



Benutzerhandbuch

Amazon Managed Workflows für Apache Airflow



Amazon Managed Workflows für Apache Airflow: Benutzerhandbuch

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Handelsmarken und Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, durch die Kunden irregeführt werden könnten oder Amazon in schlechtem Licht dargestellt oder diskreditiert werden könnte. Alle anderen Handelsmarken, die nicht Eigentum von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise zu Amazon gehören oder nicht, mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Was ist Amazon MWAA?	1
Features	1
Architektur	2
Integration	4
Unterstützte Versionen	4
Als nächstes	4
Schnellstart	5
In diesem Tutorial	5
Voraussetzungen	6
Schritt eins: Speichern Sie die AWS CloudFormation Vorlage lokal	7
Schritt zwei: Erstellen Sie den Stack mit dem AWS CLI	17
Schritt drei: Laden Sie eine DAG auf Amazon S3 hoch und führen Sie sie in der Apache Airflow-Benutzeroberfläche aus	17
Vierter Schritt: Logs in CloudWatch Logs anzeigen	18
Als nächstes	19
Erste Schritte	20
Voraussetzungen	20
Informationen zu diesem Handbuch	20
Bevor Sie beginnen	21
Verfügbare Regionen	22
Erstellen eines Buckets	22
Bevor Sie beginnen	23
Erstellen Sie den cket cket cket	23
Als nächstes	25
Erstellen Sie das VPC-Netzwerk	25
Voraussetzungen	26
Bevor Sie beginnen	26
Optionen zum Erstellen des Amazon VPC-Netzwerks	26
Als nächstes	41
Erstellen einer Umgebung	41
Bevor Sie beginnen	41
Apache Airflow-Versionen	42
Erstellen einer Umgebung	43
Als nächstes	25

Zugriffsverwaltung	49
Zugreifen auf eine Amazon MWAA-Umgebung	49
Funktionsweise	50
Vollständiger Konsolenzugriff	52
Vollständiger API-Zugriff	58
Konsolenzugriff mit Schreibschutz	63
Zugriff auf die Apache Airflow-Benutzeroberfläche	63
Apache Airflow CLI-Zugriff	64
Eine JSON-Richtlinie erstellen	65
Beispielanwendungsfall	65
Als nächstes	67
Servicegebundene Rolle	68
Servicebezogene Rollenberechtigungen für Amazon MWAA	68
Eine serviceverknüpfte Rolle für Amazon MWAA erstellen	71
Bearbeiten einer serviceverknüpften Rolle für Amazon MWAA	72
Löschen einer serviceverknüpften Rolle für Amazon MWAA	72
Unterstützte Regionen für Rollen im Zusammenhang mit Amazon MWAA, die mit dem Service verknüpft sind	72
Richtlinienaktualisierungen	73
Ausführungsrolle	73
Übersicht über die Ausführungsrollen	74
Create a new role (Neue Rolle erstellen)	76
Eine Ausführungsrollenrichtlinie anzeigen und aktualisieren	77
Gewähren Sie Zugriff auf den Amazon S3 S3-Bucket mit Sperrung des öffentlichen Zugriffs auf Kontoebene	78
Verwenden Sie Apache Airflow-Verbindungen	79
Beispielrichtlinien	79
Als nächstes	85
Serviceübergreifende Confused-Deputy-Prävention	86
Apache Airflow-Zugriffsmodi	87
Apache Airflow-Zugriffsmodi	88
Übersicht über die Zugriffsmodi	90
Einrichtung für private und öffentliche Zugriffsmodi	91
Zugreifen auf den VPC-Endpunkt für Ihren Apache Airflow Webserver (privater Netzwerkzugriff)	92
Zugreifen auf Apache Airflow	93

Voraussetzungen	93
Zugriff	93
AWS CLI	94
Öffnen Sie die Airflow-Benutzeroberfläche	94
Bei Apache Airflow anmelden	94
Erstellen Sie ein Webserver-Zugriffstoken	94
Voraussetzungen	95
Mit dem AWS CLI	96
Verwenden eines Bash-Skripts	96
Verwenden einer POST-API-Anfrage	97
Verwenden eines Python-Skripts	98
Als nächstes	98
Apache Airflow CLI-Token	98
Voraussetzungen	99
Verwendung der AWS CLI	100
Verwenden eines Curl-Skripts	100
Ein Bash-Skript verwenden	102
Verwenden Sie ein Python-Skript	104
Als nächstes	107
Verwenden der Apache Airflow REST API	107
Erstellen Sie ein Webserver-Sitzungstoken	108
Rufen Sie die Apache Airflow REST API auf	109
Befehlsreferenz für Apache Airflow CLI	111
Voraussetzungen	111
Was hat sich in Version 2 geändert	112
Unterstützte CLI-Befehle	112
Beispiel-Code	115
Verwalten von Verbindungen	119
Übersicht	119
Apache Airflow-Pakete	120
Anbieterpakete für Apache Airflow v2.8.1-Verbindungen	120
Anbieterpakete für Apache Airflow v2.7.2-Verbindungen	121
Anbieterpakete für Apache Airflow v2.6.3-Verbindungen	122
Anbieterpakete für Apache Airflow v2.5.1-Verbindungen	123
Anbieterpakete für Apache Airflow v2.4.3-Verbindungen	124
Anbieterpakete für Apache Airflow v2.2.2-Verbindungen	124

Anbieterpakete für Apache Airflow v2.0.2-Verbindungen	125
Angeben neuerer Anbieterpakete	126
Arten von Verbindungen	126
Beispiel einer URI-Zeichenfolge für eine Verbindung	127
Beispiel-Verbindungsvorlage	127
Beispiel für die Verwendung einer HTTP-Verbindungsvorlage für eine Jdbc-Verbindung	129
Secrets Manager konfigurieren	131
Schritt eins: Erteilen Sie Amazon MWAA die Erlaubnis, auf geheime Schlüssel von Secrets Manager zuzugreifen	132
Schritt zwei: Erstellen Sie das Secrets Manager Manager-Backend als Apache Airflow-Konfigurationsoption	133
Schritt drei: Generieren Sie eine Apache AWS Airflow-Verbindungs-URI-Zeichenfolge	134
Schritt 4: Fügen Sie die Variablen in Secrets Manager hinzu	137
Schritt fünf: Fügen Sie die Verbindung in Secrets Manager hinzu	139
Beispiel-Code	140
Ressourcen	140
Als nächstes	141
Verwalten von Umgebungen	142
Konfiguration der Umgebungsklasse	142
Funktionen der Umgebung	143
Apache Airflow Scheduler	144
Konfiguration von Worker Auto Scaling	145
So funktioniert Worker Scaling	145
Verwenden der Amazon MWAA-Konsole	146
Beispiel für einen Hochleistungs-Anwendungsfall	146
Problembehandlung bei Aufgaben, die im laufenden Zustand hängen geblieben sind	148
Als nächstes	148
Automatische Skalierung von Webservern konfigurieren	149
So funktioniert die Skalierung von Webservern	149
Verwenden der Amazon MWAA-Konsole	149
Verwenden von Konfigurationsoptionen	150
Voraussetzungen	151
Funktionsweise	151
Verwenden von Konfigurationsoptionen zum Laden von Plugins in Apache Airflow v2	152
Übersicht über die Konfigurationsoptionen	152
Konfigurationsreferenz	154

Beispiele und Beispielcode	160
Als nächstes	162
Aktualisierung der Version	162
Aktualisieren Sie Ihre Workflow-Ressourcen	163
Geben Sie die neue Version an	164
Verwenden eines Startskripts	165
Konfigurieren Sie ein Startskript	166
Installieren Sie Linux-Laufzeiten	170
Festlegen von Umgebungsvariablen	171
Mit DAGs arbeiten	175
Überblick über den Amazon S3 S3-Bucket	175
Hinzufügen oder Aktualisieren von DAGs	176
Voraussetzungen	176
Funktionsweise	177
Was hat sich in Version 2 geändert	178
Testen von DAGs mit dem Amazon MWAA CLI Utility	178
Hochladen von DAG-Code in Amazon S3	178
Den Pfad zu einem DAG-Ordner angeben	180
Anzeigen von Änderungen in Ihrer Apache Airflow-Benutzeroberfläche	180
Als nächstes	180
Installation benutzerdefinierter Plugins	181
Voraussetzungen	181
Funktionsweise	182
Was hat sich in Version 2 geändert	182
Übersicht über benutzerdefinierte Plugins	183
Beispiele für benutzerdefinierte Plugins	184
Eine Datei vom Typ plugins.zip erstellen	193
Auf Amazon plugins.zip S3 hochladen	194
Installation benutzerdefinierter Plugins in Ihrer Umgebung	196
Beispielhafte Anwendungsfälle für plugins.zip	197
Als nächstes	197
Python-Abhängigkeiten installieren	197
Voraussetzungen	198
Funktionsweise	199
Überblick über die Python-Abhängigkeiten	199
Eine Datei requirements.txt erstellen	200

Auf Amazon requirements.txt S3 hochladen	203
Installation von Python-Abhängigkeiten in Ihrer Umgebung	204
Logs für Ihr anzeigen requirements.txt	206
Als nächstes	206
Löschen von Dateien auf Amazon S3	207
Voraussetzungen	207
Überblick über die Versionierung	208
Funktionsweise	208
Löschen einer DAG auf Amazon S3	208
Die „aktuelle“ Datei plugins.zip oder requirements.txt wird entfernt	209
Löschen Sie die „nicht aktuelle“ Datei plugins.zip oder requirements.txt	209
Löschen von Dateien mit Lebenszyklen	210
Beispiel für eine Lebenszyklusrichtlinie	210
Als nächstes	211
Netzwerk	212
Informationen zu Netzwerken	212
Bedingungen	213
Was wird unterstützt	213
Übersicht über die VPC-Infrastruktur	213
Beispielanwendungsfälle für einen Amazon-VPC- und Apache-Airflow-Zugriffsmodus	217
Sicherheit in Ihrer VPC	219
Bedingungen	220
Überblick über die Sicherheit	220
Netzwerk-Zugriffskontrolllisten (ACLs)	221
VPC-Sicherheitsgruppen	222
VPC-Endpunktrichtlinien (nur privates Routing)	224
Verwaltung des Zugriffs auf VPC-Endpunkte	225
Preisgestaltung	226
Übersicht über VPC-Endpunkte	226
Erlaubnis zur Nutzung anderer Dienste AWS	227
VPC-Endpoints anzeigen	227
Zugreifen auf den VPC-Endpunkt für Ihren Apache Airflow Webserver (privater Netzwerkzugriff)	229
VPC-Serviceendpunkte in privaten Amazon-VPCs	231
Preisgestaltung	231
Privates Netzwerk und privates Routing	232

(Erforderlich) VPC-Endpunkte	233
Anhängen der erforderlichen VPC-Endpoints	233
(Optional) Aktivieren Sie private IP-Adressen für Ihren Amazon S3 S3-VPC-Schnittstellenendpunkt	238
Verwalten Ihrer eigenen Amazon-VPC-Endpunkte	239
Erstellen einer Umgebung in einer freigegebenen Amazon VPC	239
Tutorials	250
Tutorial: AWS Client VPN	250
Privates Netzwerk	251
Anwendungsfälle	252
Bevor Sie beginnen	252
Zielsetzungen	252
(Optional) Erster Schritt: Identifizieren Sie Ihre VPC, CIDR-Regeln und VPC-Sicherheit (en)	253
Schritt 2: Erstellen der Server- und Client-Zertifikate	254
Schritt drei: Speichern Sie dieAWS CloudFormation Vorlage lokal	256
Schritt vier: Erstellen Sie denAWS CloudFormation Client-VPN-Stack	257
Schritt fünf: Subnetze mit Ihrem Client VPN verknüpfen	258
Schritt 6: Fügen Sie Ihrem Client VPN eine Autorisierungsregel für den Eingang hinzu	258
Schritt 7: Herunterladen der Server VPN-Endpunkt-Konfigurationsdatei	259
Schritt acht: Stellen Sie eine Verbindung zum herAWS Client VPN	261
Als nächstes	261
Anleitung: Linux Bastion Host	262
Privates Netzwerk	262
Anwendungsfälle	264
Bevor Sie beginnen	264
Ziele	264
Erster Schritt: Erstellen Sie die Bastion-Instanz	265
Schritt zwei: Erstellen Sie den SSH-Tunnel	266
Schritt drei: Konfigurieren Sie die Bastion-Sicherheitsgruppe als Regel für eingehenden Datenverkehr	268
Vierter Schritt: Kopieren Sie die Apache Airflow-URL	268
Fünfter Schritt: Proxyeinstellungen konfigurieren	268
Schritt 6: Öffnen Sie die Apache Airflow-Benutzeroberfläche	271
Als nächstes	271
Tutorial: Benutzer auf eine Teilmenge von DAGs beschränken	272

Voraussetzungen	272
Schritt eins: Gewähren Sie dem Amazon MWAA-Webserver Zugriff auf Ihren IAM-Principal mit der Standardrolle <code>Public Apache Airflow</code>	273
Schritt zwei: Erstellen Sie eine neue benutzerdefinierte Apache Airflow-Rolle	274
Schritt drei: Weisen Sie Ihrem Amazon MWAA-Benutzer die von Ihnen erstellte Rolle zu	275
Nächste Schritte	276
Zugehörige Ressourcen	276
Tutorial: Automatisieren der Verwaltung Ihrer eigenen Umgebungsendpunkte	276
Voraussetzungen	277
Erstellen der Amazon VPC	277
So erstellen Sie die Lambda-Funktion:	278
Erstellen der EventBridge Regel	279
Erstellen der -Umgebung	280
Codebeispiele	282
Variablen DAG importieren	283
Version	283
Voraussetzungen	283
Berechtigungen	283
Abhängigkeiten	283
Codebeispiel	284
Als nächstes	285
Verwenden des <code>SSHOperator</code>	285
Version	286
Voraussetzungen	286
Berechtigungen	287
Voraussetzungen	287
Kopieren Sie Ihren geheimen Schlüssel zu Amazon S3	287
Erstellen Sie eine neue Apache Airflow-Verbindung	287
Codebeispiel	288
Apache Airflow Snowflake-Verbindung in Secrets Manager	290
Version	290
Voraussetzungen	290
Berechtigungen	291
Voraussetzungen	291
Codebeispiel	291
Als nächstes	292

Verwenden einer DAG zum Schreiben benutzerdefinierter Metriken	292
Version	293
Voraussetzungen	293
Berechtigungen	293
Abhängigkeiten	293
Codebeispiel	294
Aurora-PostgreSQL-Datenbankbereinigung	297
Version	297
Voraussetzungen	297
Abhängigkeiten	297
Codebeispiel	297
Exportieren von Umgebungsmetadaten nach Amazon S3	300
Version	300
Voraussetzungen	300
Berechtigungen	301
Voraussetzungen	301
Codebeispiel	301
Verwenden einer Apache Airflow-Variablen in Secrets Manager	304
Version	304
Voraussetzungen	304
Berechtigungen	305
Voraussetzungen	305
Codebeispiel	305
Als nächstes	306
Verwenden einer Apache Airflow-Verbindung in Secrets Manager	306
Version	307
Voraussetzungen	307
Berechtigungen	307
Voraussetzungen	305
Codebeispiel	308
Als nächstes	311
Benutzerdefiniertes Plugin mit Oracle	311
Version	312
Voraussetzungen	312
Berechtigungen	312
Voraussetzungen	312

Codebeispiel	313
Erstellen Sie das benutzerdefinierte Plugin	314
Airflow-Konfigurationsoptionen	317
Als nächstes	317
Benutzerdefiniertes Plugin mit Umgebungsvariablen	317
Version	318
Voraussetzungen	318
Berechtigungen	318
Voraussetzungen	318
Benutzerdefiniertes Plugin	318
Plugins.zip	319
Airflow-Konfigurationsoptionen für den Luftstrom	319
Als nächstes	320
Zeitzone einer DAG ändern	320
Version	320
Voraussetzungen	320
Berechtigungen	321
Erstellen Sie ein Plugin, um die Zeitzone in Airflow-Protokollen zu ändern	321
plugins.zipZiel erstellen	322
Codebeispiel	322
Als nächstes	323
Erfrischend undAWS CodeArtifactToken zur Laufzeit	324
Version	324
Voraussetzungen	324
Berechtigungen	324
Codebeispiel	325
Als nächstes	326
Benutzerdefiniertes Plugin mit Apache Hive und Hadoop	327
Version	327
Voraussetzungen	328
Berechtigungen	328
Voraussetzungen	305
Abhängigkeiten herunterladen	328
Benutzerdefiniertes Plugin	329
Plugins.zip	330
Codebeispiel	330

Airflow-Konfigurationsoptionen	331
Als nächstes	331
Benutzerdefiniertes Plugin zum PatchenPythonVirtualenvOperator	331
Version	332
Voraussetzungen	332
Berechtigungen	332
Voraussetzungen	332
Benutzerdefinierter Plugin-Beispielcode	333
Plugins.zip	335
Codebeispiel	335
Airflow-Konfigurationsoptionen	337
Als nächstes	337
Aufrufen von DAGs mit Lambda	338
Version	338
Voraussetzungen	338
Berechtigungen	339
Abhängigkeiten	339
Codebeispiel	339
DAGs in verschiedenen Umgebungen aufrufen	341
Version	341
Voraussetzungen	341
Berechtigungen	342
Abhängigkeiten	342
Codebeispiel	342
Amazon RDS-Server	344
Version	345
Voraussetzungen	345
Abhängigkeiten	297
Apache Airflow v2-Verbindung	346
Codebeispiel	346
Als nächstes	349
Amazon EMR Studios	349
Version	349
Codebeispiel	349
Amazon EKS (exkl.)	352
Version	353

Voraussetzungen	353
Erstellen Sie einen öffentlichen Schlüssel für Amazon EC2	353
Erstellen Sie den Cluster	354
Erstelle einemwaaNamensraum	354
Erstellen Sie eine Rolle fürmwaaNamensraum	355
Eine IAM-Rolle für den Amazon EKS-Cluster erstellen und anhängen	356
Erstellen Sie die Datei requirements.txt	359
Erstellen Sie eine Identitätszuordnung für Amazon EKS	360
Erstellen der kubeconfig	360
Erstellen Sie eine DAG	361
Fügen Sie die DAG hinzu undkubernetes_config.yamlzum Amazon S3-Bucket	363
Das Beispiel aktivieren und auslösen	363
Verwendung der ECSOperator	364
Version	364
Voraussetzungen	364
Berechtigungen	364
Erstellen Sie einen Amazon ECS-Cluster	366
Codebeispiel	371
Verwenden von dbt mit Amazon MWAA	374
Version	374
Voraussetzungen	375
Abhängigkeiten	375
Laden Sie ein dbt-Projekt auf Amazon S3 hoch	376
Verwenden Sie eine DAG, um die Installation der DBT-Abhängigkeit zu überprüfen	377
Verwenden Sie eine DAG, um ein DBT-Projekt auszuführen	378
AWS Blogs und Tutorials	379
Bewährte Methoden	380
Leistungsoptimierung für Apache Airflow	380
Hinzufügen einer Apache Airflow-Konfigurationsoption	380
Apache Airflow Scheduler	381
DAG-Ordner	387
DAG-Dateien	389
Aufgaben	394
Verwaltung von Python-Abhängigkeiten	399
Testen von DAGs mit dem Amazon MWAA CLI Utility	400
Installation von Python-Abhängigkeiten mit dem PyPi .org-Anforderungsdateiformat	400

Aktivieren von Protokollen auf der Amazon MWAA-Konsole	407
Protokolle in der CloudWatch Logs-Konsole anzeigen	408
Fehler in der Apache Airflow-Benutzeroberfläche anzeigen	409
Beispielszenarien requirements.txt	410
Überwachung und Metriken	411
Übersicht	411
CloudWatch Überblick über Amazon	412
AWS CloudTrail Überblick	412
Anzeigen von Audit-Protokollen	412
Einen Trail erstellen in CloudTrail	413
Ereignisse mit dem CloudTrail Ereignisverlauf anzeigen	413
Beispielpfad für CreateEnvironment	413
Als nächstes	415
Airflow-Protokolle anzeigen	415
Preisgestaltung	416
Bevor Sie beginnen	416
Typen von Protokollen	416
Apache Airflow-Protokolle aktivieren	417
Apache Airflow-Protokolle anzeigen	418
Beispiel für Scheduler-Protokolle	418
Als nächstes	419
Überwachung von Dashboards und Alarmen	419
Metriken	420
Übersicht über die Alarmstatus	420
Beispiel für benutzerdefinierte Dashboards und Alarme	420
Löschen von Metriken und Dashboards	426
Als nächstes	426
Apache Airflow v2-Umgebungsmetriken	426
Bedingungen	427
Dimensionen	428
Zugreifen auf Metriken in der Konsole CloudWatch	429
Apache Airflow-Metriken sind verfügbar in CloudWatch	429
Auswahl der Metriken, die gemeldet werden	444
Als nächstes	445
Container-, Warteschlangen- und Datenbank-Metriken	445
Bedingungen	446

Dimensionen	446
Zugreifen auf -Metriken	447
Liste der Metriken	448
Sicherheit	452
Datenschutz	453
Verschlüsselung	454
Verwendung von vom Kunden verwalteten Schlüsseln	456
AWS Identity and Access Management	460
Zielgruppe	461
Authentifizierung mit Identitäten	461
Verwalten des Zugriffs mit Richtlinien	465
Benutzern die Berechtigung zur Anzeige eigener Berechtigungen erteilen	468
Fehlerbehebung bei Amazon Managed Workflows für Identität und Zugriff auf Apache Airflow	469
So funktioniert Amazon MWAA mit IAM	470
Compliance-Validierung	476
Ausfallsicherheit	477
Sicherheit der Infrastruktur	478
Konfigurations- und Schwachstellenanalyse	478
Bewährte Methoden	479
Bewährte Sicherheitsmethoden in Apache Airflow	479
Versionen	482
Informationen zu Amazon MWAA-Versionen	482
Aktuelle Version	482
Apache Airflow-Versionen	482
Apache Airflow-Komponenten	484
Schedulers	484
Worker	485
Aktualisieren der Apache Airflow-Version	485
Veraltete Versionen von Apache Airflow	485
Unterstützung der Apache Airflow-Version und häufig gestellte Fragen	486
Häufig gestellte Fragen	486
Endpunkte und Kontingente	488
Service-Endpunkte	488
Servicekontingente	488
Steigende Quoten	489

Häufig gestellte Fragen	490
Unterstützte Versionen	491
Apache Airflow-Unterstützung	491
Apache Airflow-Versionen	491
Python-Version	491
Pip-Version	492
Anwendungsfälle	493
Wann sollte ich im AWS Step Functions Vergleich zu verwenden? Amazon MWAA?	493
Umgebungsspezifikationen	493
Wie viel Aufgabenspeicher ist für jede Umgebung verfügbar?	493
Standard-Betriebssystem	493
Benutzerdefinierte Images	494
Compliance mit HIPAA	494
Unterstützt Amazon MWAA Spot-Instances?	494
Benutzerdefinierte Domain	494
SSH-Zugriff	495
Selbstreferenzierende Regel	495
Benutzerdefinierte Metriken	495
Speichern von Daten	495
Worker-Kontingent	496
Gemeinsam genutzte Amazon VPCs	496
Metriken	496
Worker-Metriken	496
Benutzerdefinierte Metriken	496
DAGs , Operatoren, Verbindungen und andere Fragen	497
PythonVirtualenvOperator	497
Wie lange braucht Amazon MWAA, um eine neue DAG-Datei zu erkennen?	497
Warum wird meine DAG-Datei nicht von Apache Airflow aufgenommen?	497
Entfernen Sie plugins.zip oder requirements.txt	497
Entfernen Sie plugins.zip oder requirements.txt	498
Kann ich AWS Database Migration Service (DMS) Operatoren verwenden?	498
Fehlerbehebung	499
Apache Airflow v2	502
Verbindungen	503
Webserver	505
Aufgaben	506

CLI	509
Operatoren	511
Apache Airflow v1	512
requirements.txt wird aktualisiert	513
Defekte DAG	513
Operatoren	516
Verbindungen	516
Webserver	519
Aufgaben	520
CLI	523
Amazon MWAA Erstellen/Aktualisieren	524
Aktualisieren von requirements.txt	525
Plug-ins	526
Erstellen Buckets	526
Erstellen der -Umgebung	527
Umgebung aktualisieren	530
Zugangsumgebung	531
CloudWatch Protokolle und CloudTrail	531
Logs (Protokolle)	532
Dokumentverlauf	538
.....	dcxii

Was ist Amazon Managed Workflows for Apache Airflow?

