt- und Service-Integrationen wurde mit Informationen zur AWS CloudFormation-Integration aktualisiert.	6. Oktober 2016
Aktualisiertes Thema	Das Thema Für SSH-Verbindungen unter Windows wurde überarbeitet und enthält nun eine Anleitung zur Verwendung des Bash-Emulators mit SSH-Verbindungen in Windows (anstatt der PuTTY-Tool-Suite).	29. September 2016

Änderung	Beschreibung	Datum geändert
Aktualisiertes Thema	Die Erste Schritte mit CodeCommit Themen Anzeigen von Commit-Details und 2 wurden aktualisiert und enthalten nun Informationen über den Commit Visualizer in der CodeCommit Konsole. Das Thema Kontingente wurde mit Informationen über die Erhöhung der Anzahl zulässiger Referenzen für einen einzelnen Push aktualisiert.	14. September 2016
Aktualisiertes Thema	Die Erste Schritte mit CodeCommit Themen Anzeigen von Commit-Details und. Sie wurden aktualisiert und enthalten nun Informationen zur Anzeige des Commit-Verlaufs in der CodeCommit Konsole.	28. Juli 2016
Neue Themen	Die Themen Migrieren Sie ein Git-Repository zu AWS CodeCommit und Migrieren Sie lokalen oder unversionierten Inhalt zu AWS CodeCommit wurden hinzugefügt.	29. Juni 2016
Aktualisiertes Thema	An den Themen Fehlerbehebung und Für HTTPS-Verbindungen unter Windows mit demAWS CLICredential-Helper wurden geringfügige Aktualisierungen vorgenommen.	22. Juni 2016
Aktualisiertes Thema	Die Themen Produkt- und Service-Integrationen und Referenz zu Zugriffsberechtigungen wurden aktualisiert und enthalten nun Informationen zur Integration mit CodePipeline.	18. April 2016
Neue Themen	Der Abschnitt Verwalten von Auslösern für ein Repository wurde hinzugefügt. Die neuen Themen enthalten Beispiele (darunter Richtlinien- und Codebeispiele) zum Erstellen, Bearbeiten und Löschen von Auslösern.	7. März 2016
Neues Thema	Das Thema Produkt- und Service-Integrationen wurde hinzugefügt. Am Thema Fehlerbehebung wurden geringfügige Aktualisierungen vorgenommen.	7. März 2016

Änderung	Beschreibung	Datum geändert
Aktualisiertes Thema	Den Themen CodeCommit und Für SSH-Verbindungen unter Linux, macOS oder Unix wurde neben dem Fingerabdruck des MD5-Servers auch der Fingerabdruck des SHA256-Servers für Für SSH-Verbindungen unter Windows hinzugefügt.	9. Dezember 2015
Neues Thema	Das Thema Durchsuchen Sie Dateien in einem Repository wurde hinzugefügt. Dem Thema Fehlerbehebung wurden neue Problemlösungen hinzugefügt. Am gesamten Benutzerhandbuch wurden kleinere Verbesserungen vorgenommen.	5. Oktober 2015
Neues Thema	Das Thema Für SSH-Benutzer, die dieAWS CLI wurde hinzugefügt. Die Themen im Abschnitt Einrichtung wurden überarbeitet. Für die Benutzer wurde eine Anleitung mit Schritten für ihre jeweiligen Betriebssysteme und bevorzugten Protokolle hinzugefügt.	5. August 2015
Aktualisiertes Thema	In den Themen SSH und Linux, macOS oder Unix: Richte die öffentlichen und privaten Schlüssel für Git ein undCodeCommit und Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein wurden Erläuterungen und Beispiele für die Schritte der SSH-Schlüssel-ID hinzugefügt.	24. Juli 2015
Aktualisiertes Thema	Die unter aufgeführten Schritte Schritt 3: Richten Sie die öffentlichen und privaten Schlüssel für Git und CodeCommit ein wurden aktualisiert, um ein Problem mit IAM und dem Speichern der öffentlichen Schlüsseldatei zu beheben.	22. Juli 2015
Aktualisiertes Thema	Das Thema Fehlerbehebung wurde mit Navigationshilfen aktualisiert. Zudem wurden weitere Informationen zur Behebung von Problemen mit dem Anmeldeinformationsschlüssel hinzugefügt.	20. Juli 2015

Änderung	Beschreibung	Datum geändert
Aktualisiertes Thema	Weitere Informationen zu AWS Key Management Service Berechtigungen wurden AWS KMS und Verschlüsselung sowie den Themen "Zugriffsberechtigungen – Referenz" hinzugefügt.	17. Juli 2015
Aktualisiertes Thema	Dem Thema Fehlerbehebung wurde ein weiterer Abschnitt mit Informationen zur Fehlerbehebung in AWS Key Management Service hinzugefügt.	10. Juli 2015
Erstversion	Dies ist die erste Version des CodeCommit -Benutzer handbuchs.	9. Juli 2015

AWS-Glossar

Die neueste AWS-Terminologie finden Sie im AWS-Glossar in der AWS-Glossar-Referenz.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.