Amazon Managed Workflows for Apache Airflow ist ein verwalteter Orchestrierungsservice für [Apache Airflow](#), mit dem Sie Datenpipelines in der Cloud in großem Umfang einrichten und betreiben können. Apache Airflow ist ein Open-Source-Tool, das verwendet wird, um programmgesteuert Sequenzen von Prozessen und Aufgaben zu erstellen, zu planen und zu überwachen, die als Workflows bezeichnet werden. Mit Amazon MWAA können Sie Apache Airflow und Python verwenden, um Workflows zu erstellen, ohne die zugrunde liegende Infrastruktur für Skalierbarkeit, Verfügbarkeit und Sicherheit verwalten zu müssen. Amazon MWAA skaliert automatisch seine Workflow-Ausführungskapazität entsprechend Ihren Anforderungen. Amazon MWAA lässt sich in - AWS Sicherheitsservices integrieren, um Ihnen einen schnellen und sicheren Zugriff auf Ihre Daten zu ermöglichen.

Inhalt

- [Features](#)
- [Architektur](#)
- [Integration](#)
- [Unterstützte Versionen](#)
- [Als nächstes](#)

Features

- Automatische Airflow-Einrichtung – Richten Sie Apache Airflow schnell ein, indem Sie beim Erstellen einer Amazon MWAA-Umgebung eine [Apache Airflow-Version](#) auswählen. Amazon MWAA richtet Apache Airflow für Sie über dieselbe Apache-Airflow-Benutzeroberfläche und denselben Open-Source-Code ein, den Sie im Internet herunterladen können.
- Auto Scaling – Skalieren Sie Apache Airflow Workers automatisch, indem Sie die minimale und maximale Anzahl von Workern festlegen, die in Ihrer Umgebung ausgeführt werden. Amazon MWAA überwacht die Worker in Ihrer Umgebung und verwendet ihre [Auto-Scaling-Komponente](#), um Worker hinzuzufügen, um die Nachfrage zu erfüllen, bis und bis sie die maximale Anzahl von Workern erreicht, die Sie definiert haben.
- Integrierte Authentifizierung – Aktivieren Sie die rollenbasierte Authentifizierung und Autorisierung für Ihren Apache-Airflow-Webserver, indem Sie die [Zugriffskontrollrichtlinien](#) in AWS Identity and

Access Management (IAM) definieren. Die Apache Airflow Workers übernehmen diese Richtlinien für den sicheren Zugriff auf - AWS Services.

- Integrierte Sicherheit – Die Apache Airflow Workers und Scheduler werden in [Amazon MWAA Amazon VPC](#) ausgeführt. Daten werden auch automatisch mit verschlüsselt AWS Key Management Service, sodass Ihre Umgebung standardmäßig sicher ist.
- Öffentliche oder private Zugriffsmodi – Greifen Sie über einen privaten oder öffentlichen [Zugriffsmodus](#) auf Ihren Apache-Airflow-Webserver zu. Der Modus Öffentlicher Netzwerkzugriff verwendet einen VPC-Endpunkt für Ihren Apache-Airflow-Webserver, auf den über das Internet zugegriffen werden kann. Der private Netzwerkzugriffsmodus verwendet einen VPC-Endpunkt für Ihren Apache-Airflow-Webserver, auf den in Ihrer VPC zugegriffen werden kann. In beiden Fällen wird der Zugriff für Ihre Apache-Airflow-Benutzer durch die Zugriffskontrollrichtlinie gesteuert, die Sie in AWS Identity and Access Management (IAM) und AWS SSO definieren.
- Optimierte Upgrades und Patches – Amazon MWAA bietet regelmäßig neue Versionen von Apache Airflow. Das Amazon MWAA-Team aktualisiert und patcht die Images für diese Versionen.
- Workflow-Überwachung – Zeigen Sie Apache Airflow-Protokolle und [Apache Airflow-Metriken](#) in Amazon an CloudWatch , um Apache Airflow-Aufgabenverzögerungen oder Workflow-Fehler zu identifizieren, ohne dass zusätzliche Tools von Drittanbietern erforderlich sind. Amazon MWAA sendet automatisch Umgebungsmetriken – und, falls aktiviert – Apache Airflow-Protokolle an CloudWatch.
- -AWS Integration – Amazon MWAA unterstützt Open-Source-Integrationen mit Amazon Athena , AWS BatchAmazon CloudWatch, Amazon DynamoDB , AWS DataSyncAmazon EMR AWS Fargate, Amazon EKS, Amazon Data Firehose, AWS Glue AWS LambdaAmazon Redshift, Amazon SQS , Amazon SNS SageMaker , Amazon und Amazon S3 sowie Hunderte von integrierten und von der Community erstellten Operatoren und Sensoren.
- Worker-Flotten – Amazon MWAA bietet Unterstützung für die Verwendung von Containern, um die Worker-Flotte bei Bedarf zu skalieren und Scheduler-Ausfälle mit [Amazon ECS in AWS Fargate](#) zu reduzieren. Operatoren, die Aufgaben auf Amazon-ECS-Containern aufrufen, und Kubernetes-Operatoren, die Pods auf einem Kubernetes-Cluster erstellen und ausführen, werden unterstützt.

Architektur

Alle Komponenten im äußeren Feld (im folgenden Bild) werden als einzelne Amazon MWAA-Umgebung in Ihrem Konto angezeigt. Der Apache Airflow Scheduler und die Worker sind AWS Fargate (Fargate) Container, die eine Verbindung zu den privaten Subnetzen in der Amazon VPC für Ihre Umgebung herstellen. Jede Umgebung verfügt über eine eigene Apache Airflow-Metadatenbasis

AWS , die von verwaltet wird und über einen privat gesicherten VPC-Endpunkt für die Scheduler- und Workers-Fargate-Container zugänglich ist.

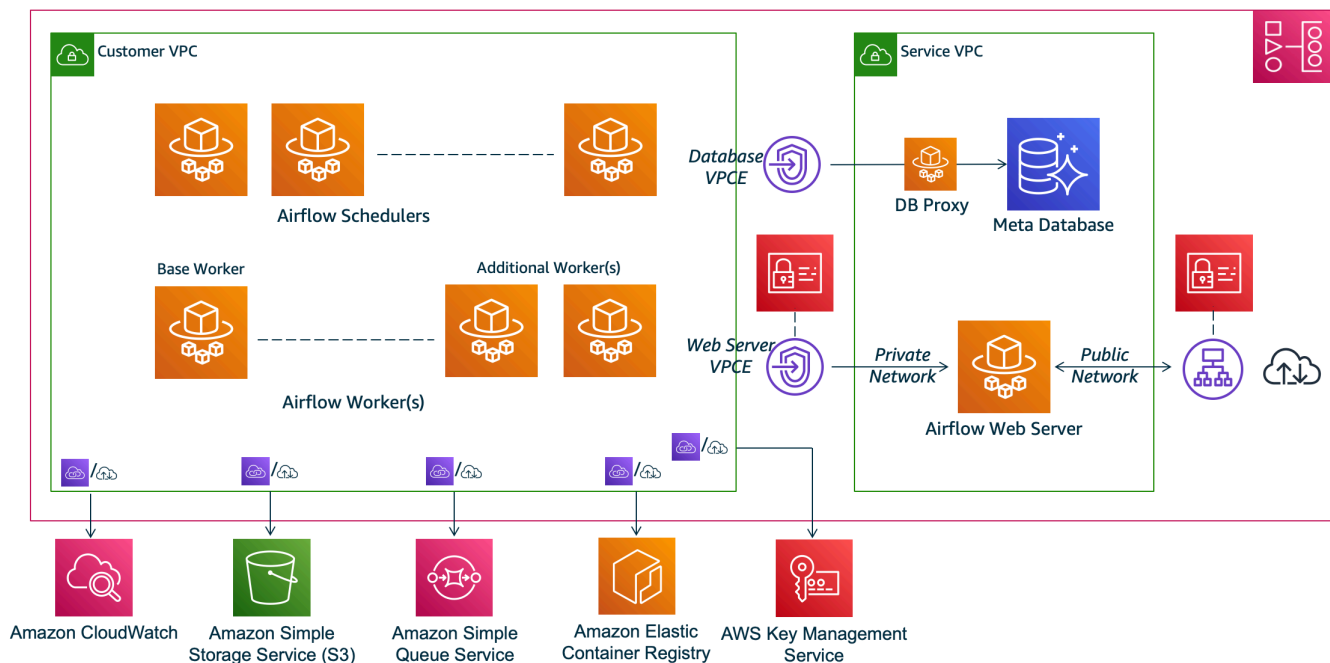
Amazon CloudWatch, Amazon S3, Amazon SQS , Amazon ECR und AWS KMS sind von Amazon MWA getrennt und müssen von Apache Airflow Scheduler(s) und Workern in den Fargate-Containern aus zugänglich sein.

Auf den Apache-Airflow-Webserver kann entweder über das Internet zugegriffen werden, indem Sie den Apache-Airflow-Zugriffsmodus für öffentliche Netzwerke auswählen, oder innerhalb Ihrer VPC, indem Sie den Apache-Airflow-Zugriffsmodus für private Netzwerke auswählen. In beiden Fällen wird der Zugriff für Ihre Apache-Airflow-Benutzer durch die Zugriffskontrollrichtlinie gesteuert, die Sie in AWS Identity and Access Management (IAM) definieren.

Note

Mehrere Apache Airflow Scheduler sind nur mit Apache Airflow v2 und höher verfügbar. Weitere Informationen zum Lebenszyklus von Apache Airflow-Aufgaben finden Sie unter [Konzepte](#) im Apache Airflow-Referenzhandbuch.

Amazon MWA Architecture



Integration

Die aktive und wachsende Open-Source-Community von Apache Airflow bietet Operatoren (Plugins, die Verbindungen zu -Services vereinfachen), damit Apache Airflow in - AWS Services integriert werden kann. Dazu gehören Services wie Amazon S3, Amazon Redshift, Amazon EMR AWS Batch und Amazon SageMaker sowie Services auf anderen Cloud-Plattformen.

Die Verwendung von Apache Airflow mit Amazon MWAA unterstützt vollständig die Integration mit - AWS Services und beliebten Tools von Drittanbietern wie Apache Hadoop, Presto, Hive und Spark zur Durchführung von Datenverarbeitungsaufgaben. Amazon MWAA verpflichtet sich, die Kompatibilität mit der Amazon-MWAA-API aufrechtzuerhalten, und Amazon MWAA beabsichtigt, zuverlässige Integrationen für - AWS Services bereitzustellen, sie der Community zur Verfügung zu stellen und an der Entwicklung von Community-Features beteiligt zu sein.

Einen Beispiel-Code finden Sie unter [Codebeispiele für Amazon Managed Workflows für Apache Airflow](#).

Unterstützte Versionen

Amazon MWAA unterstützt mehrere Versionen von Apache Airflow. Weitere Informationen zu den von uns unterstützten Apache-Airflow-Versionen und den in jeder Version enthaltenen Apache-Airflow-Komponenten finden Sie unter [Apache Airflow-Versionen auf Amazon Managed Workflows für Apache Airflow](#).

Als nächstes

- Beginnen Sie mit einer einzigen AWS CloudFormation Vorlage, die einen Amazon S3-Bucket für Ihre Airflow-DAGs und unterstützenden Dateien, eine Amazon VPC mit öffentlichem Routing und eine Amazon-MWAA-Umgebung in erstellt [Schnellstartanleitung für Amazon Managed Workflows für Apache Airflow](#).
- Beginnen Sie inkrementell, indem Sie einen Amazon S3-Bucket für Ihre Airflow-DAGs erstellen und Dateien unterstützen, eine von drei Amazon-VPC-Netzwerkoptionen auswählen und eine Amazon-MWAA-Umgebung in erstellen [Beginnen Sie mit Amazon Managed Workflows for Apache Airflow](#).

Schnellstartanleitung für Amazon Managed Workflows für Apache Airflow

Dieses Schnellstart-Tutorial verwendet eine AWS CloudFormation Vorlage, die gleichzeitig die Amazon VPC-Infrastruktur, einen Amazon S3 S3-Bucket mit einem dags Ordner und eine Amazon Managed Workflows for Apache Airflow-Umgebung erstellt.

Themen

- [In diesem Tutorial](#)
- [Voraussetzungen](#)
- [Schritt eins: Speichern Sie die AWS CloudFormation Vorlage lokal](#)
- [Schritt zwei: Erstellen Sie den Stack mit dem AWS CLI](#)
- [Schritt drei: Laden Sie eine DAG auf Amazon S3 hoch und führen Sie sie in der Apache Airflow-Benutzeroberfläche aus](#)
- [Vierter Schritt: Logs in CloudWatch Logs anzeigen](#)
- [Als nächstes](#)

In diesem Tutorial

Dieses Tutorial führt Sie durch drei AWS Command Line Interface (AWS CLI) Befehle, um eine DAG auf Amazon S3 hochzuladen, die DAG in Apache Airflow auszuführen und Logs einzusehen. CloudWatch Abschließend werden Sie durch die Schritte zur Erstellung einer IAM-Richtlinie für ein Apache Airflow-Entwicklungsteam geführt.

Note

Die AWS CloudFormation Vorlage auf dieser Seite erstellt eine Amazon Managed Workflows for Apache Airflow-Umgebung für die neueste Version von Apache Airflow, verfügbar in. AWS CloudFormation Die neueste verfügbare Version ist Apache Airflow v2.8.1.

Die AWS CloudFormation Vorlage auf dieser Seite erstellt Folgendes:

- VPC-Infrastruktur. Die Vorlage verwendet [Öffentliches Routing über das Internet](#). Sie verwendet den [Öffentlicher Netzwerkzugriffsmodus](#) für den Apache Airflow Webserver in `WebserverAccessMode: PUBLIC_ONLY`.
- Amazon S3 S3-Bucket. Die Vorlage erstellt einen Amazon S3 S3-Bucket mit einem dags Ordner. Es ist so konfiguriert, dass der gesamte öffentliche Zugriff blockiert wird, wobei die Bucket-Versionierung aktiviert ist, wie unter definiert. [Erstellen eines Amazon S3 cket cket erstellen](#)
- Amazon MWAA-Umgebung. Die Vorlage erstellt eine Amazon MWAA-Umgebung, die dem dags Ordner im Amazon S3 S3-Bucket zugeordnet ist, eine Ausführungsrolle mit Berechtigungen für von Amazon MWAA verwendete AWS Dienste und die Standardeinstellung für die Verschlüsselung mit einem [AWS eigenen Schlüssel](#), wie unter definiert. [Erstellen Sie eine Amazon MWAA-Umgebung](#)
- CloudWatch Protokolle. Die Vorlage ermöglicht, dass Apache Airflow sich CloudWatch auf der Ebene „INFO“ und höher für die Airflow-Scheduler-Protokollgruppe, die Airflow-Webserver-Protokollgruppe, die Airflow-Worker-Protokollgruppe, die Airflow-DAG-Verarbeitungsprotokollgruppe und die Airflow-Aufgabenprotokollgruppe anmeldet, wie unter definiert. [Airflow-Protokolle in Amazon anzeigen CloudWatch](#)

In diesem Tutorial werden Sie die folgenden Aufgaben erledigen:

- Laden Sie eine DAG hoch und führen Sie sie aus. Laden Sie das Tutorial-DAG von Apache Airflow für die neueste von Amazon MWAA unterstützte Apache Airflow-Version auf Amazon S3 hoch und führen Sie es dann in der Apache Airflow-Benutzeroberfläche aus, wie unter definiert. [Hinzufügen oder Aktualisieren von DAGs](#)
- Protokolle anzeigen. Zeigen Sie die Airflow-Webserver-Protokollgruppe unter CloudWatch Protokolle an, wie unter definiert. [Airflow-Protokolle in Amazon anzeigen CloudWatch](#)
- Erstellen Sie eine Zugriffskontrollrichtlinie. Erstellen Sie in IAM eine Zugriffskontrollrichtlinie für Ihr Apache Airflow-Entwicklungsteam, wie unter definiert. [Zugreifen auf eine Amazon MWAA-Umgebung](#)

Voraussetzungen

Das AWS Command Line Interface (AWS CLI) ist ein Open-Source-Tool, mit dem Sie mithilfe von Befehlen in Ihrer Befehlszeilen-Shell mit AWS Diensten interagieren können. Um die Schritte auf dieser Seite abzuschließen, benötigen Sie Folgendes:

- [AWS CLI — Installieren Sie Version 2.](#)

- [AWS CLI — Schnelle Konfiguration mit aws configure](#).

Schritt eins: Speichern Sie die AWS CloudFormation Vorlage lokal

- Kopieren Sie den Inhalt der folgenden Vorlage und speichern Sie ihn lokal unter `mwa-public-network.yml`. Sie können [die Vorlage auch herunterladen](#).

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

```
Parameters:
```

```
  EnvironmentName:
```

```
    Description: An environment name that is prefixed to resource names
```

```
    Type: String
```

```
    Default: MWAAEnvironment
```

```
  VpcCIDR:
```

```
    Description: The IP range (CIDR notation) for this VPC
```

```
    Type: String
```

```
    Default: 10.192.0.0/16
```

```
  PublicSubnet1CIDR:
```

```
    Description: The IP range (CIDR notation) for the public subnet in the first  
Availability Zone
```

```
    Type: String
```

```
    Default: 10.192.10.0/24
```

```
  PublicSubnet2CIDR:
```

```
    Description: The IP range (CIDR notation) for the public subnet in the second  
Availability Zone
```

```
    Type: String
```

```
    Default: 10.192.11.0/24
```

```
  PrivateSubnet1CIDR:
```

```
    Description: The IP range (CIDR notation) for the private subnet in the first  
Availability Zone
```

```
    Type: String
```

```
    Default: 10.192.20.0/24
```

```
  PrivateSubnet2CIDR:
```

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

```
    Type: String
```

Default: 10.192.21.0/24

MaxWorkerNodes:

Description: The maximum number of workers that can run in the environment

Type: Number

Default: 2

DagProcessingLogs:

Description: Log level for DagProcessing

Type: String

Default: INFO

SchedulerLogsLevel:

Description: Log level for SchedulerLogs

Type: String

Default: INFO

TaskLogsLevel:

Description: Log level for TaskLogs

Type: String

Default: INFO

WorkerLogsLevel:

Description: Log level for WorkerLogs

Type: String

Default: INFO

WebserverLogsLevel:

Description: Log level for WebserverLogs

Type: String

Default: INFO

Resources:

#####

CREATE VPC

#####

VPC:

Type: AWS::EC2::VPC

Properties:

CidrBlock: !Ref VpcCIDR

EnableDnsSupport: true

EnableDnsHostnames: true

Tags:

- Key: Name

Value: MWAAEnvironment

InternetGateway:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

PrivateRouteTable2:
  Type: AWS::EC2::RouteTable
```

```
Properties:
  VpcId: !Ref VPC
  Tags:
    - Key: Name
      Value: !Sub ${EnvironmentName} Private Routes (AZ2)

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

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

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

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

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

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

```
GroupName: !Sub "airflow-security-group-${AWS::StackName}-MwaaEnvironment"

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

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

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

MwaaExecutionPolicy:
  DependsOn: EnvironmentBucket
  Type: AWS::IAM::ManagedPolicy
  Properties:
    Roles:
      - !Ref MwaaExecutionRole
    PolicyDocument:
      Version: 2012-10-17
      Statement:
        - Effect: Allow
          Action: airflow:PublishMetrics
          Resource:
```



```
    - !Sub "arn:aws:airflow:${AWS::Region}:${AWS::AccountId}:environment/
    ${EnvironmentName}"
  - Effect: Deny
    Action: s3:ListAllMyBuckets
    Resource:
      - !Sub "${EnvironmentBucket.Arn}"
      - !Sub "${EnvironmentBucket.Arn}/*"

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

  - Effect: Allow
    Action:
      - logs:DescribeLogGroups
    Resource: "*"

  - Effect: Allow
    Action:
      - logs:CreateLogStream
      - logs:CreateLogGroup
      - logs:PutLogEvents
      - logs:GetLogEvents
      - logs:GetLogRecord
      - logs:GetLogGroupFields
      - logs:GetQueryResults
      - logs:DescribeLogGroups
    Resource:
      - !Sub "arn:aws:logs:${AWS::Region}:${AWS::AccountId}:log-
group:airflow-${AWS::StackName}*"
  - Effect: Allow
    Action: cloudwatch:PutMetricData
    Resource: "*"

  - Effect: Allow
    Action:
      - sqs:ChangeMessageVisibility
      - sqs>DeleteMessage
      - sqs:GetQueueAttributes
      - sqs:GetQueueUrl
      - sqs:ReceiveMessage
```

```

    - sqs:SendMessage
  Resource:
    - !Sub "arn:aws:sqs:${AWS::Region}:*:airflow-celery-*"
- Effect: Allow
  Action:
    - kms:Decrypt
    - kms:DescribeKey
    - "kms:GenerateDataKey*"
    - kms:Encrypt
  NotResource: !Sub "arn:aws:kms:*:${AWS::AccountId}:key/*"
  Condition:
    StringLike:
      "kms:ViaService":
        - !Sub "sqs.${AWS::Region}.amazonaws.com"

```

Outputs:**VPC:**

Description: A reference to the created VPC

Value: !Ref VPC

PublicSubnets:

Description: A list of the public subnets

Value: !Join [",", [!Ref PublicSubnet1, !Ref PublicSubnet2]]

PrivateSubnets:

Description: A list of the private subnets

Value: !Join [",", [!Ref PrivateSubnet1, !Ref PrivateSubnet2]]

PublicSubnet1:

Description: A reference to the public subnet in the 1st Availability Zone

Value: !Ref PublicSubnet1

PublicSubnet2:

Description: A reference to the public subnet in the 2nd Availability Zone

Value: !Ref PublicSubnet2

PrivateSubnet1:

Description: A reference to the private subnet in the 1st Availability Zone

Value: !Ref PrivateSubnet1

PrivateSubnet2:

Description: A reference to the private subnet in the 2nd Availability Zone

Value: !Ref PrivateSubnet2

SecurityGroupIngress:

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

```
MwaaApacheAirflowUI:
```

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

Schritt zwei: Erstellen Sie den Stack mit dem AWS CLI

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem `mwa-public-network.yml` es gespeichert ist. Beispielsweise:

```
cd mwaaproject
```

2. Verwenden Sie den [aws cloudformation create-stack](#) Befehl, um den Stack mit dem zu erstellen AWS CLI.

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

Note

Die Erstellung der Amazon VPC-Infrastruktur, des Amazon S3-Buckets und der Amazon MWSA-Umgebung dauert über 30 Minuten.

Schritt drei: Laden Sie eine DAG auf Amazon S3 hoch und führen Sie sie in der Apache Airflow-Benutzeroberfläche aus

1. Kopieren Sie den Inhalt der `tutorial.py` Datei für die [neueste unterstützte Apache Airflow-Version](#) und speichern Sie sie lokal unter `tutorial.py`
2. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem `tutorial.py` es gespeichert ist. Beispielsweise:

```
cd mwaaproject
```

3. Verwenden Sie den folgenden Befehl, um alle Ihre Amazon S3 S3-Buckets aufzulisten.

```
aws s3 ls
```

4. Verwenden Sie den folgenden Befehl, um die Dateien und Ordner im Amazon S3 S3-Bucket für Ihre Umgebung aufzulisten.

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

5. Verwenden Sie das folgende Skript, um die `tutorial.py` Datei in Ihren `dags` Ordner hochzuladen. Ersetzen Sie den Beispielwert in `YOUR_S3_BUCKET_NAME`.

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

6. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
7. Wählen Sie eine Umgebung aus.
8. Wählen Sie „Airflow-Benutzeroberfläche öffnen“.
9. Wählen Sie auf der Apache Airflow-Benutzeroberfläche aus der Liste der verfügbaren DAGs das Tutorial DAG aus.
10. Klicken Sie auf der Seite mit den DAG-Details neben Ihrem DAG-Namen auf die Option DAG pausieren/pausieren, um die DAG aufzuheben.
11. Wählen Sie „DAG auslösen“.

Vierter Schritt: Logs in CloudWatch Logs anzeigen

Sie können die Apache Airflow-Protokolle für alle Apache Airflow-Protokolle, die durch den Stack aktiviert wurden, in der CloudWatch AWS CloudFormation Konsole anzeigen. Der folgende Abschnitt zeigt, wie Sie Protokolle für die Airflow-Webserver-Protokollgruppe anzeigen.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich Überwachung die Airflow-Webserver-Protokollgruppe aus.
4. Wählen Sie unter `webserver_console_ip` Log-Streams die Option Log Streams aus.

Als nächstes

- Erfahren Sie mehr darüber, wie Sie DAGs hochladen, Python-Abhängigkeiten in a `requirements.txt` und benutzerdefinierte Plugins in `plugins.zip` angeben. [Arbeiten mit DAGs auf Amazon MWAA](#)
- Erfahren Sie mehr über die bewährten Methoden, die wir empfehlen, um die Leistung Ihrer Umgebung zu optimieren. [Leistungsoptimierung für Apache Airflow auf Amazon MWAA](#)
- Erstellen Sie ein Monitoring-Dashboard für Ihre Umgebung unter [Überwachen von Dashboards und Alarmen auf Amazon MWAA](#).
- Führen Sie einige der DAG-Codebeispiele in aus [Codebeispiele für Amazon Managed Workflows für Apache Airflow](#).

Beginnen Sie mit Amazon Managed Workflows for Apache Airflow

Amazon Managed Workflows for Apache Airflow verwendet Amazon VPC, DAG-Code und unterstützende Dateien in Ihrem Amazon S3 S3-Speicher-Bucket, um eine Umgebung zu erstellen. In diesem Handbuch werden die Voraussetzungen und die erforderlichen AWS Ressourcen beschrieben, die für den Einstieg in Amazon MWAA erforderlich sind.

Themen

- [Voraussetzungen](#)
- [Informationen zu diesem Handbuch](#)
- [Bevor Sie beginnen](#)
- [Verfügbare Regionen](#)
- [Erstellen eines Amazon S3 cket cket erstellen](#)
- [Erstellen Sie das VPC-Netzwerk](#)
- [Erstellen Sie eine Amazon MWAA-Umgebung](#)
- [Als nächstes](#)

Voraussetzungen

Um eine Amazon MWAA-Umgebung zu erstellen, sollten Sie möglicherweise zusätzliche Schritte unternehmen, um sicherzustellen, dass Sie über die Berechtigungen für die AWS Ressourcen verfügen, die Sie erstellen müssen.

- **AWS-Konto** — Ein AWS-Konto mit der Erlaubnis, Amazon MWAA und die von Ihrer Umgebung genutzten AWS-Dienste und Ressourcen zu nutzen.

Informationen zu diesem Handbuch

In diesem Abschnitt werden die AWS-Infrastruktur und die Ressourcen beschrieben, die Sie in diesem Handbuch erstellen werden.

- **Amazon VPC** — Die Amazon VPC-Netzwerkkomponenten, die für eine Amazon MWAA-Umgebung erforderlich sind. Sie können eine vorhandene VPC konfigurieren, die diese Anforderungen erfüllt

(erweitert), wie in beschrieben [Informationen zu Netzwerken in Amazon MWAA](#), oder die VPC- und Netzwerkkomponenten erstellen, wie in definiert [the section called “Erstellen Sie das VPC-Netzwerk”](#).

- Amazon S3 S3-Bucket — Ein Amazon S3 S3-Bucket zum Speichern Ihrer DAGs und zugehörigen Dateien wie `plugins.zip` und `requirements.txt`. Ihr Amazon S3 S3-Bucket muss so konfiguriert sein, dass er jeglichen öffentlichen Zugriff blockiert, wobei die Bucket-Versionierung aktiviert ist, wie unter definiert [Erstellen eines Amazon S3 cket cket erstellen](#).
- Amazon MWAA-Umgebung — Eine Amazon MWAA-Umgebung, konfiguriert mit dem Standort Ihres Amazon S3 S3-Buckets, dem Pfad zu Ihrem DAG-Code und allen benutzerdefinierten Plugins oder Python-Abhängigkeiten sowie Ihrer Amazon VPC und ihrer Sicherheitsgruppe, wie in definiert [Erstellen Sie eine Amazon MWAA-Umgebung](#).

Bevor Sie beginnen

Um eine Amazon MWAA-Umgebung zu erstellen, möchten Sie möglicherweise zusätzliche Schritte unternehmen, um andere AWS Ressourcen zu erstellen und zu konfigurieren, bevor Sie Ihre Umgebung erstellen.

Um eine Umgebung zu erstellen, muss Folgendes sichergestellt sein:

- AWS KMS Schlüssel — Ein AWS KMS Schlüssel für die Datenverschlüsselung in Ihrer Umgebung. Sie können die Standardoption auf der Amazon MWAA-Konsole wählen, um beim Erstellen [AWSeiner Umgebung einen eigenen Schlüssel](#) zu erstellen, oder einen vorhandenen, vom [Kunden verwalteten Schlüssel](#) mit Berechtigungen für andere von Ihrer Umgebung verwendete AWS Dienste angeben (erweitert). Weitere Informationen hierzu finden Sie unter [Verwendung von vom Kunden verwalteten Schlüsseln zur Verschlüsselung](#).
- Ausführungsrolle — Eine Ausführungsrolle, die es Amazon MWAA ermöglicht, auf AWS Ressourcen in Ihrer Umgebung zuzugreifen. Sie können die Standardoption auf der Amazon MWAA-Konsole wählen, um beim Erstellen einer Umgebung eine Ausführungsrolle zu erstellen. Weitere Informationen hierzu finden Sie unter [Amazon MWAA-Ausführungsrolle](#).
- VPC-Sicherheitsgruppe — Eine VPC-Sicherheitsgruppe, die es Amazon MWAA ermöglicht, auf andere AWS Ressourcen in Ihrem VPC-Netzwerk zuzugreifen. Sie können die Standardoption auf der Amazon MWAA-Konsole wählen, um beim Erstellen einer Umgebung eine Sicherheitsgruppe zu erstellen, oder eine Sicherheitsgruppe mit den entsprechenden Regeln für eingehenden und ausgehenden Datenverkehr bereitstellen (erweitert). Weitere Informationen hierzu finden Sie unter [Sicherheit in Ihrer VPC auf Amazon MWAA](#).

Verfügbare Regionen

Amazon MWAA ist in den folgenden AWS -Regionen verfügbar.

- Europa (Stockholm) — eu-north-1
- Europa (Frankfurt) - eu-central-1
- Europa (Irland) — eu-west-1
- Europa (London) — eu-west-2
- Europa (Paris) - eu-west-3
- Asien-Pazifik (Mumbai) — ap-south-1
- Asien-Pazifik (Singapur) — ap-southeast-1
- Asien-Pazifik (Sydney) – ap-southeast-2
- Asien-Pazifik (Tokio) – ap-northeast-1
- Asien-Pazifik (Seoul) — ap-northeast-2
- USA Ost (Nord-Virginia) – us-east-1
- USA Ost (Ohio) – us-east-2
- USA West (Oregon) – us-west-2
- Kanada (Zentral)
- Südamerika (São Paulo) - sa-east-1

Erstellen eines Amazon S3 cket cket erstellen

Dieses Handbuch beschreibt die Schritte zur Erstellung eines Amazon S3 S3-Buckets zum Speichern Ihrer Apache Airflow Directed Acyclic Graphs (DAGs), benutzerdefinierter Plug-ins in einer `plugins.zip` Datei und Python-Abhängigkeiten in einer `requirements.txt`

Inhalt

- [Bevor Sie beginnen](#)
- [Erstellen Sie den cket cket cket](#)
- [Als nächstes](#)

Bevor Sie beginnen

- Der Amazon-S3-Bucket kann nicht mehr geändert werden, nachdem Sie den Bucket erstellt haben. Weitere Informationen finden Sie unter [Regeln für die Benennung von Buckets](#) im Amazon Simple Storage Service-Benutzerhandbuch.
- Ein Amazon S3 Bucket, der für eine Amazon MWAA-Umgebung verwendet wird, muss so konfiguriert sein, dass er jeglichen öffentlichen Zugriff blockiert, wobei die Bucket-Versionierung aktiviert sein muss.
- Ein Amazon S3 Bucket, der für eine Amazon MWAA-Umgebung verwendet wird, muss sich in derselben AWS Region wie eine Amazon MWAA-Umgebung befinden. Eine Liste der AWS Regionen für Amazon MWAA finden Sie unter [Amazon MWAA-Endpoints und Kontingente](#) in der Allgemeine AWS-Referenz

Erstellen Sie den Bucket

Dieser Abschnitt beschreibt die Schritte zur Erstellen Sie den Amazon-S3-Bucket.

So erstellen Sie einen Bucket

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter <https://console.aws.amazon.com/s3/>.
2. Wählen Sie Create Bucket (Bucket erstellen).
3. Geben Sie unter Bucket name (Bucket-Name) einen DNS-kompatiblen Namen für Ihren Bucket ein.

Der Bucket-Name muss ...:


- überall in Amazon S3 eindeutig sein.
- zwischen 3 und 63 Zeichen lang sein,
- Enthält keine Großbuchstaben.
- mit einem Kleinbuchstaben oder einer Zahl beginnen.

⚠ Important

Vermeiden Sie, vertrauliche Informationen, wie Kontonummern, in den Bucket-Namen einzubeziehen. Der Bucket-Name wird in der URL angezeigt, die auf die Objekte im Bucket verweist.

4. Wählen Sie unter AWS Region eine Region aus. Dies muss dieselbe AWS Region wie Ihre Amazon MWAA-Umgebung sein.
 - Wir empfehlen, eine Region in der Nähe auszuwählen, um Latenz und Kosten auf einem Minimum.
5. Wählen Sie Block all public access (Öffentlichen Zugriff blockieren) aus.
6. Wählen Sie In Bucket Versioning aktivieren aus.
7. Optional — Schlagworte. Fügen Sie Schlüssel-Wert-Tag-Paare hinzu, um Ihren Amazon S3 S3-Bucket unter Tags zu identifizieren. Zum Beispiel Bucket:Staging.
8. Optional — Serverseitige Verschlüsselung. Sie können optional eine der folgenden Verschlüsselungsoptionen in Ihrem Amazon S3 S3-Bucket aktivieren.
 - a. Wählen Sie unter Serverseitige Verschlüsselung den Amazon-S3-Schlüssel (SSE-S3).
 - b. Wählen Sie AWS Key Management Serviceden Schlüssel (SSE-KMS), um einen AWS KMS Schlüssel für die Verschlüsselung in Ihrem Amazon S3 S3-Bucket zu verwenden:
 - i. AWSverwalteter Schlüssel (aws/s3) — Wenn Sie diese Option wählen, können Sie entweder einen [AWSeigenen Schlüssel](#) verwenden, der von Amazon MWAA verwaltet wird, oder einen vom [Kunden verwalteten Schlüssel](#) für die Verschlüsselung Ihrer Amazon MWAA-Umgebung angeben.
 - ii. Wählen Sie aus Ihren AWS KMS Schlüsseln oder geben Sie den AWS KMS Schlüssel-ARN ein. Wenn Sie in diesem Schritt einen vom [Kunden verwalteten Schlüssel](#) angeben möchten, müssen Sie eine AWS KMS Schlüssel-ID oder einen ARN angeben. [AWS KMS Aliase und Schlüssel mit mehreren Regionen werden von Amazon MWAA nicht unterstützt](#). Der von Ihnen angegebene AWS KMS Schlüssel muss auch für die Verschlüsselung in Ihrer Amazon MWAA-Umgebung verwendet werden.
9. Optional — Erweiterte Einstellungen. Wenn Sie Amazon S3 Object Lock Object Lock aktivieren möchten:

- a. Wählen Sie Erweiterte Einstellungen, Aktivieren.

 **Important**

Durch die Aktivierung von Object Lock können Objekte in diesem Bucket dauerhaft gesperrt werden. Weitere Informationen finden Sie unter [Sperrungen von Amazon S3 Object Lock](#) Simple Storage Service-Benutzerhandbuch.

- b. Wählen Sie die Bestätigung.


10. Wählen Sie Create Bucket (Bucket erstellen) aus.

Als nächstes

- Erfahren Sie, wie Sie das erforderliche Amazon VPC-Netzwerk für eine Umgebung in [Erstellen Sie das VPC-Netzwerk](#) erstellen.
- Wie Sie Zugriffsberechtigungen verwalten, erfahren Sie unter [Wie lege ich ACL-Bucket-Berechtigungen fest?](#)
- Informationen zum Löschen eines Speicherbuckets finden Sie unter [Wie lösche ich einen S3 Bucket?](#)

Erstellen Sie das VPC-Netzwerk

Amazon Managed Workflows for Apache Airflow erfordert eine Amazon VPC und bestimmte Netzwerkkomponenten, um eine Umgebung zu unterstützen. In diesem Handbuch werden die verschiedenen Optionen zum Erstellen des Amazon VPC-Netzwerks für eine Amazon Managed Workflows for Apache Airflow-Umgebung beschrieben.

 **Note**

Apache Airflow funktioniert am besten in einer Netzwerkkomponente mit niedriger Latenz. Wenn Sie eine bestehende Amazon VPC verwenden, die den Datenverkehr in eine andere Region oder in eine lokale Umgebung weiterleitet, empfehlen wir, Folgendes hinzuzufügen: AWS PrivateLink-Endpunkte für Amazon SQS, CloudWatch, Amazon S3, AWS KMS, und Amazon ECR. Für weitere Informationen zur Konfiguration AWS

PrivateLinkInformationen zu Amazon MWAA finden Sie unter [Ein Amazon VPC-Netzwerk ohne Internetzugang erstellen](#).

Inhalt

- [Voraussetzungen](#)
- [Bevor Sie beginnen](#)
- [Optionen zum Erstellen des Amazon VPC-Netzwerks](#)
 - [Option eins: Erstellen des VPC-Netzwerks auf der Amazon MWAA-Konsole](#)
 - [Option zwei: Erstellen eines Amazon VPC-NetzwerksmitInternetzugang](#)
 - [Option drei: Erstellen eines Amazon VPC-NetzwerksohneInternetzugang](#)
- [Als nächstes](#)

Voraussetzungen

Die AWS Command Line Interface (AWS CLI) ist ein Open-Source-Tool, mit dem Sie über Befehle in Ihrer Befehlszeilen-Shell mit den AWS-Services interagieren können. Um die Schritte auf dieser Seite abzuschließen, benötigen Sie Folgendes:

- [AWS CLI— Installieren Sie Version 2](#).
- [AWS CLI— Schnelle Konfiguration mitaws configure](#).

Bevor Sie beginnen

- Das [VPC-Netzwerk](#) Das, was Sie für Ihre Umgebung angeben, kann nach der Erstellung der Umgebung nicht mehr geändert werden.
- Sie können privates oder öffentliches Routing für Ihre Amazon VPC und Apache Airflow verwendenWebserver. Eine Liste der Optionen finden Sie unter [the section called “Beispielanwendungsfälle für einen Amazon-VPC- und Apache-Airflow-Zugriffsmodus”](#).

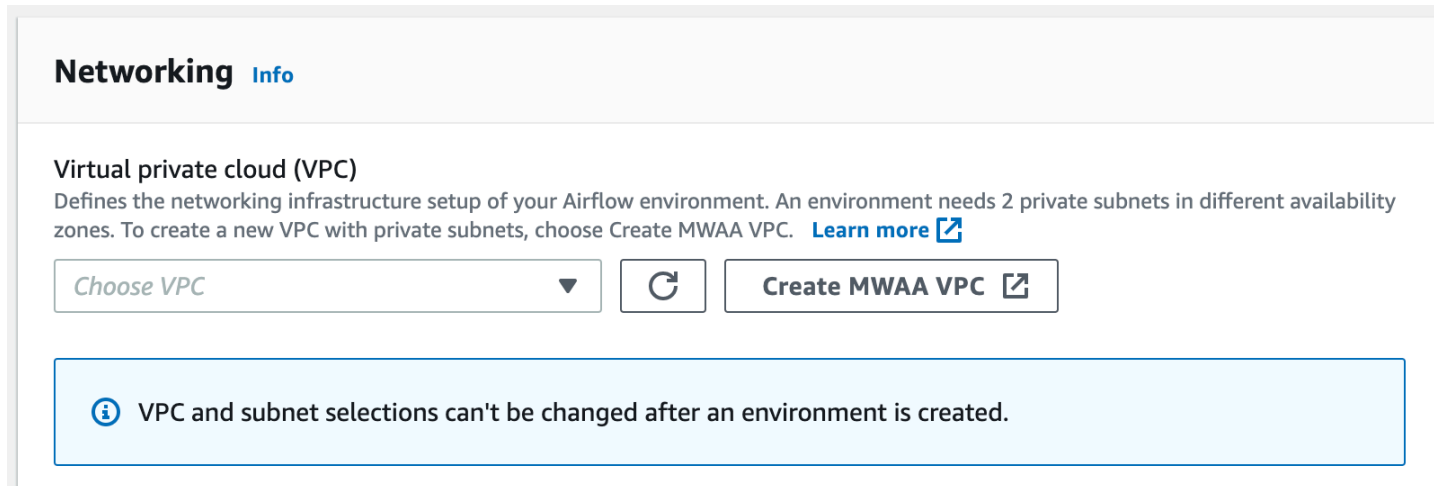
Optionen zum Erstellen des Amazon VPC-Netzwerks

Im folgenden Abschnitt werden die verfügbaren Optionen beschrieben, um das Amazon VPC-Netzwerk für eine Umgebung zu erstellen.

Option eins: Erstellen des VPC-Netzwerks auf der Amazon MWAA-Konsole

Der folgende Abschnitt zeigt, wie Sie ein Amazon VPC-Netzwerk auf der Amazon MWAA-Konsole erstellen. Diese Option verwendet [Öffentliches Routing über das Internet](#). Es kann für einen Apache Airflow verwendet werden Webserver mit dem Privates Netzwerk oder Öffentliches Netzwerk Zugriffsmodi.

Das folgende Bild zeigt, wo Sie die finden MWAA-VPC erstellen Schaltfläche auf der Amazon MWAA-Konsole.



Option zwei: Erstellen eines Amazon VPC-Netzwerks mit Internetzugang

Das Folgende AWS CloudFormation Die Vorlage erstellt ein Amazon VPC-Netzwerk mit Internetzugang in deiner Standardeinstellung AWS Region. Diese Option verwendet [Öffentliches Routing über das Internet](#). Diese Vorlage kann für einen Apache Airflow verwendet werden Webserver mit dem Privates Netzwerk oder Öffentliches Netzwerk Zugriffsmodi.

1. Kopieren Sie den Inhalt der folgenden Vorlage und speichern Sie ihn lokal unter `cf-n-vpc-public-private.yaml`. Du kannst auch [die Vorlage herunterladen](#).

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

Parameters:

EnvironmentName:

Description: An environment name that is prefixed to resource names

Type: String
Default: mwa-

VpcCIDR:

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

PublicSubnet1CIDR:

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

PublicSubnet2CIDR:

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

PrivateSubnet1CIDR:

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

PrivateSubnet2CIDR:

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

Resources:**VPC:**

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

InternetGateway:

```
Type: AWS::EC2::InternetGateway
```

```
Properties:
```

```
Tags:
```

- Key: Name
Value: !Ref EnvironmentName

```
InternetGatewayAttachment:
```

```
Type: AWS::EC2::VPCEGatewayAttachment
```

```
Properties:
```

- InternetGatewayId: !Ref InternetGateway
- VpcId: !Ref VPC

```
PublicSubnet1:
```

```
Type: AWS::EC2::Subnet
```

```
Properties:
```

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

```
PublicSubnet2:
```

```
Type: AWS::EC2::Subnet
```

```
Properties:
```

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

```
PrivateSubnet1:
```

```
Type: AWS::EC2::Subnet
```

```
Properties:
```

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

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

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

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

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

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

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



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

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

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

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

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

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

PrivateRouteTable2:
 Type: AWS::EC2::RouteTable
```

```
Properties:
 VpcId: !Ref VPC
 Tags:
 - Key: Name
 Value: !Sub ${EnvironmentName} Private Routes (AZ2)

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

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

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

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

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

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

PrivateSubnets:
 Description: A list of the private subnets
```

```
Value: !Join [",", [!Ref PrivateSubnet1, !Ref PrivateSubnet2]]
```

**PublicSubnet1:**

Description: A reference to the public subnet in the 1st Availability Zone

Value: !Ref PublicSubnet1

**PublicSubnet2:**

Description: A reference to the public subnet in the 2nd Availability Zone

Value: !Ref PublicSubnet2

**PrivateSubnet1:**

Description: A reference to the private subnet in the 1st Availability Zone

Value: !Ref PrivateSubnet1

**PrivateSubnet2:**

Description: A reference to the private subnet in the 2nd Availability Zone

Value: !Ref PrivateSubnet2

**SecurityGroupIngress:**

Description: Security group with self-referencing inbound rule

Value: !Ref SecurityGroupIngress

2. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem `cfn-vpc-public-private.yaml` gespeichert. Beispiele:

```
cd mwaaproject
```

3. Benutze die [aws cloudformation create-stack](#) Befehl, um den Stack mit dem zu erstellen AWS CLI.

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

**Note**

Die Erstellung der Amazon VPC-Infrastruktur dauert etwa 30 Minuten.

## Option drei: Erstellen eines Amazon VPC-Netzwerk ohne Internetzugang

Das Folgende AWS CloudFormation Die Vorlage erstellt ein Amazon VPC-Netzwerk ohne Internetzugang in deiner Standardeinstellung AWS Region.

### Important

Wenn Sie eine Amazon VPC ohne Internetzugang verwenden, müssen Sie Amazon ECR die Erlaubnis erteilen, über einen Gateway-Endpunkt auf Amazon S3 zuzugreifen. Sie können einen Gateway-Endpunkt erstellen, indem Sie wie folgt vorgehen:

1. Kopieren Sie Folgendes JSON IAM-Richtlinie und speichern Sie sie lokal als `s3-gw-endpoint-policy.json`. Die Richtlinie gewährt Amazon ECR die erforderliche Mindestberechtigung für den Zugriff auf Amazon S3-Ressourcen.

```
{
 "Statement": [
 {
 "Sid": "Access-to-specific-bucket-only",
 "Principal": "*",
 "Action": [
 "s3:GetObject"
],
 "Effect": "Allow",
 "Resource": ["arn:aws:s3:::prod-region-starport-layer-bucket/*"]
 }
]
}
```

2. Erstellen Sie den Endpunkt wie folgt AWS CLI Befehl. Ersetzen Sie die Werte für `--vpc-id` und `--route-table-ids` mit den Informationen für Ihre Amazon VPC. Ersetzen `--service-name` durch den Namen, der Ihrer Region entspricht.

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

Weitere Informationen zur Erstellung von Amazon S3-Gateway-Endpunkten für Amazon ECR finden Sie unter [Erstellen Sie den Amazon S3-Gateway-Endpunkt](#) in der Amazon Elastic Container Registry — Benutzerhandbuch.

Diese Option verwendet [Privates Routing ohne Internetzugang](#). Diese Vorlage kann für einen Apache Airflow verwendet werden, um Webserver mit dem Privates Netzwerk Nur Zugriffsmodus. Es erstellt das Erforderliche [VPC-Endpunkte für die AWS Dienste, die von einer Umgebung verwendet werden](#).

1. Kopieren Sie den Inhalt der folgenden Vorlage und speichern Sie ihn lokal unter `cf-n-vpc-private.yaml`. Du kannst auch [die Vorlage herunterladen](#).

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

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

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

 PrivateSubnet2CIDR:
 Description: The IP range (CIDR notation) for the private subnet in the second
 Availability Zone
 Type: String
 Default: 10.192.11.0/24

Resources:
 VPC:
 Type: AWS::EC2::VPC
 Properties:
 CidrBlock: !Ref VpcCIDR
 EnableDnsSupport: true
 EnableDnsHostnames: true
 Tags:
```

```
- Key: Name
 Value: !Ref AWS::StackName
```

**RouteTable:**

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

**PrivateSubnet1:**

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

**PrivateSubnet2:**

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

**PrivateSubnet1RouteTableAssociation:**

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

**PrivateSubnet2RouteTableAssociation:**

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

**S3VpcEndpoint:**

Type: AWS::EC2::VPCEndpoint

**Properties:**

ServiceName: !Sub "com.amazonaws.\${AWS::Region}.s3"

VpcEndpointType: Gateway

VpcId: !Ref VPC

RouteTableIds:

- !Ref RouteTable

**SecurityGroup:**

Type: AWS::EC2::SecurityGroup

**Properties:**

VpcId: !Ref VPC

GroupDescription: Security Group for Amazon MWA environments to access VPC endpoints

GroupName: !Sub "\${AWS::StackName}-mwa-vpc-endpoints"

**SecurityGroupIngress:**

Type: AWS::EC2::SecurityGroupIngress

**Properties:**

GroupId: !Ref SecurityGroup

IpProtocol: "-1"

SourceSecurityGroupId: !Ref SecurityGroup

**SqsVpcEndpoint:**

Type: AWS::EC2::VPCEndpoint

**Properties:**

ServiceName: !Sub "com.amazonaws.\${AWS::Region}.sqs"

VpcEndpointType: Interface

VpcId: !Ref VPC

PrivateDnsEnabled: true

SubnetIds:

- !Ref PrivateSubnet1
- !Ref PrivateSubnet2

SecurityGroupIds:

- !Ref SecurityGroup

**CloudWatchLogsVpcEndpoint:**

Type: AWS::EC2::VPCEndpoint

**Properties:**

ServiceName: !Sub "com.amazonaws.\${AWS::Region}.logs"

VpcEndpointType: Interface

VpcId: !Ref VPC

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

**CloudWatchMonitoringVpcEndpoint:**

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

**KmsVpcEndpoint:**

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

**EcrApiVpcEndpoint:**

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



```
- !Ref SecurityGroup
```

```
EcrDkrVpcEndpoint:
```

```
Type: AWS::EC2::VPCEndpoint
```

```
Properties:
```

```
ServiceName: !Sub "com.amazonaws.${AWS::Region}.ecr.dkr"
```

```
VpcEndpointType: Interface
```

```
VpcId: !Ref VPC
```

```
PrivateDnsEnabled: true
```

```
SubnetIds:
```

```
- !Ref PrivateSubnet1
```

```
- !Ref PrivateSubnet2
```

```
SecurityGroupIds:
```

```
- !Ref SecurityGroup
```

```
AirflowApiVpcEndpoint:
```

```
Type: AWS::EC2::VPCEndpoint
```

```
Properties:
```

```
ServiceName: !Sub "com.amazonaws.${AWS::Region}.airflow.api"
```

```
VpcEndpointType: Interface
```

```
VpcId: !Ref VPC
```

```
PrivateDnsEnabled: true
```

```
SubnetIds:
```

```
- !Ref PrivateSubnet1
```

```
- !Ref PrivateSubnet2
```

```
SecurityGroupIds:
```

```
- !Ref SecurityGroup
```

```
AirflowEnvVpcEndpoint:
```

```
Type: AWS::EC2::VPCEndpoint
```

```
Properties:
```

```
ServiceName: !Sub "com.amazonaws.${AWS::Region}.airflow.env"
```

```
VpcEndpointType: Interface
```

```
VpcId: !Ref VPC
```

```
PrivateDnsEnabled: true
```

```
SubnetIds:
```

```
- !Ref PrivateSubnet1
```

```
- !Ref PrivateSubnet2
```

```
SecurityGroupIds:
```

```
- !Ref SecurityGroup
```

```
Outputs:
```

```
VPC:
```

```
Description: A reference to the created VPC
```

```
Value: !Ref VPC
```

```
MwaaSecurityGroupId:
```

```
Description: Associates the Security Group to the environment to allow access
to the VPC endpoints
```

```
Value: !Ref SecurityGroup
```

```
PrivateSubnets:
```

```
Description: A list of the private subnets
```

```
Value: !Join [",", [!Ref PrivateSubnet1, !Ref PrivateSubnet2]]
```

```
PrivateSubnet1:
```

```
Description: A reference to the private subnet in the 1st Availability Zone
```

```
Value: !Ref PrivateSubnet1
```

```
PrivateSubnet2:
```

```
Description: A reference to the private subnet in the 2nd Availability Zone
```

```
Value: !Ref PrivateSubnet2
```

2. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem `cfn-vpc-private.yml` gespeichert. Beispiele:

```
cd mwaaproject
```

3. Benutze die [aws cloudformation create-stack](#) Befehl, um den Stack mit dem zu erstellen AWS CLI.

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

#### Note

Die Erstellung der Amazon VPC-Infrastruktur dauert etwa 30 Minuten.

4. Sie müssen einen Mechanismus erstellen, um von Ihrem Computer aus auf diese VPC-Endpunkte zuzugreifen. Weitere Informationen hierzu finden Sie unter [Verwaltung des Zugriffs auf servicespezifische Amazon VPC-Endpunkte auf Amazon MWAA](#).

**Note**

Sie können den ausgehenden Zugriff im CIDR Ihrer Amazon MWAA-Sicherheitsgruppe weiter einschränken. Sie können sich beispielsweise auf sich selbst beschränken, indem Sie eine selbstreferenzierende Regel für ausgehenden Datenverkehr hinzufügen, die [Präfix-Liste](#) für Amazon S3 und den CIDR Ihrer Amazon VPC.

## Als nächstes

- Erfahren Sie in, wie Sie eine Amazon MWAA-Umgebung erstellen [Erstellen Sie eine Amazon MWAA-Umgebung](#).
- Erfahren Sie, wie Sie einen VPN-Tunnel von Ihrem Computer zu Ihrer Amazon VPC mit privatem Routing erstellen können in [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem AWS Client VPN](#).

## Erstellen Sie eine Amazon MWAA-Umgebung

Amazon Managed Workflows for Apache Airflow richtet Apache Airflow in einer Umgebung in der von Ihnen ausgewählten Version ein und verwendet dabei denselben Open-Source-Apache Airflow und dieselbe Benutzeroberfläche, die von Apache erhältlich sind. In diesem Handbuch werden die Schritte zum Erstellen einer Amazon MWAA-Umgebung beschrieben.

### Inhalt

- [Bevor Sie beginnen](#)
- [Apache Airflow-Versionen](#)
- [Erstellen einer Umgebung](#)
  - [Erster Schritt: Geben Sie Details an](#)
  - [Schritt zwei: Konfigurieren Sie erweiterte Einstellungen](#)
  - [Schritt drei: Überprüfen und erstellen](#)

## Bevor Sie beginnen

- Das [VPC-Netzwerk](#), das Sie für Ihre Umgebung angeben, kann nach der Erstellung der Umgebung nicht geändert werden.

- Sie benötigen einen Amazon S3 S3-Bucket, der so konfiguriert ist, dass er den gesamten öffentlichen Zugriff blockiert und die Bucket-Versionierung aktiviert ist.
- Sie benötigen ein AWS Konto mit [Berechtigungen zur Nutzung von Amazon MWSAA](#) und Berechtigungen in AWS Identity and Access Management (IAM), um IAM-Rollen zu erstellen. Wenn Sie den privaten Netzwerkzugriffsmodus für den Apache Airflow-Webserver wählen, der den Apache Airflow-Zugriff innerhalb Ihrer Amazon VPC einschränkt, benötigen Sie eine Genehmigung in IAM, um Amazon VPC-Endpunkte zu erstellen.

## Apache Airflow-Versionen

Die folgenden Apache Airflow-Versionen werden von Amazon Managed Workflows for Apache Airflow unterstützt.

### Note

- Ab Apache Airflow v2.2.2 unterstützt Amazon MWSAA die Installation von Python-Anforderungen, Provider-Paketen und benutzerdefinierten Plugins direkt auf dem Apache Airflow-Webserver.
- Ab Apache Airflow v2.7.2 muss Ihre Anforderungsdatei eine Erklärung enthalten. `--constraint` Wenn Sie keine Einschränkung angeben, gibt Amazon MWSAA eine für Sie an, um sicherzustellen, dass die in Ihren Anforderungen aufgeführten Pakete mit der Version von Apache Airflow kompatibel sind, die Sie verwenden.

Weitere Informationen zum Einrichten von Einschränkungen in Ihrer Anforderungsdatei finden Sie unter [Python-Abhängigkeiten installieren](#).

| Apache Airflow-Version | Apache Airflow-Leitfaden                               | Einschränkungen bei Apache Airflow                                  | Python-Version              |
|------------------------|--------------------------------------------------------|---------------------------------------------------------------------|-----------------------------|
| <a href="#">v2.8.1</a> | <a href="#">Apache Airflow v2.8.1 Referenzhandbuch</a> | <a href="#">Datei mit Einschränkungen für Apache Airflow v2.8.1</a> | <a href="#">Python 3.11</a> |

| Apache Airflow-Version        | Apache Airflow-Leitfaden                               | Einschränkungen bei Apache Airflow                                  | Python-Version              |
|-------------------------------|--------------------------------------------------------|---------------------------------------------------------------------|-----------------------------|
| <a href="#">Version 2.7.2</a> | <a href="#">Apache Airflow v2.7.2 Referenzhandbuch</a> | <a href="#">Datei mit Einschränkungen für Apache Airflow v2.7.2</a> | <a href="#">Python 3.11</a> |
| <a href="#">v2.6.3</a>        | <a href="#">Apache Airflow v2.6.3 Referenzhandbuch</a> | <a href="#">Datei mit Einschränkungen für Apache Airflow v2.6.3</a> | <a href="#">Python 3.10</a> |
| <a href="#">v2.5.1</a>        | <a href="#">Apache Airflow v2.5.1 Referenzhandbuch</a> | <a href="#">Datei mit Einschränkungen für Apache Airflow v2.5.1</a> | <a href="#">Python 3.10</a> |
| <a href="#">v2.4.3</a>        | <a href="#">Apache Airflow v2.4.3 Referenzhandbuch</a> | <a href="#">Datei mit Einschränkungen für Apache Airflow v2.4.3</a> | <a href="#">Python 3.10</a> |
| <a href="#">v2.2.2</a>        | <a href="#">Apache Airflow v2.2.2 Referenzhandbuch</a> | <a href="#">Datei mit Einschränkungen für Apache Airflow v2.2.2</a> | <a href="#">Python 3.7</a>  |
| <a href="#">v2.0.2</a>        | <a href="#">Apache Airflow v2.0.2 Referenzhandbuch</a> | <a href="#">Datei mit Einschränkungen für Apache Airflow v2.0.2</a> | <a href="#">Python 3.7</a>  |

[Weitere Informationen zur Migration Ihrer selbstverwalteten Apache Airflow-Bereitstellungen oder zur Migration einer vorhandenen Amazon MWAA-Umgebung, einschließlich Anweisungen zur Sicherung Ihrer Metadatendatenbank, finden Sie im Amazon MWAA-Migrationsleitfaden.](#)

## Erstellen einer Umgebung

Im folgenden Abschnitt werden die Schritte zum Erstellen einer Amazon MWAA-Umgebung beschrieben.

## Erster Schritt: Geben Sie Details an

Um Details für die Umgebung anzugeben

1. Öffnen Sie die [Amazon MWAA-Konsole](#).
2. Verwenden Sie die AWS Regionsauswahl, um Ihre Region auszuwählen.
3. Wählen Sie Create environment (Umgebung erstellen) aus.
4. Gehen Sie auf der Seite „Details angeben“ unter Umgebungsdetails wie folgt vor:
  - a. Geben Sie im Feld Name einen eindeutigen Namen für Ihre Umgebung ein.
  - b. Wählen Sie die Apache Airflow-Version in der Airflow-Version.

### Note

Wenn kein Wert angegeben ist, wird standardmäßig die neueste Airflow-Version verwendet. Die neueste verfügbare Version ist Apache Airflow v2.8.1.

5. Geben Sie unter DAG-Code in Amazon S3 Folgendes an:
  - a. S3-Bucket. Wählen Sie Browse S3 und wählen Sie Ihren Amazon S3 S3-Bucket aus, oder geben Sie die Amazon S3 S3-URI ein.
  - b. DAGs-Ordner. Wählen Sie Browse S3 und wählen Sie den dags Ordner in Ihrem Amazon S3 S3-Bucket aus, oder geben Sie die Amazon S3 S3-URI ein.
  - c. Plugin-Datei — optional. Wählen Sie Browse S3 und wählen Sie die `plugins.zip` Datei in Ihrem Amazon S3 S3-Bucket aus, oder geben Sie die Amazon S3 S3-URI ein.
  - d. Anforderungsdatei — optional. Wählen Sie Browse S3 und wählen Sie die `requirements.txt` Datei in Ihrem Amazon S3 S3-Bucket aus, oder geben Sie die Amazon S3 S3-URI ein.
  - e. Startskriptdatei — optional, wählen Sie Durchsuchen S3 und wählen Sie die Skriptdatei in Ihrem Amazon S3 S3-Bucket aus, oder geben Sie den Amazon S3 S3-URI ein.
6. Wählen Sie Weiter aus.

## Schritt zwei: Konfigurieren Sie erweiterte Einstellungen

### Konfigurieren von erweiterten Einstellungen

1. Gehen Sie auf der Seite Erweiterte Einstellungen konfigurieren unter Netzwerk wie folgt vor:

- Wählen Sie Ihre [Amazon VPC](#).

In diesem Schritt werden zwei der privaten Subnetze in Ihrer Amazon VPC gefüllt.

2. Wählen Sie unter Webserver-Zugriff Ihren bevorzugten [Apache Airflow-Zugriffsmodus](#) aus:

- a. Privates Netzwerk. Dadurch wird der Zugriff auf die Apache Airflow-Benutzeroberfläche auf Benutzer in Ihrer Amazon VPC beschränkt, denen Zugriff auf die [IAM-Richtlinie für Ihre Umgebung](#) gewährt wurde. Für diesen Schritt benötigen Sie die Erlaubnis, Amazon VPC-Endpoints zu erstellen.

#### Note

Wählen Sie die Option Privates Netzwerk, wenn auf Ihre Apache Airflow-Benutzeroberfläche nur innerhalb eines Unternehmensnetzwerks zugegriffen wird und Sie für die Installation der Webserver-Anforderungen keinen Zugriff auf öffentliche Repositorys benötigen. Wenn Sie diese Option für den Zugriffsmodus wählen, müssen Sie einen Mechanismus für den Zugriff auf Ihren Apache Airflow-Webserver in Ihrer Amazon VPC erstellen. Weitere Informationen finden Sie unter [Zugreifen auf den VPC-Endpoint für Ihren Apache Airflow Webserver \(privater Netzwerkzugriff\)](#).

- b. Öffentliches Netzwerk. Dadurch können Benutzer, denen Zugriff auf die [IAM-Richtlinie für Ihre Umgebung](#) gewährt wurde, über das Internet auf die Apache Airflow-Benutzeroberfläche zugreifen.
3. Wählen Sie unter Sicherheitsgruppe (n) die Sicherheitsgruppe aus, die zum Schutz Ihrer [Amazon VPC](#) verwendet wird:
- a. Standardmäßig erstellt Amazon MWAA eine Sicherheitsgruppe in Ihrer Amazon VPC mit spezifischen Regeln für eingehenden und ausgehenden Datenverkehr unter Neue Sicherheitsgruppe erstellen.
  - b. Optional. Deaktivieren Sie das Kontrollkästchen unter Neue Sicherheitsgruppe erstellen, um bis zu 5 Sicherheitsgruppen auszuwählen.

**Note**

Eine bestehende Amazon VPC-Sicherheitsgruppe muss mit bestimmten Regeln für eingehenden und ausgehenden Datenverkehr konfiguriert werden, um Netzwerkverkehr zuzulassen. Weitere Informationen hierzu finden Sie unter [Sicherheit in Ihrer VPC auf Amazon MWAA](#).

**4. [Wählen Sie unter Umgebungsklasse eine Umgebungsklasse aus.](#)**

Wir empfehlen, die kleinste Größe zu wählen, die zur Unterstützung Ihrer Arbeitslast erforderlich ist. Sie können die Umgebungsklasse jederzeit ändern.

**5. Geben Sie für Maximale Anzahl von Workern die maximale Anzahl von Apache Airflow-Workern an, die in der Umgebung ausgeführt werden sollen.**

Weitere Informationen finden Sie unter [Beispiel für einen Hochleistungs-Anwendungsfall](#).

**6. Geben Sie die maximale Anzahl an Webservern und die minimale Anzahl an Webservern an, um zu konfigurieren, wie Amazon MWAA die Apache Airflow-Webserver in Ihrer Umgebung skaliert.**

Weitere Informationen zur automatischen Skalierung von Webservern finden Sie unter [the section called "Automatische Skalierung von Webservern konfigurieren"](#)

**7. Wählen Sie unter Verschlüsselung eine Datenverschlüsselungsoption aus:**

- a. Standardmäßig verwendet Amazon MWAA einen AWS eigenen Schlüssel, um Ihre Daten zu verschlüsseln.
- b. Optional. Wählen Sie Verschlüsselungseinstellungen anpassen (erweitert), um einen anderen Schlüssel auszuwählen. AWS KMS Wenn Sie in diesem Schritt einen vom [Kunden verwalteten Schlüssel](#) angeben möchten, müssen Sie eine AWS KMS Schlüssel-ID oder einen ARN angeben. [AWS KMS Aliase und Schlüssel für mehrere Regionen werden von Amazon MWAA nicht unterstützt](#). Wenn Sie in Ihrem Amazon S3-Bucket einen Amazon S3-Schlüssel für die serverseitige Verschlüsselung angegeben haben, müssen Sie denselben Schlüssel für Ihre Amazon MWAA-Umgebung angeben.

**Note**

Sie müssen über Berechtigungen für den Schlüssel verfügen, um ihn auf der Amazon MWAA-Konsole auswählen zu können. Sie müssen Amazon MWAA



außerdem Berechtigungen zur Verwendung des Schlüssels erteilen, indem Sie die unter beschriebene Richtlinie anhängen. [Wichtige Richtlinien anhängen](#)

8. Empfohlen. Wählen Sie unter Überwachung eine oder mehrere Protokollkategorien für die Airflow-Protokollierungskonfiguration aus, um Apache Airflow-Protokolle an Logs zu senden: CloudWatch
  - a. Airflow-Aufgabenprotokolle. Wählen Sie den Typ der Apache Airflow-Taskprotokolle aus, die an Logs auf CloudWatch Protokollebene gesendet werden sollen.
  - b. Airflow-Webserver-Protokolle. Wählen Sie den Typ der Apache Airflow-Webserver-Protokolle aus, die an Logs in CloudWatch Log-Ebene gesendet werden sollen.
  - c. Airflow Scheduler-Protokolle. Wählen Sie den Typ der Apache Airflow Scheduler-Protokolle aus, die an Protokolle auf Protokollebene gesendet werden sollen CloudWatch .
  - d. Airflow-Worker-Protokolle. Wählen Sie den Typ der Apache Airflow-Worker-Protokolle aus, die an Logs auf CloudWatch Protokollebene gesendet werden sollen.
  - e. Airflow DAG-Verarbeitungsprotokolle. Wählen Sie den Typ der Apache Airflow DAG-Verarbeitungsprotokolle aus, die an Protokolle auf CloudWatch Protokollebene gesendet werden sollen.
9. Optional. Wählen Sie für Airflow-Konfigurationsoptionen die Option Benutzerdefinierte Konfigurationsoption hinzufügen aus.

Sie können aus der vorgeschlagenen Dropdownliste der [Apache Airflow-Konfigurationsoptionen](#) für Ihre Apache Airflow-Version wählen oder benutzerdefinierte Konfigurationsoptionen angeben. Zum Beispiel: `core.default_task_retries 3`

10. Optional. Wählen Sie unter Tags die Option Neues Tag hinzufügen aus, um Ihrer Umgebung Tags zuzuordnen. Zum Beispiel `Environment:Staging`.
11. Wählen Sie unter Berechtigungen eine Ausführungsrolle aus:
  - a. Standardmäßig erstellt Amazon MWAA unter Neue Rolle erstellen eine [Ausführungsrolle](#). Sie müssen über die Berechtigung zum Erstellen von IAM-Rollen verfügen, um diese Option verwenden zu können.
  - b. Optional. Wählen Sie „Rollen-ARN eingeben“, um den Amazon-Ressourcennamen (ARN) einer vorhandenen Ausführungsrolle einzugeben.
12. Wählen Sie Weiter aus.

## Schritt drei: Überprüfen und erstellen

Um eine Zusammenfassung der Umgebung zu überprüfen

- Sehen Sie sich die Zusammenfassung der Umgebung an und wählen Sie Umgebung erstellen aus.

### Note

Das Erstellen einer Umgebung dauert etwa zwanzig bis dreißig Minuten.

## Als nächstes

- Erfahren Sie, wie Sie einen Amazon S3 Bucket erstellen [Erstellen eines Amazon S3 cket cket erstellen](#).

# Verwaltung des Zugriffs auf eine Amazon MWAA-Umgebung

Amazon Managed Workflows for Apache Airflow muss berechtigt sein, andere AWS Dienste und Ressourcen zu nutzen, die von einer Umgebung genutzt werden. Sie benötigen außerdem die Erlaubnis, auf eine Amazon MWAA-Umgebung und Ihre Apache Airflow-Benutzeroberfläche in AWS Identity and Access Management (IAM) zuzugreifen. In diesem Abschnitt wird die Ausführungsrolle beschrieben, die verwendet wird, um Zugriff auf die AWS Ressourcen für Ihre Umgebung zu gewähren, sowie das Hinzufügen von Berechtigungen sowie die AWS Kontoberechtigungen, die Sie für den Zugriff auf Ihre Amazon MWAA-Umgebung und die Apache Airflow-Benutzeroberfläche benötigen.

## Themen

- [Zugreifen auf eine Amazon MWAA-Umgebung](#)
- [Servicebezogene Rolle für Amazon MWAA](#)
- [Amazon MWAA-Ausführungsrolle](#)
- [Serviceübergreifende Confused-Deputy-Prävention](#)
- [Apache Airflow-Zugriffsmodi](#)

## Zugreifen auf eine Amazon MWAA-Umgebung

Um Amazon Managed Workflows für Apache Airflow verwenden zu können, müssen Sie ein Konto und IAM-Entitäten mit den erforderlichen Berechtigungen verwenden. Auf dieser Seite werden die Zugriffsrichtlinien beschrieben, die Sie Ihrem Apache Airflow-Entwicklungsteam und Apache Airflow-Benutzern für Ihre Amazon Managed Workflows for Apache Airflow-Umgebung zuordnen können.

Wir empfehlen, temporäre Anmeldeinformationen zu verwenden und föderierte Identitäten mit Gruppen und Rollen zu konfigurieren, um auf Ihre Amazon MWAA-Ressourcen zuzugreifen. Es hat sich bewährt, Richtlinien nicht direkt an Ihre IAM-Benutzer anzuhängen, und definieren Sie stattdessen Gruppen oder Rollen, um temporären Zugriff auf Ressourcen zu gewähren. AWS

Eine IAM-[Rolle](#) ist eine IAM-Identität, die Sie in Ihrem Konto mit bestimmten Berechtigungen erstellen können. Eine IAM-Rolle ähnelt einem IAM-Benutzer insofern, als es sich um eine AWS Identität mit Berechtigungsrichtlinien handelt, die festlegen, was die Identität tun kann und was nicht. AWS Eine Rolle ist jedoch nicht einer einzigen Person zugeordnet, sondern kann von allen Personen angenommen werden, die diese Rolle benötigen. Einer Rolle sind außerdem keine standardmäßigen,

langfristigen Anmeldeinformationen (Passwörter oder Zugriffsschlüssel) zugeordnet. Wenn Sie eine Rolle übernehmen, erhalten Sie stattdessen temporäre Anmeldeinformationen für Ihre Rollensitzung.

Um einer föderierten Identität 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.

Sie können eine IAM-Rolle in Ihrem Konto verwenden, um anderen AWS-Konto Berechtigungen für den Zugriff auf die Ressourcen Ihres Kontos zu gewähren. Ein Beispiel finden Sie unter [Tutorial: Delegieren des Zugriffs AWS-Konten mithilfe von IAM-Rollen im IAM-Benutzerhandbuch](#).

## Sections

- [Funktionsweise](#)
- [Vollständige Richtlinie für den Konsolenzugriff: AmazonMWAA FullConsole Access](#)
- [Vollständige API- und Konsolenzugriffsrichtlinie: AmazonMWAA FullApi Access](#)
- [Richtlinie für den Zugriff auf Konsolen mit Schreibschutz: AmazonMWAA Access ReadOnly](#)
- [Zugriffsrichtlinie für die Apache Airflow-Benutzeroberfläche: AmazonMWAA Access WebServer](#)
- [Apache Airflow CLI-Richtlinie: AirflowCli AmazonMWAA Access](#)
- [Eine JSON-Richtlinie erstellen](#)
- [Beispiel für einen Anwendungsfall zum Anhängen von Richtlinien an eine Entwicklergruppe](#)
- [Als nächstes](#)

## Funktionsweise

Die in einer Amazon MWAA-Umgebung verwendeten Ressourcen und Dienste sind nicht für alle AWS Identity and Access Management (IAM-) Entitäten zugänglich. Sie müssen eine Richtlinie erstellen, die Apache Airflow-Benutzern den Zugriff auf diese Ressourcen gewährt. Beispielsweise müssen Sie Ihrem Apache Airflow-Entwicklungsteam Zugriff gewähren.

Amazon MWAA verwendet diese Richtlinien, um zu überprüfen, ob ein Benutzer über die erforderlichen Berechtigungen verfügt, um eine Aktion auf der AWS Konsole oder über die von einer Umgebung verwendeten APIs auszuführen.

Sie können die JSON-Richtlinien in diesem Thema verwenden, um eine Richtlinie für Ihre Apache Airflow-Benutzer in IAM zu erstellen und die Richtlinie dann einem Benutzer, einer Gruppe oder einer Rolle in IAM zuzuordnen.

- [FullConsoleAmazonMWAA-Zugriff](#) — Verwenden Sie diese Richtlinie, um die Erlaubnis zur Konfiguration einer Umgebung auf der Amazon MWAA-Konsole zu erteilen.
- [FullApiAmazonMWAA-Zugriff](#) — Verwenden Sie diese Richtlinie, um Zugriff auf alle Amazon MWAA-APIs zu gewähren, die zur Verwaltung einer Umgebung verwendet werden.
- [ReadOnlyAmazonMWAA-Zugriff](#) — Verwenden Sie diese Richtlinie, um Zugriff auf die Ressourcen zu gewähren, die von einer Umgebung auf der Amazon MWAA-Konsole verwendet werden.
- [WebServerAmazonMWAA-Zugriff](#) — [Verwenden Sie diese Richtlinie, um Zugriff auf](#) den Apache Airflow-Webserver zu gewähren.
- [AirflowCliAmazonMWAA-Zugriff](#) — [Verwenden Sie diese Richtlinie, um Zugriff auf](#) die Ausführung von Apache Airflow CLI-Befehlen zu gewähren.

Um Zugriff zu gewähren, fügen Sie Ihren Benutzern, Gruppen oder Rollen Berechtigungen hinzu:

- Benutzer und Gruppen in: AWS IAM Identity Center

Erstellen Sie einen Berechtigungssatz. Befolgen Sie die Anweisungen unter [Erstellen eines Berechtigungssatzes](#) im AWS IAM Identity Center -Benutzerhandbuch.

- Benutzer, die in IAM über einen Identitätsanbieter verwaltet werden:

Erstellen Sie eine Rolle für den Identitätsverbund. Befolgen Sie die Anweisungen unter [Erstellen einer Rolle für einen externen Identitätsanbieter \(Verbund\)](#) im IAM-Benutzerhandbuch.

- IAM-Benutzer:

- Erstellen Sie eine Rolle, die Ihr Benutzer annehmen kann. Folgen Sie den Anweisungen unter [Erstellen einer Rolle für einen IAM-Benutzer](#) im IAM-Benutzerhandbuch.

- (Nicht empfohlen) Weisen Sie einem Benutzer eine Richtlinie direkt zu oder fügen Sie einen Benutzer zu einer Benutzergruppe hinzu. Befolgen Sie die Anweisungen unter [Hinzufügen von Berechtigungen zu einem Benutzer \(Konsole\)](#) im IAM-Benutzerhandbuch.

## Vollständige Richtlinie für den Konsolenzugriff: AmazonMWAA FullConsole Access

Ein Benutzer benötigt möglicherweise Zugriff auf die `AmazonMWAAFullConsoleAccess` Berechtigungsrichtlinie, wenn er eine Umgebung auf der Amazon MWAA-Konsole konfigurieren muss.

### Note

Ihre vollständige Zugriffsrichtlinie für die Konsole muss die erforderlichen Berechtigungen beinhalten. `iam:PassRole` Auf diese Weise kann der Benutzer [dienstbezogene Rollen](#) und [Ausführungsrollen](#) an Amazon MWAA übergeben. Amazon MWAA übernimmt jede Rolle, um in Ihrem Namen andere AWS Dienste anzurufen. Im folgenden Beispiel wird der `iam:PassedToService` Bedingungsschlüssel verwendet, um den Amazon MWAA-Service Principal (`airflow.amazonaws.com`) als den Service anzugeben, an den eine Rolle übergeben werden kann.

Weitere Informationen `iam:PassRole` dazu finden Sie unter [Erteilen von Benutzerberechtigungen zur Übergabe einer Rolle an einen AWS Service](#) im IAM-Benutzerhandbuch.

Verwenden Sie die folgende Richtlinie, wenn Sie Ihre Amazon MWAA-Umgebungen mithilfe von [AWS-eigener Schlüssel](#) für [Encryption](#) at Rest erstellen und verwalten möchten.

Verwenden Sie ein AWS-eigener Schlüssel

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

```

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

```

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

```



```

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

```

Verwenden Sie die folgende Richtlinie, wenn Sie Ihre Amazon MWAA-Umgebungen mit einem vom [Kunden verwalteten Schlüssel](#) für die Verschlüsselung im Ruhezustand erstellen und verwalten möchten. Um einen vom Kunden verwalteten Schlüssel verwenden zu können, muss der IAM-Principal berechtigt sein, mithilfe des in Ihrem Konto gespeicherten Schlüssels auf AWS KMS Ressourcen zuzugreifen.

#### Verwenden eines vom Kunden verwalteten Schlüssels

```

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

```

```

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

```

```

 "Action":[
 "s3:CreateBucket",
 "s3:PutObject",
 "s3:GetEncryptionConfiguration"
],
 "Resource":"arn:aws:s3:::*"
 },
 {
 "Effect":"Allow",
 "Action":[
 "ec2:DescribeSecurityGroups",
 "ec2:DescribeSubnets",
 "ec2:DescribeVpcs",
 "ec2:DescribeRouteTables"
],
 "Resource": "*"
 },
 {
 "Effect":"Allow",
 "Action":[
 "ec2:AuthorizeSecurityGroupIngress",
 "ec2:CreateSecurityGroup"
],
 "Resource":"arn:aws:ec2:*:*:security-group/airflow-security-group-*"
 },
 {
 "Effect":"Allow",
 "Action":[
 "kms:ListAliases"
],
 "Resource": "*"
 },
 {
 "Effect":"Allow",
 "Action":[
 "kms:DescribeKey",
 "kms:ListGrants",
 "kms:CreateGrant",
 "kms:RevokeGrant",
 "kms:Decrypt",
 "kms:Encrypt",
 "kms:GenerateDataKey*",
 "kms:ReEncrypt*"
],
 },

```

```

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

```

## Vollständige API- und Konsolenzugriffsrichtlinie: AmazonMWAA FullApi Access

Ein Benutzer benötigt möglicherweise Zugriff auf die AmazonMWAAFullApiAccess Berechtigungsrichtlinie, wenn er Zugriff auf alle Amazon MWAA-APIs benötigt, die zur Verwaltung einer Umgebung verwendet werden. Es werden keine Berechtigungen für den Zugriff auf die Apache Airflow-Benutzeroberfläche gewährt.

### Note

Eine vollständige API-Zugriffsrichtlinie muss `iam:PassRole` Leistungsberechtigungen beinhalten. Auf diese Weise kann der Benutzer [dienstbezogene Rollen](#) und [Ausführungsrollen](#) an Amazon MWAA übergeben. Amazon MWAA übernimmt jede Rolle, um in Ihrem Namen andere AWS Dienste anzurufen. Im folgenden Beispiel wird der `iam:PassedToService` Bedingungsschlüssel verwendet, um den Amazon MWAA-Service

Principal (`airflow.amazonaws.com`) als den Service anzugeben, an den eine Rolle übergeben werden kann.

Weitere Informationen `iam:PassRole` dazu finden Sie unter [Erteilen von Benutzerberechtigungen zur Übergabe einer Rolle an einen AWS Service](#) im IAM-Benutzerhandbuch.

Verwenden Sie die folgende Richtlinie, wenn Sie Ihre Amazon MWAA-Umgebungen mithilfe von AWS-eigener Schlüssel for Encryption at Rest erstellen und verwalten möchten.

Mit einem AWS-eigener Schlüssel

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": "airflow:*",
 "Resource": "*"
 },
 {
 "Effect": "Allow",
 "Action": [
 "iam:PassRole"
],
 "Resource": "*",
 "Condition": {
 "StringLike": {
 "iam:PassedToService": "airflow.amazonaws.com"
 }
 }
 },
 {
 "Effect": "Allow",
 "Action": [
 "iam:CreateServiceLinkedRole"
],
 "Resource": "arn:aws:iam::*:role/aws-service-role/airflow.amazonaws.com/AWSServiceRoleForAmazonMWAA"
 },
 {
 "Effect": "Allow",
```

```

 "Action": [
 "ec2:DescribeSecurityGroups",
 "ec2:DescribeSubnets",
 "ec2:DescribeVpcs",
 "ec2:DescribeRouteTables"
],
 "Resource": "*"
 },
 {
 "Effect": "Allow",
 "Action": [
 "s3:GetEncryptionConfiguration"
],
 "Resource": "arn:aws:s3:::*"
 },
 {
 "Effect": "Allow",
 "Action": "ec2:CreateVpcEndpoint",
 "Resource": [
 "arn:aws:ec2:*:*:vpc-endpoint/*",
 "arn:aws:ec2:*:*:vpc/*",
 "arn:aws:ec2:*:*:subnet/*",
 "arn:aws:ec2:*:*:security-group/*"
]
 },
 {
 "Effect": "Allow",
 "Action": [
 "ec2:CreateNetworkInterface"
],
 "Resource": [
 "arn:aws:ec2:*:*:subnet/*",
 "arn:aws:ec2:*:*:network-interface/*"
]
 }
]
}

```

Verwenden Sie die folgende Richtlinie, wenn Sie Ihre Amazon MWAA-Umgebungen mit einem vom Kunden verwalteten Schlüssel für die Verschlüsselung im Ruhezustand erstellen und verwalten möchten. Um einen vom Kunden verwalteten Schlüssel verwenden zu können, muss der IAM-Principal berechtigt sein, mithilfe des in Ihrem Konto gespeicherten Schlüssels auf AWS KMS Ressourcen zuzugreifen.

## Verwenden eines vom Kunden verwalteten Schlüssels

```
{
 "Version":"2012-10-17",
 "Statement":[
 {
 "Effect":"Allow",
 "Action":"airflow:*",
 "Resource":"*"
 },
 {
 "Effect":"Allow",
 "Action":[
 "iam:PassRole"
],
 "Resource":"*",
 "Condition":{"
 "StringLike":{"
 "iam:PassedToService":"airflow.amazonaws.com"
 }
 }
 }
],
 {
 "Effect":"Allow",
 "Action":[
 "iam:CreateServiceLinkedRole"
],
 "Resource":"arn:aws:iam::*:role/aws-service-role/airflow.amazonaws.com/
AWSServiceRoleForAmazonMWSAA"
 },
 {
 "Effect":"Allow",
 "Action":[
 "ec2:DescribeSecurityGroups",
 "ec2:DescribeSubnets",
 "ec2:DescribeVpcs",
 "ec2:DescribeRouteTables"
],
 "Resource":"*"
 },
 {
 "Effect":"Allow",
 "Action":[
 "kms:DescribeKey",
```

```

 "kms:ListGrants",
 "kms:CreateGrant",
 "kms:RevokeGrant",
 "kms:Decrypt",
 "kms:Encrypt",
 "kms:GenerateDataKey*",
 "kms:ReEncrypt*"
],
 "Resource": "arn:aws:kms:*:YOUR_ACCOUNT_ID:key/YOUR_KMS_ID"
},
{
 "Effect": "Allow",
 "Action": [
 "s3:GetEncryptionConfiguration"
],
 "Resource": "arn:aws:s3:::*"
},
{
 "Effect": "Allow",
 "Action": "ec2:CreateVpcEndpoint",
 "Resource": [
 "arn:aws:ec2:*:*:vpc-endpoint/*",
 "arn:aws:ec2:*:*:vpc/*",
 "arn:aws:ec2:*:*:subnet/*",
 "arn:aws:ec2:*:*:security-group/*"
]
},
{
 "Effect": "Allow",
 "Action": [
 "ec2:CreateNetworkInterface"
],
 "Resource": [
 "arn:aws:ec2:*:*:subnet/*",
 "arn:aws:ec2:*:*:network-interface/*"
]
}
]
}

```



## Richtlinie für den Zugriff auf Konsolen mit Schreibschutz: AmazonMWAA Access ReadOnly

Ein Benutzer benötigt möglicherweise Zugriff auf die `AmazonMWAAReadOnlyAccess` Berechtigungsrichtlinie, wenn er die von einer Umgebung verwendeten Ressourcen auf der Detailseite der Amazon MWAA-Konsolenumgebung einsehen möchte. Es erlaubt einem Benutzer nicht, neue Umgebungen zu erstellen, bestehende Umgebungen zu bearbeiten oder einem Benutzer das Anzeigen der Apache Airflow-Benutzeroberfläche zu ermöglichen.

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

## Zugriffsrichtlinie für die Apache Airflow-Benutzeroberfläche: AmazonMWAA Access WebServer

Ein Benutzer benötigt möglicherweise Zugriff auf die `AmazonMWAAWebServerAccess` Berechtigungsrichtlinie, wenn er auf die Apache Airflow-Benutzeroberfläche zugreifen muss. Es erlaubt dem Benutzer nicht, Umgebungen auf der Amazon MWAA-Konsole anzusehen oder die Amazon MWAA-APIs zu verwenden, um Aktionen auszuführen. Geben Sie die `Admin`, `OpUser`, `Viewer` oder die `Public` Rolle an, in `{airflow-role}` der Sie die Zugriffsebene für den Benutzer des Web-Tokens anpassen möchten. Weitere Informationen finden Sie unter [Standardrollen](#) im Apache Airflow-Referenzhandbuch.

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

```

 "Effect": "Allow",
 "Action": "airflow:CreateWebLoginToken",
 "Resource": [
 "arn:aws:airflow:{your-region}:YOUR_ACCOUNT_ID:role/{your-environment-
name}/{airflow-role}"
]
 }
]
}

```

### Note

Amazon MWAA bietet IAM-Integration mit den fünf [standardmäßigen Apache Airflow-Rollen für die rollenbasierte Zugriffskontrolle](#) (RBAC). Weitere Informationen zur Arbeit mit benutzerdefinierten Apache Airflow-Rollen finden Sie unter [the section called “Tutorial: Benutzer auf eine Teilmenge von DAGs beschränken”](#)

## Apache Airflow CLI-Richtlinie: AirflowCli AmazonMWAA Access

Ein Benutzer benötigt möglicherweise Zugriff auf die AmazonMWAAirflowCliAccess Berechtigungsrichtlinie, wenn er Apache Airflow CLI-Befehle (z. B. `trigger_dag`) ausführen muss. Es erlaubt dem Benutzer nicht, Umgebungen auf der Amazon MWAA-Konsole anzusehen oder die Amazon MWAA-APIs zu verwenden, um Aktionen auszuführen.

```

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

```

## Eine JSON-Richtlinie erstellen

Sie können die JSON-Richtlinie erstellen und die Richtlinie an Ihren Benutzer, Ihre Rolle oder Ihre Gruppe auf der IAM-Konsole anhängen. In den folgenden Schritten wird beschrieben, wie Sie eine JSON-Richtlinie in IAM erstellen.

Um die JSON-Richtlinie zu erstellen

1. Öffnen Sie die [Seite Richtlinien](#) in der IAM-Konsole.
2. Wählen Sie Richtlinie erstellen aus.
3. Wählen Sie den Tab JSON.
4. Fügen Sie Ihre JSON-Richtlinie hinzu.
5. Wählen Sie Richtlinie prüfen.
6. Geben Sie einen Wert in das Textfeld für Name und Beschreibung ein (optional).

Sie könnten der Richtlinie beispielsweise einen Namen geben `AmazonMWAAReadOnlyAccess`.

7. Wählen Sie Richtlinie erstellen aus.

## Beispiel für einen Anwendungsfall zum Anhängen von Richtlinien an eine Entwicklergruppe

Nehmen wir an, Sie verwenden eine Gruppe in IAM mit dem Namen `AirflowDevelopmentGroup`, um allen Entwicklern in Ihrem Apache Airflow-Entwicklungsteam Berechtigungen zuzuweisen. Diese Benutzer benötigen Zugriff auf die `AmazonMWAACliAccess`, `AmazonMWAAReadOnlyAccess` und `AmazonMWAAServerAccess` Berechtigungsrichtlinien. In diesem Abschnitt wird beschrieben, wie Sie eine Gruppe in IAM erstellen, diese Richtlinien erstellen und anhängen und die Gruppe einem IAM-Benutzer zuordnen. Bei den Schritten wird davon ausgegangen, dass Sie einen [AWS eigenen](#) Schlüssel verwenden.

Um die `FullConsoleAccess AmazonMWAACliAccess`-Richtlinie zu erstellen

1. Laden Sie die [FullConsoleAccess AmazonMWAACliAccess-Zugriffsrichtlinie](#) herunter.
2. Öffnen Sie die [Seite Richtlinien](#) in der IAM-Konsole.
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie den Tab JSON.

5. Fügen Sie die JSON-Richtlinie für ein `AmazonMWAACFullConsoleAccess`.
6. Ersetzen Sie die folgenden Werte:
  - a. *{your-account-id}* – Ihre AWS Konto-ID (z. B.) 0123456789
  - b. *{your-kms-id}* — Die eindeutige Kennung für einen vom Kunden verwalteten Schlüssel. Gilt nur, wenn Sie einen vom Kunden verwalteten Schlüssel für die Verschlüsselung im Ruhezustand verwenden.
7. Wählen Sie die Bewertungsrichtlinie aus.
8. Geben Sie `AmazonMWAACFullConsoleAccess` Name ein.
9. Wählen Sie Richtlinie erstellen aus.

Um die `WebServerAccess` AmazonMWAARichtlinie zu erstellen

1. Laden Sie die [WebServerAccess AmazonMWAAC-Zugriffsrichtlinie](#) herunter.
2. Öffnen Sie die [Seite Richtlinien](#) in der IAM-Konsole.
3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie den Tab JSON.
5. Fügen Sie die JSON-Richtlinie für ein `AmazonMWAACWebServerAccess`.
6. Ersetzen Sie die folgenden Werte:
  - a. *{your-region}* — die Region Ihrer Amazon MWAA-Umgebung (z. B.) us-east-1
  - b. *{your-account-id}* – Ihre Konto-ID (z. B.) AWS 0123456789
  - c. *{your-environment-name}* — Ihr Amazon MWAA-Umgebungsname (z. B.) MyAirflowEnvironment
  - d. [{airflow-role} – die Apache Airflow-Standardrolle Admin](#)
7. Wählen Sie Richtlinie prüfen.
8. Geben Sie Name ein `AmazonMWAACWebServerAccess`.
9. Wählen Sie Richtlinie erstellen aus.

Um die `AirflowCliAccess` AmazonMWAARichtlinie zu erstellen

1. Laden Sie die [AirflowCliAccess AmazonMWAAC-Zugriffsrichtlinie](#) herunter.
2. Öffnen Sie die [Seite Richtlinien](#) in der IAM-Konsole.

3. Wählen Sie Richtlinie erstellen aus.
4. Wählen Sie den Tab JSON.
5. Fügen Sie die JSON-Richtlinie für einAmazonMWAACliAccess ein.
6. Wählen Sie die Überprüfungsrichtlinie aus.
7. Geben Sie AmazonMWAACliAccess Name ein.
8. Wählen Sie Richtlinie erstellen aus.

Um die Gruppe zu erstellen

1. Öffnen Sie die [Seite Gruppen](#) in der IAM-Konsole.
2. Geben Sie einen Namen von AirflowDevelopmentGroup ein.
3. Wählen Sie Next Step (Weiter) aus.
4. Geben Sie in AmazonMWAACliAccess das Feld Filter ein, um die Ergebnisse zu filtern.
5. Wählen Sie die drei Richtlinien aus, die Sie erstellt haben.
6. Wählen Sie Next Step (Weiter) aus.
7. Wählen Sie Create Group.

Um sie einem Benutzer zuzuordnen

1. Öffnen Sie die [Seite Benutzer](#) in der IAM-Konsole.
2. Wählen Sie einen Benutzer aus.
3. Klicken Sie auf Groups (Gruppen).
4. Wählen Sie Benutzer zu Gruppen hinzufügen.
5. Wählen Sie die AirflowDevelopmentGruppe aus.
6. Wählen Sie dann Add to Groups (Zu Gruppen hinzufügen) aus.

## Als nächstes

- Erfahren Sie unter, wie Sie ein Token für den Zugriff auf die Apache Airflow-Benutzeroberfläche generieren. [Zugreifen auf Apache Airflow](#)
- Weitere Informationen zum Erstellen von IAM-Richtlinien finden Sie unter IAM-Richtlinien [erstellen](#).

## Servicebezogene Rolle für Amazon MWAA

Amazon Managed Workflows for Apache Airflow verwendet AWS Identity and Access Management (IAM) [service-verknüpfte](#) Rollen. Eine serviceverknüpfte Rolle ist eine einzigartige Art von IAM-Rolle, die direkt mit Amazon MWAA verknüpft ist. Servicebezogene Rollen sind von Amazon MWAA vordefiniert und beinhalten alle Berechtigungen, die der Service benötigt, um andere AWS Services in Ihrem Namen aufzurufen.

Eine serviceverknüpfte Rolle erleichtert die Einrichtung von Amazon MWAA, da Sie die erforderlichen Berechtigungen nicht manuell hinzufügen müssen. Amazon MWAA definiert die Berechtigungen seiner serviceverknüpften Rollen, und sofern nicht anders definiert, kann nur Amazon MWAA seine Rollen übernehmen. Die definierten Berechtigungen umfassen die Vertrauens- und Berechtigungsrichtlinie. Diese Berechtigungsrichtlinie kann keinen anderen IAM-Entitäten zugewiesen werden.

Sie können eine serviceverknüpfte Rolle erst löschen, nachdem ihre verwandten Ressourcen gelöscht wurden. Dadurch werden Ihre Amazon MWAA-Ressourcen geschützt, da Sie die Zugriffsberechtigung für die Ressourcen nicht versehentlich entziehen können.

Informationen zu anderen Services, die serviceverknüpfte Rollen unterstützen, finden Sie unter [AWS -Services, die mit IAM funktionieren](#). Suchen Sie nach den Services, für die Ja in der Spalte Serviceverknüpfte Rollen angegeben ist. Wählen Sie über einen Link Ja aus, um die Dokumentation zu einer serviceverknüpften Rolle für diesen Service anzuzeigen.

## Servicebezogene Rollenberechtigungen für Amazon MWAA

Amazon MWAA verwendet die serviceverknüpfte Rolle mit dem Namen `AWSServiceRoleForAmazonMWAA` — Die in Ihrem Konto erstellte serviceverknüpfte Rolle gewährt Amazon MWAA Zugriff auf die folgenden Dienste: AWS

- Amazon CloudWatch Logs (CloudWatch Logs) — Um Protokollgruppen für Apache Airflow-Protokolle zu erstellen.
- Amazon CloudWatch (CloudWatch) — Um Metriken zu Ihrer Umgebung und den zugrundeliegenden Komponenten in Ihrem Konto zu veröffentlichen.
- Amazon Elastic Compute Cloud (Amazon EC2) — Um die folgenden Ressourcen zu erstellen:
  - Ein Amazon VPC-Endpoint in Ihrer VPC für einen AWS verwalteten Amazon Aurora PostgreSQL-Datenbankcluster, der vom Apache Airflow Scheduler and Worker verwendet werden soll.

- Ein zusätzlicher Amazon VPC-Endpunkt, um den Netzwerkzugriff auf den Webserver zu ermöglichen, wenn Sie die [private Netzwerkooption](#) für Ihren Apache Airflow-Webserver wählen.
- [Elastic Network Interfaces \(ENIs\)](#) in Ihrer Amazon VPC, um den Netzwerkzugriff auf AWS Ressourcen zu ermöglichen, die in Ihrer Amazon VPC gehostet werden.

Die folgende Vertrauensrichtlinie ermöglicht es dem Service Principal, die dienstbezogene Rolle zu übernehmen. Der Service Principal für Amazon MWAA ist in der Richtlinie `airflow.amazonaws.com` dargelegt.

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

Die genannte Rollenberechtigungsrichtlinie `AmazonMWAAServiceRolePolicy` ermöglicht es Amazon MWAA, die folgenden Aktionen für die angegebenen Ressourcen durchzuführen:

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": [
 "logs:CreateLogStream",
 "logs:CreateLogGroup",
 "logs:DescribeLogGroups"
],
 "Resource": "arn:aws:logs:*:*:log-group:airflow-*:*"
 },
 {
 "Effect": "Allow",
 "Action": [
 "ec2:AttachNetworkInterface",

```

```

 "ec2:CreateNetworkInterface",
 "ec2:CreateNetworkInterfacePermission",
 "ec2>DeleteNetworkInterface",
 "ec2>DeleteNetworkInterfacePermission",
 "ec2:DescribeDhcpOptions",
 "ec2:DescribeNetworkInterfaces",
 "ec2:DescribeSecurityGroups",
 "ec2:DescribeSubnets",
 "ec2:DescribeVpcEndpoints",
 "ec2:DescribeVpcs",
 "ec2:DetachNetworkInterface"
],
 "Resource": "*"
},
{
 "Effect": "Allow",
 "Action": "ec2:CreateVpcEndpoint",
 "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
 "Condition": {
 "ForAnyValue:StringEquals": {
 "aws:TagKeys": "AmazonMWAAManaged"
 }
 }
},
{
 "Effect": "Allow",
 "Action": [
 "ec2:ModifyVpcEndpoint",
 "ec2>DeleteVpcEndpoints"
],
 "Resource": "arn:aws:ec2:*:*:vpc-endpoint/*",
 "Condition": {
 "Null": {
 "aws:ResourceTag/AmazonMWAAManaged": false
 }
 }
},
{
 "Effect": "Allow",
 "Action": [
 "ec2:CreateVpcEndpoint",
 "ec2:ModifyVpcEndpoint"
],
 "Resource": [

```



```

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

```

Sie müssen Berechtigungen konfigurieren, damit eine juristische Stelle von IAM (z. B. Benutzer, Gruppe oder Rolle) eine serviceverknüpfte Rolle erstellen, bearbeiten oder löschen kann. Weitere Informationen finden Sie unter [Serviceverknüpfte Rollenberechtigung](#) im IAM-Benutzerhandbuch.

## Eine serviceverknüpfte Rolle für Amazon MWAA erstellen

Sie müssen eine serviceverknüpfte Rolle nicht manuell erstellen. Wenn Sie mithilfe der, der oder der AWS Management Console AWS API eine neue Amazon MWAA-Umgebung erstellen AWS CLI, erstellt Amazon MWAA die serviceverknüpfte Rolle für Sie.

Wenn Sie diese serviceverknüpfte Rolle löschen und sie dann erneut erstellen müssen, können Sie dasselbe Verfahren anwenden, um die Rolle in Ihrem Konto neu anzulegen. Wenn Sie eine andere Umgebung erstellen, erstellt Amazon MWAA die serviceverknüpfte Rolle erneut für Sie.

## Bearbeiten einer serviceverknüpften Rolle für Amazon MWAA

Amazon MWAA erlaubt Ihnen nicht, die `AWSServiceRoleForAmazonMWAA` serviceverknüpfte Rolle zu bearbeiten. Da möglicherweise verschiedene Entitäten auf die Rolle verweisen, kann der Rollename nach dem Erstellen einer serviceverknüpften Rolle nicht mehr geändert werden. Sie können jedoch die Beschreibung der Rolle mit IAM bearbeiten. Weitere Informationen finden Sie unter [Bearbeiten einer serviceverknüpften Rolle](#) im IAM-Benutzerhandbuch.

## Löschen einer serviceverknüpften Rolle für Amazon MWAA

Wenn Sie ein Feature oder einen Dienst, die bzw. der eine serviceverknüpften Rolle erfordert, nicht mehr benötigen, sollten Sie diese Rolle löschen. Auf diese Weise haben Sie keine ungenutzte juristische Stelle, die nicht aktiv überwacht oder verwaltet wird.

Wenn Sie eine Amazon MWAA-Umgebung löschen, löscht Amazon MWAA alle zugehörigen Ressourcen, die es als Teil des Service verwendet. Sie müssen jedoch warten, bis Amazon MWAA das Löschen Ihrer Umgebung abgeschlossen hat, bevor Sie versuchen, die serviceverknüpfte Rolle zu löschen. Wenn Sie die serviceverknüpfte Rolle löschen, bevor Amazon MWAA die Umgebung löscht, kann Amazon MWAA möglicherweise nicht alle zugehörigen Ressourcen der Umgebung löschen.

So löschen Sie die serviceverknüpfte Rolle mit IAM

Verwenden Sie die IAM-Konsole, die oder die API, um die AWS CLI serviceverknüpfte Rolle zu löschen `AWS . AWSServiceRoleForAmazonMWAA`. Weitere Informationen finden Sie unter [Löschen einer serviceverknüpften Rolle](#) im IAM-Benutzerhandbuch.

## Unterstützte Regionen für Rollen im Zusammenhang mit Amazon MWAA, die mit dem Service verknüpft sind

Amazon MWAA unterstützt die Verwendung von servicebezogenen Rollen in allen Regionen, in denen der Service verfügbar ist. Weitere Informationen finden Sie unter [Amazon Managed Workflows für Apache Airflow-Endpunkte und Kontingente](#).

## Richtlinienaktualisierungen

| Änderung                                                                            | Beschreibung                                                                                                                                                                                                                                                                                                                                           | Datum             |
|-------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| Amazon MWAA aktualisiert seine Berechtigungsrichtlinie für serviceverknüpfte Rollen | <a href="#">AmazonMWAAServiceRolePolicy</a> — Amazon MWAA aktualisiert die Berechtigungsrichtlinie für seine servicebezogene Rolle, um Amazon MWAA die Erlaubnis zu erteilen, zusätzliche Kennzahlen zu den dem Service zugrunde liegenden Ressourcen auf Kundenkonten zu veröffentlichen. Diese neuen Kennzahlen werden veröffentlicht unter AWS/MWAA | 18. November 2022 |
| Amazon MWAA hat begonnen, Änderungen nachzuverfolgen                                | Amazon MWAA hat damit begonnen, Änderungen an seiner Berechtigungsrichtlinie für Rollen im Zusammenhang mit AWS verwalteten Diensten nachzuverfolgen.                                                                                                                                                                                                  | 18. November 2022 |

## Amazon MWAA-Ausführungsrolle

Eine Ausführungsrolle ist eine AWS Identity and Access Management (IAM) -Rolle mit einer Berechtigungsrichtlinie, die Amazon Managed Workflows for Apache Airflow die Erlaubnis erteilt, die Ressourcen anderer AWS Services in Ihrem Namen aufzurufen. Dies kann Ressourcen wie Ihren Amazon S3 S3-Bucket, Ihren [AWS eigenen Schlüssel](#) und CloudWatch Logs beinhalten. Amazon MWAA-Umgebungen benötigen eine Ausführungsrolle pro Umgebung. Auf dieser Seite wird beschrieben, wie Sie die Ausführungsrolle für Ihre Umgebung verwenden und konfigurieren, damit Amazon MWAA auf andere von Ihrer Umgebung verwendete AWS Ressourcen zugreifen kann.

### Inhalt

- [Übersicht über die Ausführungsrollen](#)
  - [Standardmäßig zugeordnete Berechtigungen](#)
  - [Wie füge ich die Erlaubnis zur Nutzung anderer Dienste hinzu AWS](#)
  - [Wie ordnet man eine neue Ausführungsrolle zu](#)
- [Create a new role \(Neue Rolle erstellen\)](#)
- [Eine Ausführungsrollenrichtlinie anzeigen und aktualisieren](#)
  - [Hängen Sie eine JSON-Richtlinie an, um andere AWS Dienste zu nutzen](#)
- [Gewähren Sie Zugriff auf den Amazon S3 S3-Bucket mit Sperrung des öffentlichen Zugriffs auf Kontoebene](#)
- [Verwenden Sie Apache Airflow-Verbindungen](#)
- [JSON-Beispielrichtlinien für eine Ausführungsrolle](#)
  - [Beispielrichtlinie für einen vom Kunden verwalteten Schlüssel](#)
  - [Beispielrichtlinie für einen AWS eigenen Schlüssel](#)
- [Als nächstes](#)

## Übersicht über die Ausführungsrollen

Die Erlaubnis für Amazon MWAA, andere von Ihrer Umgebung verwendete AWS Dienste zu nutzen, erhalten Sie von der Ausführungsrolle. Eine Amazon MWAA-Ausführungsrolle benötigt Berechtigungen für die folgenden AWS Dienste, die von einer Umgebung verwendet werden:

- Amazon CloudWatch (CloudWatch) — um Apache Airflow-Metriken und -Protokolle zu senden.
- Amazon Simple Storage Service (Amazon S3) — um den DAG-Code Ihrer Umgebung und unterstützende Dateien (z. B. `requirements.txt`) zu analysieren.
- Amazon Simple Queue Service (Amazon SQS) — zur Warteschlange der Apache Airflow-Aufgaben Ihrer Umgebung in einer Amazon SQS SQS-Warteschlange, die Amazon MWAA gehört.
- AWS Key Management Service ([AWS KMS](#)) — [für die Datenverschlüsselung Ihrer Umgebung \(entweder mit einem AWS eigenen Schlüssel oder mit Ihrem vom Kunden verwalteten Schlüssel\).](#)

### Note

Wenn Sie sich dafür entschieden haben, dass Amazon MWAA einen AWS verwalteten KMS-Schlüssel zur Verschlüsselung Ihrer Daten verwendet, müssen Sie in einer Richtlinie, die Ihrer Amazon MWAA-Ausführungsrolle zugeordnet ist, Berechtigungen definieren,

die Zugriff auf beliebige KMS-Schlüssel gewähren, die außerhalb Ihres Kontos über Amazon SQS gespeichert sind. Die folgenden zwei Bedingungen sind erforderlich, damit die Ausführungsrolle Ihrer Umgebung auf beliebige KMS-Schlüssel zugreifen kann:

- Ein KMS-Schlüssel in einem Drittanbieterkonto muss diesen kontoübergreifenden Zugriff über dessen Ressourcenrichtlinie ermöglichen.
- Ihr DAG-Code muss auf eine Amazon SQS SQS-Warteschlange zugreifen, die mit dem `airflow-celery`-Drittanbieter-Konto beginnt und denselben KMS-Schlüssel für die Verschlüsselung verwendet.

Um die mit dem kontoübergreifenden Zugriff auf Ressourcen verbundenen Risiken zu minimieren, empfehlen wir, den Code in Ihren DAGs zu überprüfen, um sicherzustellen, dass Ihre Workflows nicht auf willkürliche Amazon SQS SQS-Warteschlangen außerhalb Ihres Kontos zugreifen. Darüber hinaus können Sie einen vom Kunden verwalteten KMS-Schlüssel verwenden, der in Ihrem eigenen Konto gespeichert ist, um die Verschlüsselung auf Amazon MWAA zu verwalten. Dadurch wird die Ausführungsrolle Ihrer Umgebung darauf beschränkt, nur auf den KMS-Schlüssel in Ihrem Konto zuzugreifen.

Beachten Sie, dass Sie Ihre Auswahl für eine bestehende Umgebung nicht mehr ändern können, nachdem Sie eine Verschlüsselungsoption ausgewählt haben.

Eine Ausführungsrolle benötigt außerdem Berechtigungen für die folgenden IAM-Aktionen:

- `airflow:PublishMetrics`— um es Amazon MWAA zu ermöglichen, den Zustand einer Umgebung zu überwachen.

## Standardmäßig zugeordnete Berechtigungen

Sie können die Standardoptionen auf der Amazon MWAA-Konsole verwenden, um eine Ausführungsrolle und einen [AWS eigenen Schlüssel](#) zu erstellen. Verwenden Sie dann die Schritte auf dieser Seite, um Ihrer Ausführungsrolle Berechtigungsrichtlinien hinzuzufügen.

- Wenn Sie in der Konsole die Option Neue Rolle erstellen wählen, ordnet Amazon MWAA Ihrer Ausführungsrolle die für eine Umgebung erforderlichen Mindestberechtigungen zu.
- In einigen Fällen fügt Amazon MWAA die maximalen Berechtigungen hinzu. Wir empfehlen beispielsweise, die Option auf der Amazon MWAA-Konsole auszuwählen, um beim Erstellen einer Umgebung eine Ausführungsrolle zu erstellen. Amazon MWAA fügt die Berechtigungsrichtlinien für alle CloudWatch Logs-Gruppen automatisch hinzu, indem es das Regex-Muster in der

```
Ausführungsrolle als verwendet. "arn:aws:logs:your-region:your-account-id:log-group:airflow-your-environment-name-*
```

## Wie füge ich die Erlaubnis zur Nutzung anderer Dienste hinzu AWS

Amazon MWAA kann einer vorhandenen Ausführungsrolle keine Berechtigungsrichtlinien hinzufügen oder bearbeiten, nachdem eine Umgebung erstellt wurde. Sie müssen Ihre Ausführungsrolle mit zusätzlichen Berechtigungsrichtlinien aktualisieren, die für Ihre Umgebung erforderlich sind. Wenn Ihre DAG beispielsweise Zugriff auf benötigt AWS Glue, kann Amazon MWAA nicht automatisch erkennen, dass diese Berechtigungen für Ihre Umgebung erforderlich sind, oder die Berechtigungen Ihrer Ausführungsrolle hinzufügen.

Sie können einer Ausführungsrolle auf zwei Arten Berechtigungen hinzufügen:

- Indem Sie die JSON-Richtlinie für Ihre Ausführungsrolle direkt ändern. Sie können die [JSON-Richtliniendokumente](#) auf dieser Seite verwenden, um die JSON-Richtlinie Ihrer Ausführungsrolle in der IAM-Konsole entweder zu ergänzen oder zu ersetzen.
- Indem Sie eine JSON-Richtlinie für einen AWS Service erstellen und sie an Ihre Ausführungsrolle anhängen. Mithilfe der Schritte auf dieser Seite können Sie Ihrer Ausführungsrolle in der IAM-Konsole ein neues JSON-Richtliniendokument für einen AWS Service zuordnen.

Unter der Annahme, dass die Ausführungsrolle bereits mit Ihrer Umgebung verknüpft ist, kann Amazon MWAA sofort damit beginnen, die hinzugefügten Berechtigungsrichtlinien zu verwenden. Das bedeutet auch, dass Ihre DAGs fehlschlagen können, wenn Sie einer Ausführungsrolle alle erforderlichen Berechtigungen entziehen.

## Wie ordnet man eine neue Ausführungsrolle zu

Sie können die Ausführungsrolle für Ihre Umgebung jederzeit ändern. Wenn Ihrer Umgebung noch keine neue Ausführungsrolle zugeordnet ist, verwenden Sie die Schritte auf dieser Seite, um eine neue Ausführungsrollenrichtlinie zu erstellen und die Rolle Ihrer Umgebung zuzuordnen.

## Create a new role (Neue Rolle erstellen)

Standardmäßig erstellt Amazon MWAA in Ihrem Namen einen [AWS eigenen Schlüssel](#) für die Datenverschlüsselung und eine Ausführungsrolle. Sie können die Standardoptionen auf der Amazon MWAA-Konsole auswählen, wenn Sie eine Umgebung erstellen. Die folgende Abbildung zeigt die Standardoption zum Erstellen einer Ausführungsrolle für eine Umgebung.

## Permissions [Info](#)

### Execution role

The IAM role used by your environment to access your DAG code, write logs, and perform other actions.

Create a new role



### Role name

AmazonMWAA-MyAirflowEnvironment-rdfjhHm

Use alphanumeric and '+=, @-\_' characters. Maximum 64 characters.

## Eine Ausführungsrollenrichtlinie anzeigen und aktualisieren

Sie können die Ausführungsrolle für Ihre Umgebung auf der Amazon MWAA-Konsole anzeigen und die JSON-Richtlinie für die Rolle auf der IAM-Konsole aktualisieren.

Um eine Richtlinie für eine Ausführungsrolle zu aktualisieren

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich „Berechtigungen“ die Ausführungsrolle aus, um die Seite mit den Berechtigungen in IAM zu öffnen.
4. Wählen Sie den Namen der Ausführungsrolle, um die Berechtigungsrichtlinie zu öffnen.
5. Wählen Sie Edit policy (Richtlinie bearbeiten).
6. Wählen Sie den Tab JSON.
7. Aktualisieren Sie Ihre JSON-Richtlinie.
8. Wählen Sie Richtlinie prüfen.
9. Wählen Sie Änderungen speichern aus.

## Hängen Sie eine JSON-Richtlinie an, um andere AWS Dienste zu nutzen

Sie können eine JSON-Richtlinie für einen AWS Dienst erstellen und sie an Ihre Ausführungsrolle anhängen. Sie können beispielsweise die folgende JSON-Richtlinie anhängen, um schreibgeschützten Zugriff auf alle Ressourcen in zu gewähren. AWS Secrets Manager

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

So fügen Sie Ihrer Ausführungsrolle eine Richtlinie hinzu

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich „Berechtigungen“ Ihre Ausführungsrolle aus.
4. Wählen Sie Richtlinien anfügen.
5. Wählen Sie Create Policy (Richtlinie erstellen) aus.
6. Wählen Sie JSON.
7. Fügen Sie die JSON-Richtlinie ein.
8. Wählen Sie Weiter: Tags, Weiter: Überprüfen aus.
9. Geben Sie einen aussagekräftigen Namen (z. B. `SecretsManagerReadPolicy`) und eine Beschreibung für die Richtlinie ein.
10. Wählen Sie Richtlinie erstellen aus.

## Gewähren Sie Zugriff auf den Amazon S3 S3-Bucket mit Sperrung des öffentlichen Zugriffs auf Kontoebene

Möglicherweise möchten Sie den Zugriff auf alle Buckets in Ihrem Konto mithilfe des [PutPublicAccessBlock](#) Amazon S3 S3-Vorgangs sperren. Wenn Sie den Zugriff



auf alle Buckets in Ihrem Konto sperren, muss Ihre Umgebungsausführungsrolle die `s3:GetAccountPublicAccessBlock` Aktion in einer Berechtigungsrichtlinie enthalten.

Das folgende Beispiel zeigt die Richtlinie, die Sie Ihrer Ausführungsrolle zuordnen müssen, wenn Sie den Zugriff auf alle Amazon S3 S3-Buckets in Ihrem Konto blockieren.

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

Weitere Informationen zur Beschränkung des Zugriffs auf Ihre Amazon S3 S3-Buckets finden Sie unter [Blockieren des öffentlichen Zugriffs auf Ihren Amazon S3 S3-Speicher](#) im Amazon Simple Storage Service-Benutzerhandbuch.

## Verwenden Sie Apache Airflow-Verbindungen

Sie können auch eine Apache Airflow-Verbindung erstellen und Ihre Ausführungsrolle und deren ARN in Ihrem Apache Airflow-Verbindungsobjekt angeben. Weitere Informationen hierzu finden Sie unter [Verbindungen zu Apache Airflow verwalten](#).

## JSON-Beispielrichtlinien für eine Ausführungsrolle

Die Beispielberechtigungsrichtlinien in diesem Abschnitt zeigen zwei Richtlinien, mit denen Sie die für Ihre bestehende Ausführungsrolle verwendete Berechtigungsrichtlinie ersetzen oder eine neue Ausführungsrolle erstellen und für Ihre Umgebung verwenden können. Diese Richtlinien enthalten [Ressourcen-ARN-Platzhalter](#) für Apache Airflow-Protokollgruppen, einen [Amazon S3 S3-Bucket](#) und eine [Amazon MWAA-Umgebung](#).

Wir empfehlen, die Beispielrichtlinie zu kopieren, die Beispiel-ARNs oder Platzhalter zu ersetzen und dann die JSON-Richtlinie zu verwenden, um eine Ausführungsrolle zu erstellen oder zu aktualisieren. Zum Beispiel das Ersetzen durch `{your-region}. us-east-1`

## Beispielrichtlinie für einen vom Kunden verwalteten Schlüssel

Das folgende Beispiel zeigt eine Ausführungsrollenrichtlinie, die Sie für einen vom [Kunden verwalteten Schlüssel](#) verwenden können.

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

```

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

```

```

 "StringLike": {
 "kms:ViaService": [
 "sqs.{your-region}.amazonaws.com",
 "s3.{your-region}.amazonaws.com"
]
 }
 }
}

```

Als Nächstes müssen Sie Amazon MWA erlauben, diese Rolle zu übernehmen, um Aktionen in Ihrem Namen durchzuführen. [Dies kann geschehen, indem "airflow.amazonaws.com" Sie mithilfe der IAM-Konsole "airflow-env.amazonaws.com" Service Principals zur Liste der vertrauenswürdigen Entitäten für diese Ausführungsrolle hinzufügen oder indem Sie diese Service Principals mithilfe des IAM-Befehls create-role in das Dokument mit der Richtlinie „Rolle übernehmen“ für diese Ausführungsrolle aufnehmen.](#) AWS CLI Ein Beispiel für ein Dokument mit der Richtlinie „Rolle annehmen“ finden Sie unten:

```

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

```

Fügen Sie dann Ihrem vom [Kunden verwalteten Schlüssel](#) die folgende JSON-Richtlinie bei. Diese Richtlinie verwendet das [kms:EncryptionContext](#)Bedingungsschlüsselpräfix, um den Zugriff auf Ihre Apache Airflow-Protokollgruppe in CloudWatch Logs zu ermöglichen.

```

{
 "Sid": "Allow logs access",
 "Effect": "Allow",
 "Principal": {
 "Service": "logs.{your-region}.amazonaws.com"
 }
}

```

```

 },
 "Action": [
 "kms:Encrypt*",
 "kms:Decrypt*",
 "kms:ReEncrypt*",
 "kms:GenerateDataKey*",
 "kms:Describe*"
],
 "Resource": "*",
 "Condition": {
 "ArnLike": {
 "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:{your-region}:{your-
account-id}:*"
 }
 }
 }
}

```

## Beispielrichtlinie für einen AWS eigenen Schlüssel

Das folgende Beispiel zeigt eine Ausführungsrollenrichtlinie, die Sie für einen [AWS eigenen Schlüssel](#) verwenden können.

```

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

```

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

```

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

```

## Als nächstes

- Erfahren Sie mehr über die erforderlichen Berechtigungen, die Sie und Ihre Apache Airflow-Benutzer für den Zugriff auf Ihre Umgebung benötigen. [Zugreifen auf eine Amazon MWAA-Umgebung](#)
- Erfahren Sie mehr über [Verwendung von vom Kunden verwalteten Schlüsseln zur Verschlüsselung](#).
- Entdecken Sie weitere [Beispiele für vom Kunden verwaltete Richtlinien](#).

## Serviceübergreifende Confused-Deputy-Prävention

Das Confused-Deputy-Problem ist ein Sicherheitsproblem, bei dem eine juristische Stelle, die nicht über die Berechtigung zum Ausführen einer Aktion verfügt, eine privilegiere juristische Stelle zwingen kann, die Aktion auszuführen. In AWS kann ein dienstübergreifendes Identitätswechsels zu einem Problem mit dem verwirrten Stellvertreter führen. Ein dienstübergreifender Identitätswechsel kann auftreten, wenn ein Dienst (der Anruf-Dienst) einen anderen Dienst anruft (den aufgerufenen Dienst). Der aufrufende Service kann manipuliert werden, um seine Berechtigungen zu verwenden, um Aktionen auf die Ressourcen eines anderen Kunden auszuführen, für die er sonst keine Zugriffsberechtigung haben sollte. Um dies zu verhindern, bietet AWS Tools, mit denen Sie Ihre Daten für alle Services mit Serviceprinzipalen schützen können, die Zugriff auf Ressourcen in Ihrem Konto erhalten haben.

Wir empfehlen, die Kontextschlüssel `aws:SourceArn` und die `aws:SourceAccount` globalen Bedingungsschlüssel in der Ausführungsrolle Ihrer Umgebung zu verwenden, um die Berechtigungen einzuschränken, die Amazon MWAA einem anderen Service für den Zugriff auf die Ressource gewährt. Verwenden Sie `aws:SourceArn`, wenn Sie nur eine Ressource mit dem betriebsübergreifenden Zugriff verknüpfen möchten. Verwenden Sie `aws:SourceAccount`, wenn Sie zulassen möchten, dass Ressourcen in diesem Konto mit der betriebsübergreifenden Verwendung verknüpft werden.

Der effektivste Weg, um sich vor dem Confused-Deputy-Problem zu schützen, ist die Verwendung des globalen Bedingungskontext-Schlüssels `aws:SourceArn` mit dem vollständigen ARN der Ressource. Wenn Sie den vollständigen ARN der Ressource nicht kennen oder wenn Sie mehrere Ressourcen angeben, verwenden Sie den globalen Kontextbedingungsschlüssel `aws:SourceArn` mit Platzhalterzeichen (\*) für die unbekannt Teile des ARN. z. B. `arn:aws:airflow:*:123456789012:environment/*`.

Der Wert von `aws:SourceArn` muss Ihr ARN für die Amazon MWAA-Umgebung sein, für den Sie eine Ausführungsrolle erstellen.

Das folgende Beispiel zeigt, wie Sie die Kontextschlüssel `aws:SourceArn` und die `aws:SourceAccount` globalen Bedingungsschlüssel in der Vertrauensrichtlinie für Ausführungsrollen Ihrer Umgebung verwenden können, um das Problem des verwirrten Stellvertreters zu verhindern. Sie können die folgende Vertrauensrichtlinie verwenden, wenn Sie eine neue Ausführungsrolle erstellen.

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



```
"Statement": [
 {
 "Effect": "Allow",
 "Principal": {
 "Service": ["airflow.amazonaws.com", "airflow-env.amazonaws.com"]
 },
 "Action": "sts:AssumeRole",
 "Condition": {
 "ArnLike": {
 "aws:SourceArn": "arn:aws:airflow:your-
region:123456789012:environment/your-environment-name"
 },
 "StringEquals": {
 "aws:SourceAccount": "123456789012"
 }
 }
 }
]
```

## Apache Airflow-Zugriffsmodi

Die Amazon Managed Workflows for Apache Airflow-Konsole enthält integrierte Optionen zur Konfiguration von privatem oder öffentlichem Routing zum Apache Airflow-Webserver in Ihrer Umgebung. Dieses Handbuch beschreibt die Zugriffsmodi, die für den Apache Airflow-Webserver in Ihrer Amazon Managed Workflows for Apache Airflow-Umgebung verfügbar sind, und die zusätzlichen Ressourcen, die Sie in Ihrer Amazon VPC konfigurieren müssen, wenn Sie die private Netzwerkooption wählen.

### Inhalt

- [Apache Airflow-Zugriffsmodi](#)
  - [Öffentliches Netzwerk](#)
  - [Privates Netzwerk](#)
- [Übersicht über die Zugriffsmodi](#)
  - [Öffentlicher Netzwerkzugriffsmodus](#)
  - [Privater Netzwerkzugriffsmodus](#)
- [Einrichtung für private und öffentliche Zugriffsmodi](#)
  - [Einrichtung für ein öffentliches Netzwerk](#)

- [Einrichtung für ein privates Netzwerk](#)
- [Zugreifen auf den VPC-Endpunkt für Ihren Apache Airflow Webserver \(privater Netzwerkzugriff\)](#)

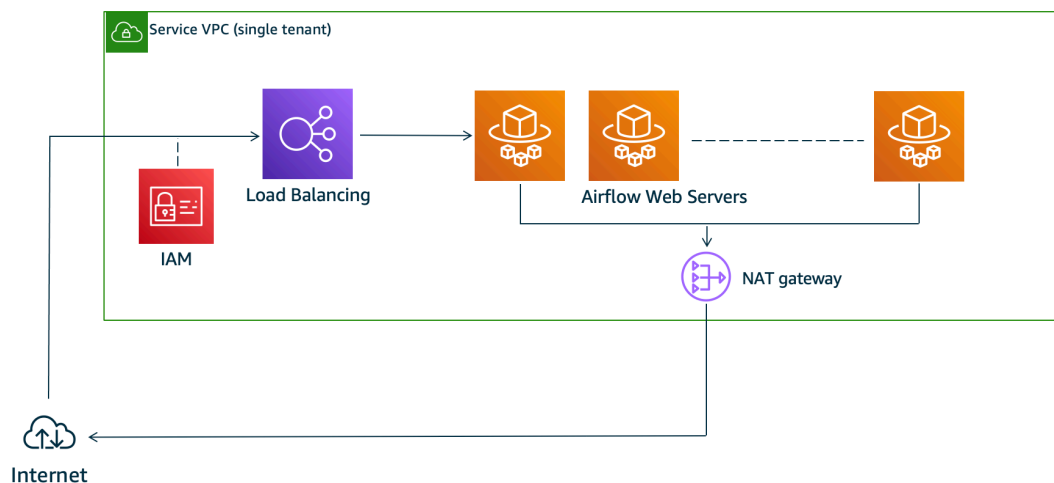
## Apache Airflow-Zugriffsmodi

Sie können zwischen privatem oder öffentlichem Routing für Ihren Apache Airflow-Webserver wählen. Um privates Routing zu aktivieren, wählen Sie Privates Netzwerk. Dadurch wird der Benutzerzugriff auf einen Apache Airflow-Webserver auf eine Amazon VPC beschränkt. Um öffentliches Routing zu aktivieren, wählen Sie Öffentliches Netzwerk. Dadurch können Benutzer über das Internet auf den Apache Airflow-Webserver zugreifen.

### Öffentliches Netzwerk

Das folgende Architekturdiagramm zeigt eine Amazon MWAA-Umgebung mit einem öffentlichen Webserver.

### Public Web Server Option



Im öffentlichen Netzwerkzugriffsmodus können Benutzer, denen Zugriff auf die [IAM-Richtlinie](#) für Ihre Umgebung gewährt wurde, über das Internet auf die Apache Airflow-Benutzeroberfläche zugreifen.

Die folgende Abbildung zeigt, wo Sie die Option Öffentliches Netzwerk auf der Amazon MWAA-Konsole finden.

## Web server access

### Private network (Recommended)

Additional setup required. Your Airflow UI can only be accessed by secure login behind your VPC. Choose this option if your Airflow UI is only accessed within a corporate network. IAM must be used to handle user authentication.

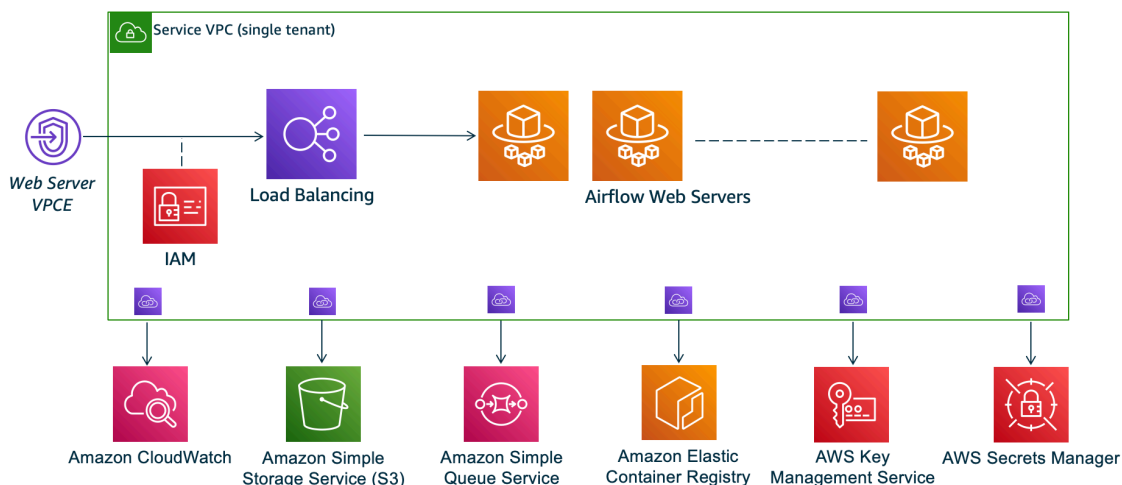
### Public network (No additional setup)

Your Airflow UI can be accessed by secure login over the Internet. Choose this option if your Airflow UI is accessed outside of a corporate network. IAM must be used to handle user authentication.

## Privates Netzwerk

Das folgende Architekturdiagramm zeigt eine Amazon MWAA-Umgebung mit einem privaten Webserver.

### Private Web Server Option



Der private Netzwerkzugriffsmodus beschränkt den Zugriff auf die Apache Airflow-Benutzeroberfläche auf Benutzer in Ihrer Amazon VPC, denen Zugriff auf die [IAM-Richtlinie für Ihre Umgebung](#) gewährt wurde.

Wenn Sie eine Umgebung mit privatem Webserverzugriff erstellen, müssen Sie alle Ihre Abhängigkeiten in einem Python-Radarchiv (.whl) verpacken und dann auf das .whl in Ihrem `requirements.txt` verweisen. Anweisungen zum Paketieren und Installieren Ihrer Abhängigkeiten mit Wheel finden Sie unter [Abhängigkeiten mit Python Wheel verwalten](#).

Die folgende Abbildung zeigt, wo Sie die Option Privates Netzwerk auf der Amazon MWAA-Konsole finden.

## Web server access

**Private network (Recommended)**

Additional setup required. Your Airflow UI can only be accessed by secure login behind your VPC. Choose this option if your Airflow UI is only accessed within a corporate network. IAM must be used to handle user authentication.

**Public network (No additional setup)**

Your Airflow UI can be accessed by secure login over the Internet. Choose this option if your Airflow UI is accessed outside of a corporate network. IAM must be used to handle user authentication.

## Übersicht über die Zugriffsmodi

In diesem Abschnitt werden die VPC-Endpunkte (AWS PrivateLink) beschrieben, die in Ihrer Amazon VPC erstellt wurden, wenn Sie den Zugriffsmodus Öffentliches Netzwerk oder Privates Netzwerk wählen.

### Öffentlicher Netzwerkzugriffsmodus

Wenn Sie den Modus Öffentlicher Netzwerkzugriff für Ihren Apache Airflow-Webserver gewählt haben, wird der Netzwerkverkehr öffentlich über das Internet geleitet.

- Amazon MWAA erstellt einen VPC-Schnittstellenendpunkt für Ihre Amazon Aurora PostgreSQL-Metadatendatenbank. Der Endpunkt wird in den Availability Zones erstellt, die Ihren privaten Subnetzen zugeordnet sind, und ist unabhängig von anderen Konten. AWS
- Amazon MWAA bindet dann eine IP-Adresse aus Ihren privaten Subnetzen an die Schnittstellenendpunkte. Dies soll die bewährte Methode unterstützen, eine einzelne IP aus jeder Availability Zone der Amazon VPC zu binden.

### Privater Netzwerkzugriffsmodus

Wenn Sie den privaten Netzwerkzugriffsmodus für Ihren Apache Airflow-Webserver gewählt haben, wird der Netzwerkverkehr innerhalb Ihrer Amazon VPC privat weitergeleitet.

- Amazon MWAA erstellt einen VPC-Schnittstellenendpunkt für Ihren Apache Airflow-Webserver und einen Schnittstellenendpunkt für Ihre Amazon Aurora PostgreSQL-Metadatendatenbank. Die Endpunkte werden in den Availability Zones erstellt, die Ihren privaten Subnetzen zugeordnet sind, und sind unabhängig von anderen Konten. AWS

- Amazon MWAA bindet dann eine IP-Adresse aus Ihren privaten Subnetzen an die Schnittstellenendpunkte. Dies soll die bewährte Methode unterstützen, eine einzelne IP aus jeder Availability Zone der Amazon VPC zu binden.

Weitere Informationen hierzu finden Sie unter [the section called “Beispielanwendungsfälle für einen Amazon-VPC- und Apache-Airflow-Zugriffsmodus”](#).

## Einrichtung für private und öffentliche Zugriffsmodi

Im folgenden Abschnitt werden die zusätzlichen Einstellungen und Konfigurationen beschrieben, die Sie je nach dem Apache Airflow-Zugriffsmodus benötigen, den Sie für Ihre Umgebung ausgewählt haben.

### Einrichtung für ein öffentliches Netzwerk

Wenn Sie die Option Öffentliches Netzwerk für Ihren Apache Airflow-Webserver wählen, können Sie nach dem Erstellen Ihrer Umgebung mit der Verwendung der Apache Airflow-Benutzeroberfläche beginnen.

Sie müssen die folgenden Schritte ausführen, um den Zugriff für Ihre Benutzer und die Erlaubnis für Ihre Umgebung zur Nutzung anderer AWS Dienste zu konfigurieren.

1. Fügen Sie Berechtigungen hinzu. Amazon MWAA benötigt eine Genehmigung zur Nutzung anderer AWS Dienste. Wenn Sie eine Umgebung erstellen, erstellt Amazon MWAA eine [serviceverknüpfte Rolle](#), die es ihr ermöglicht, bestimmte IAM-Aktionen für Amazon Elastic Container Registry (Amazon ECR), CloudWatch Logs und Amazon EC2 zu verwenden.

Sie können die Erlaubnis zur Verwendung zusätzlicher Aktionen für diese Services oder zur Nutzung anderer AWS Services hinzufügen, indem Sie Ihrer Ausführungsrolle Berechtigungen hinzufügen. Weitere Informationen hierzu finden Sie unter [Amazon MWAA-Ausführungsrolle](#).

2. Erstellen Sie Benutzerrichtlinien. Möglicherweise müssen Sie mehrere IAM-Richtlinien für Ihre Benutzer erstellen, um den Zugriff auf Ihre Umgebung und die Apache Airflow-Benutzeroberfläche zu konfigurieren. Weitere Informationen hierzu finden Sie unter [Zugreifen auf eine Amazon MWAA-Umgebung](#).

## Einrichtung für ein privates Netzwerk

Wenn Sie die Option Privates Netzwerk für Ihren Apache Airflow-Webserver wählen, müssen Sie den Zugriff für Ihre Benutzer und die Erlaubnis für Ihre Umgebung zur Nutzung anderer AWS Dienste konfigurieren und einen Mechanismus für den Zugriff auf die Ressourcen in Ihrer Amazon VPC von Ihrem Computer aus einrichten.

1. Fügen Sie Berechtigungen hinzu. Amazon MWAA benötigt eine Genehmigung zur Nutzung anderer AWS Dienste. Wenn Sie eine Umgebung erstellen, erstellt Amazon MWAA eine [serviceverknüpfte Rolle](#), die es ihr ermöglicht, bestimmte IAM-Aktionen für Amazon Elastic Container Registry (Amazon ECR), CloudWatch Logs und Amazon EC2 zu verwenden.

Sie können die Erlaubnis zur Verwendung zusätzlicher Aktionen für diese Services oder zur Nutzung anderer AWS Services hinzufügen, indem Sie Ihrer Ausführungsrolle Berechtigungen hinzufügen. Weitere Informationen hierzu finden Sie unter [Amazon MWAA-Ausführungsrolle](#).

2. Erstellen Sie Benutzerrichtlinien. Möglicherweise müssen Sie mehrere IAM-Richtlinien für Ihre Benutzer erstellen, um den Zugriff auf Ihre Umgebung und die Apache Airflow-Benutzeroberfläche zu konfigurieren. Weitere Informationen hierzu finden Sie unter [Zugreifen auf eine Amazon MWAA-Umgebung](#).
3. Aktivieren Sie den Netzwerkzugriff. Sie müssen in Ihrer Amazon VPC einen Mechanismus erstellen, um eine Verbindung zum VPC-Endpunkt (AWS PrivateLink) für Ihren Apache Airflow-Webserver herzustellen. Zum Beispiel, indem Sie mit einem VPN-Tunnel von Ihrem Computer aus erstellen. [AWS Client VPN](#)

## Zugreifen auf den VPC-Endpunkt für Ihren Apache Airflow Webserver (privater Netzwerkzugriff)

Wenn Sie die Option Privates Netzwerk ausgewählt haben, müssen Sie in Ihrer Amazon VPC einen Mechanismus für den Zugriff auf den VPC-Endpunkt (AWS PrivateLink) für Ihren Apache Airflow-Webserver einrichten. Wir empfehlen, für diese Ressourcen dieselbe Amazon VPC, VPC-Sicherheitsgruppe und dieselben privaten Subnetze wie Ihre Amazon MWAA-Umgebung zu verwenden.

Weitere Informationen finden Sie unter [Zugriff für VPC-Endpoints verwalten](#).

# Zugreifen auf Apache Airflow

Mit Amazon MWAA können Sie mit mehreren Methoden auf Ihre Apache Airflow-Umgebung zugreifen: über die Apache Airflow-Benutzerschnittstelle (UI), die Apache Airflow CLI und die Apache Airflow REST API. Sie können die Amazon MWAA-Konsole verwenden, um eine DAG in Ihrer Apache Airflow-Benutzeroberfläche anzuzeigen und aufzurufen, oder Amazon MWAA-APIs verwenden, um ein Token abzurufen und eine DAG aufzurufen. In diesem Abschnitt werden die Berechtigungen beschrieben, die für den Zugriff auf die Apache Airflow-Benutzeroberfläche erforderlich sind, wie Sie ein Token generieren, um Amazon MWAA-API-Aufrufe direkt in Ihrer Befehlsshell durchzuführen, und die unterstützten Befehle in der Apache Airflow CLI.

## Themen

- [Voraussetzungen](#)
- [Öffnen Sie die Airflow-Benutzeroberfläche](#)
- [Bei Apache Airflow anmelden](#)
- [Erstellen Sie ein Apache Airflow-Webserver-Zugriffstoken](#)
- [Ein Apache Airflow CLI-Token erstellen](#)
- [Verwenden der Apache Airflow REST API](#)
- [Befehlsreferenz für Apache Airflow CLI](#)

## Voraussetzungen

Im folgenden Abschnitt werden die vorbereitenden Schritte beschrieben, die zur Verwendung der Befehle und Skripts in diesem Abschnitt erforderlich sind.

## Zugriff

- AWS Kontozugriff in AWS Identity and Access Management (IAM) auf die Amazon MWAA-Berechtigungsrichtlinie in. [Zugriffsrichtlinie für die Apache Airflow-Benutzeroberfläche: AmazonMWAA Access WebServer](#)
- AWS Kontozugriff in AWS Identity and Access Management (IAM) auf die Amazon MWAA-Berechtigungsrichtlinie. [Vollständige API- und Konsolenzugriffsrichtlinie: AmazonMWAA FullApi Access](#)

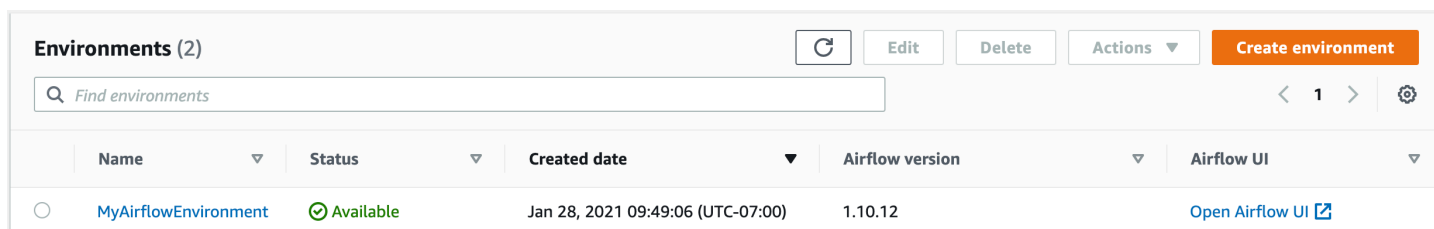
## AWS CLI

Das AWS Command Line Interface (AWS CLI) ist ein Open-Source-Tool, mit dem Sie mithilfe von Befehlen in Ihrer AWS Befehlszeilen-Shell mit Diensten interagieren können. Um die Schritte auf dieser Seite abzuschließen, benötigen Sie Folgendes:

- [AWS CLI — Installieren Sie Version 2.](#)
- [AWS CLI — Schnelle Konfiguration mit `aws configure`.](#)

## Öffnen Sie die Airflow-Benutzeroberfläche

Die folgende Abbildung zeigt den Link zu Ihrer Apache Airflow-Benutzeroberfläche auf der Amazon MWAA-Konsole.



| Name                 | Status    | Created date                      | Airflow version | Airflow UI                      |
|----------------------|-----------|-----------------------------------|-----------------|---------------------------------|
| MyAirflowEnvironment | Available | Jan 28, 2021 09:49:06 (UTC-07:00) | 1.10.12         | <a href="#">Open Airflow UI</a> |

## Bei Apache Airflow anmelden

Sie benötigen [Zugriffsrichtlinie für die Apache Airflow-Benutzeroberfläche: AmazonMWAA Access WebServer](#) Berechtigungen für Ihr AWS Konto in AWS Identity and Access Management (IAM), um Ihre Apache Airflow-Benutzeroberfläche anzeigen zu können.

Um auf Ihre Apache Airflow-Benutzeroberfläche zuzugreifen

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie „Airflow-Benutzeroberfläche öffnen“.

## Erstellen Sie ein Apache Airflow-Webserver-Zugriffstoken

Sie können die Befehle auf dieser Seite verwenden, um ein Webserver-Zugriffstoken zu erstellen. Ein Zugriffstoken ermöglicht Ihnen den Zugriff auf Ihre Amazon MWAA-Umgebung. Sie können beispielsweise ein Token abrufen und dann DAGs programmgesteuert mithilfe von Amazon MWAA-



APIs bereitstellen. Der folgende Abschnitt enthält die Schritte zum Erstellen eines Apache Airflow-Web-Anmeldetokens mithilfe eines Bash-Skripts, einer POST-API-Anfrage oder eines Python-Skripts. AWS CLI Das in der Antwort zurückgegebene Token ist 60 Sekunden gültig.

## Inhalt

- [Voraussetzungen](#)
  - [Zugriff](#)
  - [AWS CLI](#)
- [Mit dem AWS CLI](#)
- [Verwenden eines Bash-Skripts](#)
- [Verwenden einer POST-API-Anfrage](#)
- [Verwenden eines Python-Skripts](#)
- [Als nächstes](#)

## Voraussetzungen

Im folgenden Abschnitt werden die vorbereitenden Schritte beschrieben, die zur Verwendung der Befehle und Skripts auf dieser Seite erforderlich sind.

### Zugriff

- AWS Kontozugriff in AWS Identity and Access Management (IAM) auf die Amazon MWAA-Berechtigungsrichtlinie in. [Zugriffsrichtlinie für die Apache Airflow-Benutzeroberfläche: AmazonMWAA Access WebServer](#)
- AWS Kontozugriff in AWS Identity and Access Management (IAM) auf die Amazon MWAA-Berechtigungsrichtlinie. [Vollständige API- und Konsolenzugriffsrichtlinie: AmazonMWAA FullApi Access](#)

### AWS CLI

Das AWS Command Line Interface (AWS CLI) ist ein Open-Source-Tool, mit dem Sie mithilfe von Befehlen in Ihrer AWS Befehlszeilen-Shell mit Diensten interagieren können. Um die Schritte auf dieser Seite abzuschließen, benötigen Sie Folgendes:

- [AWS CLI — Installieren Sie Version 2.](#)

- [AWS CLI — Schnelle Konfiguration mit `aws configure`](#).

## Mit dem AWS CLI

Im folgenden Beispiel wird der [create-web-login-token](#) Befehl verwendet, um ein Apache Airflow-Web-Anmeldetoken zu erstellen.

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

## Verwenden eines Bash-Skripts

Im folgenden Beispiel wird ein Bash-Skript verwendet, um den [create-web-login-token](#) Befehl AWS CLI zum Erstellen eines Apache Airflow-Web-Anmeldetokens aufzurufen.

1. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `get-web-token.sh`

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

2. Ersetzen Sie `YOUR_HOST_NAME` und `YOUR_ENVIRONMENT_NAME` durch die roten Platzhalter. Ein Hostname für ein öffentliches Netzwerk könnte beispielsweise so aussehen (ohne `https://`):

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

3. (optional) MacOS- und Linux-Benutzer müssen möglicherweise den folgenden Befehl ausführen, um sicherzustellen, dass das Skript ausführbar ist.

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

4. Führen Sie das folgende Skript aus, um ein Web-Login-Token zu erhalten.

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

5. In Ihrer Eingabeaufforderung sollte Folgendes angezeigt werden:

```
https://123456a0-0101-2020-9e11-1b159eec9000.c2.us-east-1.airflow.amazonaws.com/
aws_mwaa/aws-console-ss0?login=true#{your-web-login-token}
```

## Verwenden einer POST-API-Anfrage

Im folgenden Beispiel wird eine POST-API-Anfrage verwendet, um ein Apache Airflow-Web-Login-Token zu erstellen.

1. Kopieren Sie die folgende URL und fügen Sie sie in das URL-Feld Ihres REST-API-Clients ein.

```
https://YOUR_HOST_NAME/aws_mwaa/aws-console-ss0?login=true#WebToken
```

2. Ersetzen Sie die *roten* Platzhalter durch. YOUR\_HOST\_NAME Ein Hostname für ein öffentliches Netzwerk könnte beispielsweise so aussehen (ohne https://):

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

3. Kopieren Sie den folgenden JSON-Code und fügen Sie ihn in das Hauptfeld Ihres REST-API-Clients ein.

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

4. Ersetzen Sie den *roten* Platzhalter durch. YOUR\_ENVIRONMENT\_NAME
5. Fügen Sie Schlüssel-Wert-Paare in das Autorisierungsfeld ein. Wenn Sie beispielsweise Postman verwenden, wählen Sie AWS Signature und geben Sie dann Folgendes ein:

- AWS\_ACCESS\_KEY\_ID in AccessKey
- AWS\_SECRET\_ACCESS\_KEY in SecretKey

6. Sie sollten die folgende Antwort sehen:

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

## Verwenden eines Python-Skripts

Im folgenden Beispiel wird die Methode [boto3 create\\_web\\_login\\_token in einem Python-Skript verwendet, um ein Apache Airflow-Web-Login-Token](#) zu erstellen. Sie können dieses Skript außerhalb von Amazon MWAA ausführen. Sie müssen lediglich die boto3-Bibliothek installieren. Möglicherweise möchten Sie eine virtuelle Umgebung erstellen, um die Bibliothek zu installieren. Es wird davon ausgegangen, dass Sie die [AWS Authentifizierungsdaten für Ihr Konto konfiguriert](#) haben.

1. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `create-web-login-token.py`.

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

2. Ersetzen Sie den *roten* Platzhalter durch `YOUR_ENVIRONMENT_NAME`
3. Führen Sie das folgende Skript aus, um ein Web-Login-Token zu erhalten.

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

## Als nächstes

- Erkunden Sie den Amazon MWAA-API-Vorgang, mit dem ein Web-Anmeldetoken erstellt wurde, unter [CreateWebLoginToken](#)

## Ein Apache Airflow CLI-Token erstellen

Sie können die Befehle auf dieser Seite verwenden, um ein CLI-Token zu generieren und dann Amazon Managed Workflows for Apache Airflow API-Aufrufe direkt in Ihrer Befehlsshell durchzuführen. Sie können beispielsweise ein Token abrufen und dann DAGs mithilfe von Amazon

MWAA-APIs programmgesteuert bereitstellen. Der folgende Abschnitt enthält die Schritte zum Erstellen eines Apache Airflow-CLI-Tokens mithilfe des AWS CLI, eines Curl-Skripts, eines Python-Skripts oder eines Bash-Skripts. Das in der Antwort zurückgegebene Token ist 60 Sekunden lang gültig.

### Note

Das AWS CLI Token ist als Ersatz für synchrone Shell-Aktionen gedacht, nicht für asynchrone API-Befehle. Daher ist die verfügbare Parallelität begrenzt. Um sicherzustellen, dass der Webserver für Benutzer weiterhin reagiert, wird empfohlen, keine neue AWS CLI Anfrage zu öffnen, bis die vorherige erfolgreich abgeschlossen wurde.

## Inhalt

- [Voraussetzungen](#)
  - [Zugriff](#)
  - [AWS CLI](#)
- [Verwendung der AWS CLI](#)
- [Verwenden eines Curl-Skripts](#)
- [Ein Bash-Skript verwenden](#)
- [Verwenden Sie ein Python-Skript](#)
- [Als nächstes](#)

## Voraussetzungen

Im folgenden Abschnitt werden die vorbereitenden Schritte beschrieben, die für die Verwendung der Befehle und Skripts auf dieser Seite erforderlich sind.

### Zugriff

- AWS-Kontozugriff in AWS Identity and Access Management (IAM) zur Amazon MWAA-Berechtigungsrichtlinie in [Zugriffsrichtlinie für die Apache Airflow-Benutzeroberfläche: AmazonMWAA Access WebServer](#).
- AWS-Kontozugriff in AWS Identity and Access Management (IAM) gemäß der Amazon MWAA-Berechtigungsrichtlinie [Vollständige API- und Konsolenzugriffsrichtlinie: AmazonMWAA FullApi Access](#).

## AWS CLI

Die AWS Command Line Interface (AWS CLI) ist ein Open-Source-Tool, mit dem Sie über Befehle in Ihrer Befehlszeilen-Shell mit den AWS-Services interagieren können. Um die Schritte auf dieser Seite abzuschließen, benötigen Sie Folgendes:

- [AWS CLI— Installieren Sie Sie Version](#)
- [AWS CLI— Schnelle Konfiguration mit `aws configure`](#).

## Verwendung der AWS CLI

Im folgenden Beispiel wird der [create-cli-token](#) Befehl in der verwendet AWS CLI, um ein Apache Airflow CLI-Token zu erstellen.

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

## Verwenden eines Curl-Skripts

Im folgenden Beispiel wird ein Curl-Skript verwendet, um den [create-web-login-token](#) Befehl in der aufzurufen, AWS CLI um die Apache Airflow CLI über einen Endpunkt auf dem Apache Airflow-Webserver aufzurufen.

### Apache Airflow v2

1. Kopieren Sie die Curl-Anweisung aus Ihrer Textdatei und fügen Sie sie in Ihre Befehlsshell ein.

#### Note

Nachdem Sie es in Ihre Zwischenablage kopiert haben, müssen Sie möglicherweise Bearbeiten > Einfügen aus Ihrem Shell-Menü verwenden.

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

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

- Ersetzen Sie die Platzhalter für `YOUR_REGION` durch die AWS Region für Ihre Umgebung `YOUR_DAG_NAME`, und `YOUR_ENVIRONMENT_NAME`. Ein Hostname für ein öffentliches Netzwerk kann beispielsweise so aussehen (ohne `https://`):

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

- Sie sollten Sie Folgendes in Ihrer Eingabeaufforderung sehen:

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

## Apache Airflow v1

- Kopieren Sie die cURL-Anweisung aus Ihrer Textdatei und fügen Sie sie in Ihre Befehlsshell ein.

### Note

Nachdem Sie es in Ihre Zwischenablage kopiert haben, müssen Sie möglicherweise Bearbeiten > Einfügen aus Ihrem Shell-Menü verwenden.

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

```
--data-raw "trigger_dag YOUR_DAG_NAME") \
&& echo "Output:" \
&& echo $CLI_RESULTS | jq -r '.stdout' | base64 --decode \
&& echo "Errors:" \
&& echo $CLI_RESULTS | jq -r '.stderr' | base64 --decode
```

- Ersetzen Sie die Platzhalter für `YOUR_REGION` durch die AWS Region für Ihre Umgebung `YOUR_DAG_NAME`, und `YOUR_HOST_NAME`. Ein Hostname für ein öffentliches Netzwerk kann beispielsweise so aussehen (ohne `https://`):

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

- Sie sollten Sie Folgendes in Ihrer Eingabeaufforderung sehen:

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

- Ersetzen Sie die Platzhalter durch `YOUR_ENVIRONMENT_NAME` und `YOUR_DAG_NAME`.

## Ein Bash-Skript verwenden

Im folgenden Beispiel wird ein Bash-Skript verwendet, um den [create-cli-token](#) Befehl in der aufzurufen AWS CLI, um ein Apache Airflow-CLI-Token zu erstellen.

### Apache Airflow v2

- Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie es lokal unter `get-cli-token.sh`.

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



- Ersetzen Sie die *roten* Platzhalter durch `YOUR_ENVIRONMENT_NAME`, `YOUR_HOST_NAME`, und `YOUR_DAG_NAME`. Ein Hostname für ein öffentliches Netzwerk kann beispielsweise so aussehen (ohne `https://`):

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

- (optional) macOS- und Linux-Benutzer müssen möglicherweise den folgenden Befehl ausführen, um sicherzustellen, dass das Skript ausführbar ist.

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

- Führen Sie das folgende Skript aus, um ein Apache Airflow CLI-Token zu erstellen.

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

## Apache Airflow v1

- Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie es lokal unter `get-cli-token.sh`.

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

- Ersetzen Sie die *roten* Platzhalter durch `YOUR_ENVIRONMENT_NAME`, `YOUR_HOST_NAME`, und `YOUR_DAG_NAME`. Ein Hostname für ein öffentliches Netzwerk kann beispielsweise so aussehen (ohne `https://`):

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

- (optional) macOS- und Linux-Benutzer müssen möglicherweise den folgenden Befehl ausführen, um sicherzustellen, dass das Skript ausführbar ist.

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

- Führen Sie das folgende Skript aus, um ein Apache Airflow CLI-Token zu erstellen.

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

## Verwenden Sie ein Python-Skript

Im folgenden Beispiel wird die [boto3-Methode `create\_cli\_token`](#) in einem Python-Skript verwendet, um ein Apache Airflow-CLI-Token zu erstellen und eine DAG auszulösen. Sie können dieses Skript außerhalb von Amazon MWAA ausführen. Sie müssen nur noch die boto3-Bibliothek installieren. Möglicherweise möchten Sie eine virtuelle Umgebung erstellen, um die Bibliothek zu installieren. Es wird davon ausgegangen, dass Sie die [AWSAuthentifizierungsdaten für Ihr Konto konfiguriert](#) haben.

### Apache Airflow v2

1. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie es lokal unter `create-cli-token.py`.

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

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

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

import boto3
import json
import requests
import base64

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

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

mwaas_cli_token = client.create_cli_token(
 Name=mwaas_env_name
)

mwaas_auth_token = 'Bearer ' + mwaas_cli_token['CliToken']
mwaas_webserver_hostname = 'https://{0}/aws_mwaas/
cli'.format(mwaas_cli_token['WebServerHostname'])
raw_data = '{0} {1}'.format(mwaas_cli_command, dag_name)

mwaas_response = requests.post(
 mwaas_webserver_hostname,
 headers={
 'Authorization': mwaas_auth_token,
 'Content-Type': 'text/plain'
 },
 data=raw_data
)

mwaas_std_err_message = base64.b64decode(mwaas_response.json()
['stderr']).decode('utf8')
mwaas_std_out_message = base64.b64decode(mwaas_response.json()
['stdout']).decode('utf8')

print(mwaas_response.status_code)
print(mwaas_std_err_message)
print(mwaas_std_out_message)
```

2. Ersetzen Sie die Platzhalter durch `YOUR_ENVIRONMENT_NAME` und `YOUR_DAG_NAME`.
3. Führen Sie das folgende Skript aus, um ein Apache Airflow CLI-Token zu erstellen.

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

## Apache Airflow v1

1. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie es lokal unter `create-cli-token.py`.

```
import boto3
import json
```

```
import requests
import base64

mwa_env_name = 'YOUR_ENVIRONMENT_NAME'
dag_name = 'YOUR_DAG_NAME'
mwa_cli_command = 'trigger_dag'

client = boto3.client('mwa')

mwa_cli_token = client.create_cli_token(
 Name=mwa_env_name
)

mwa_auth_token = 'Bearer ' + mwa_cli_token['CliToken']
mwa_webserver_hostname = 'https://{0}/aws_mwa/
cli'.format(mwa_cli_token['WebServerHostname'])
raw_data = '{0} {1}'.format(mwa_cli_command, dag_name)

mwa_response = requests.post(
 mwa_webserver_hostname,
 headers={
 'Authorization': mwa_auth_token,
 'Content-Type': 'text/plain'
 },
 data=raw_data
)

mwa_std_err_message = base64.b64decode(mwa_response.json()
['stderr']).decode('utf8')
mwa_std_out_message = base64.b64decode(mwa_response.json()
['stdout']).decode('utf8')

print(mwa_response.status_code)
print(mwa_std_err_message)
print(mwa_std_out_message)
```

2. Ersetzen Sie die Platzhalter durch `YOUR_ENVIRONMENT_NAME` und `YOUR_DAG_NAME`.
3. Führen Sie das folgende Skript aus, um ein Apache Airflow CLI-Token zu erstellen.

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

## Als nächstes

- Informieren Sie sich über den Amazon MWAA-API-Vorgang, mit dem ein CLI-Token erstellt wurde, unter [CreateCliToken](#).

## Verwenden der Apache Airflow REST API

Amazon Managed Workflows for Apache Airflow (Amazon MWAA) unterstützt die direkte Interaktion mit Ihren Apache Airflow-Umgebungen mithilfe der Apache Airflow-REST-API für Umgebungen, in denen Apache Airflow v2.4.3 und höher ausgeführt wird. Auf diese Weise können Sie programmgesteuert auf Ihre Amazon MWAA-Umgebungen zugreifen und diese verwalten. Dies bietet eine standardisierte Möglichkeit, Datenorchestrierungs-Workflows aufzurufen, Ihre DAGs zu verwalten und den Status verschiedener Apache Airflow-Komponenten wie der Metadatendatenbank, des Triggers und des Schedulers zu überwachen.

Um die direkte Verwendung der Apache Airflow-REST-API zu unterstützen, bietet Ihnen Amazon MWAA die Möglichkeit, die Webserverkapazität horizontal zu skalieren, um den erhöhten Bedarf zu bewältigen, sei es durch REST-API-Anfragen, die Verwendung der Befehlszeilenschnittstelle (CLI) oder mehr gleichzeitige Benutzer der Apache Airflow-Benutzeroberfläche (UI). Weitere Informationen zur Skalierung von Webservern durch Amazon MWAA finden Sie unter [the section called "Automatische Skalierung von Webservern konfigurieren"](#)

Sie können die Apache Airflow REST API verwenden, um die folgenden Anwendungsfälle für Ihre Umgebungen zu implementieren:

- **Programmatischer Zugriff** — Sie können jetzt Apache Airflow DAG-Läufe starten, Datensätze verwalten und den Status verschiedener Komponenten wie der Metadatendatenbank, Trigger und Scheduler abrufen, ohne sich auf die Apache Airflow-Benutzeroberfläche oder CLI verlassen zu müssen.
- **Integration mit externen Anwendungen und Microservices** — Die REST-API-Unterstützung ermöglicht es Ihnen, maßgeschneiderte Lösungen zu entwickeln, die Ihre Amazon MWAA-Umgebungen in andere Systeme integrieren. Sie können Workflows beispielsweise als Reaktion auf Ereignisse aus externen Systemen starten, wie z. B. abgeschlossene Datenbankaufträge oder neue Benutzeranmeldungen.
- **Zentralisierte Überwachung** — Sie können Überwachungs-Dashboards erstellen, die den Status Ihrer DAGs in mehreren Amazon MWAA-Umgebungen zusammenfassen und so eine zentrale Überwachung und Verwaltung ermöglichen.

Die folgenden Themen zeigen, wie Sie ein Webserver-Zugriffstoken erhalten und dieses Token dann verwenden, um API-Aufrufe an die Apache Airflow-REST-API zu tätigen. Im folgenden Beispiel rufen Sie die API auf, um einen neuen DAG-Lauf zu starten.

Weitere Informationen zur Apache Airflow REST API finden Sie in [der Apache Airflow REST API-Referenz](#).

## Themen

- [Erstellen Sie ein Webserver-Sitzungstoken](#)
- [Rufen Sie die Apache Airflow REST API auf](#)

## Erstellen Sie ein Webserver-Sitzungstoken

Verwenden Sie die folgende Python-Funktion, um ein Webserver-Zugriffstoken zu erstellen. Diese Funktion ruft zunächst die Amazon MWAA-API auf, um ein Web-Login-Token zu erhalten. Das Web-Login-Token, das nach 60 Sekunden abläuft, wird dann gegen ein Web-Session-Token ausgetauscht, mit dem Sie auf den Webserver zugreifen und die Apache Airflow-REST-API verwenden können.

### Note

Das Sitzungstoken läuft nach 12 Stunden ab.

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

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

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

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

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

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

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

## Rufen Sie die Apache Airflow REST API auf

Sobald die Authentifizierung abgeschlossen ist, verfügen Sie über die Anmeldeinformationen, um mit dem Senden von Anfragen an die API-Endpunkte zu beginnen. Verwenden Sie im folgenden Beispiel den Endpunkt/dags/*dag\_id*/dag.

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

 Args:
 region (str): AWS region where the MAAA environment is hosted.
 env_name (str): Name of the MAAA environment.
```

```
dag_name (str): Name of the DAG to trigger.
"""

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

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

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

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

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

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

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



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

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

## Befehlsreferenz für Apache Airflow CLI

Auf dieser Seite werden die unterstützten und nicht unterstützten Apache Airflow CLI-Befehle in Amazon Managed Workflows for Apache Airflow beschrieben.

### Inhalt

- [Voraussetzungen](#)
  - [Zugriff](#)
  - [AWS CLI](#)
- [Was hat sich in Version 2 geändert](#)
- [Unterstützte CLI-Befehle](#)
  - [Unterstützte Befehle](#)
  - [Verwenden von Befehlen, die DAGs analysieren](#)
- [Beispiel-Code](#)
  - [Eine Apache Airflow v2-Variable festlegen, abrufen oder löschen](#)
  - [Fügen Sie beim Auslösen einer DAG eine Konfiguration hinzu](#)
  - [Führen Sie CLI-Befehle in einem SSH-Tunnel zu einem Bastion-Host aus](#)
  - [Beispiele GitHub und Tutorials AWS](#)

## Voraussetzungen

Im folgenden Abschnitt werden die vorbereitenden Schritte beschrieben, die zur Verwendung der Befehle und Skripts auf dieser Seite erforderlich sind.

## Zugriff

- AWS Kontozugriff in AWS Identity and Access Management (IAM) auf die Amazon MWAA-Berechtigungsrichtlinie in. [Zugriffsrichtlinie für die Apache Airflow-Benutzeroberfläche: AmazonMWAA Access WebServer](#)
- AWS Kontozugriff in AWS Identity and Access Management (IAM) auf die Amazon MWAA-Berechtigungsrichtlinie. [Vollständige API- und Konsolenzugriffsrichtlinie: AmazonMWAA FullApi Access](#)

## AWS CLI

Das AWS Command Line Interface (AWS CLI) ist ein Open-Source-Tool, mit dem Sie mithilfe von Befehlen in Ihrer AWS Befehlszeilen-Shell mit Diensten interagieren können. Um die Schritte auf dieser Seite abzuschließen, benötigen Sie Folgendes:

- [AWS CLI — Installieren Sie Version 2.](#)
- [AWS CLI — Schnelle Konfiguration mit `aws configure`.](#)

## Was hat sich in Version 2 geändert

- Neu: Airflow CLI Befehlsstruktur. Die Apache Airflow v2-CLI ist so organisiert, dass verwandte Befehle als Unterbefehle gruppiert sind. Das bedeutet, dass Sie Apache Airflow v1-Skripte aktualisieren müssen, wenn Sie auf Apache Airflow v2 aktualisieren möchten. In Apache Airflow ist Version 1 beispielsweise jetzt `unpause dags unpause` in Apache Airflow v2 enthalten. Weitere Informationen finden Sie unter [Airflow CLI Changes in 2](#) im Apache Airflow-Referenzhandbuch.

## Unterstützte CLI-Befehle

Im folgenden Abschnitt sind die Apache Airflow CLI-Befehle aufgeführt, die auf Amazon MWAA verfügbar sind.

## Unterstützte Befehle

### Apache Airflow v2

| Nebenversionen                  | Befehl                                         |  |
|---------------------------------|------------------------------------------------|--|
| Version 2.0 und höher           | <a href="#">Spickzettel</a>                    |  |
| Version 2.0 oder höher          | <a href="#">Verbindungen hinzufügen</a>        |  |
| v2.0+                           | <a href="#">Verbindungen löschen</a>           |  |
| <a href="#">v2.2+ (Hinweis)</a> | <a href="#">Tage Verfüllung</a>                |  |
| Version 2.0 und höher           | <a href="#">Tage löschen</a>                   |  |
| <a href="#">v2.2+ (Hinweis)</a> | <a href="#">Liste der Tage</a>                 |  |
| Version 2.0 oder höher          | <a href="#">Tagesliste — Jobs</a>              |  |
| Version 2.6 und höher           | <a href="#">Tage list-import-errors</a>        |  |
| <a href="#">v2.2+ (Hinweis)</a> | <a href="#">Die Tagesliste wird ausgeführt</a> |  |
| <a href="#">v2.2+ (Hinweis)</a> | <a href="#">Tage nächste Ausführung</a>        |  |
| Version 2.0 und höher           | <a href="#">2 Tage Pause</a>                   |  |
| Version 2.0 und höher           | <a href="#">3 Tage Bericht</a>                 |  |
| Version 2.4 oder höher          | <a href="#">Tage reserialisieren</a>           |  |
| Version 2.0 und höher           | <a href="#">Tage zeigen</a>                    |  |
| Version 2.0 oder höher          | <a href="#">Zustand der Tage</a>               |  |
| Version 2.0 und höher           | <a href="#">Tage Test</a>                      |  |
| Version 2.0 und höher           | <a href="#">Tage auslösen</a>                  |  |
| Version 2.0 und höher           | <a href="#">Tage Pause</a>                     |  |

| Nebenversionen         | Befehl                                           |  |
|------------------------|--------------------------------------------------|--|
| Version 2.4 oder höher | <a href="#">db sauber</a>                        |  |
| Version 2.0 oder höher | <a href="#">Verhalten der Anbieter</a>           |  |
| Version 2.0 und höher  | <a href="#">Anbieter erhalten</a>                |  |
| v2.0+                  | <a href="#">Hooks von Anbietern</a>              |  |
| Version 2.0 und höher  | <a href="#">Links zu Anbietern</a>               |  |
| Version 2.0 oder höher | <a href="#">Liste der Anbieter</a>               |  |
| v2.8+                  | <a href="#">Benachrichtigungen von Anbietern</a> |  |
| Version 2.6 oder höher | <a href="#">Geheimnisse der Anbieter</a>         |  |
| Version 2.7 oder höher | <a href="#">Trigger durch Anbieter</a>           |  |
| Version 2.0 und höher  | <a href="#">Widgets für Anbieter</a>             |  |
| Version 2.6 oder höher | <a href="#">Rollen: Hinzufügen von Perms</a>     |  |
| Version 2.6 oder höher | <a href="#">Rollen del-perms</a>                 |  |
| Version 2.6 oder höher | <a href="#">Rollen erstellen</a>                 |  |
| Version 2.0 oder höher | <a href="#">Liste der Rollen</a>                 |  |
| Version 2.0 oder höher | <a href="#">Aufgaben geklärt</a>                 |  |
| v2.0+                  | <a href="#">Aufgaben fehlgeschlagen-Deps</a>     |  |
| Version 2.0 und höher  | <a href="#">Liste der Aufgaben</a>               |  |
| v2.0+                  | <a href="#">Aufgaben rendern</a>                 |  |
| Version 2.0 oder höher | <a href="#">Status der Aufgaben</a>              |  |

| Nebenversionen         | Befehl                                      |  |
|------------------------|---------------------------------------------|--|
| Version 2.0 oder höher | <a href="#">Aufgaben states-for-dag-run</a> |  |
| v2.0+                  | <a href="#">Aufgaben testen</a>             |  |
| v2.0+                  | <a href="#">Variablen löschen</a>           |  |
| v2.0+                  | <a href="#">Variablen erhalten</a>          |  |
| Version 2.0 oder höher | <a href="#">Variablen gesetzt</a>           |  |
| Version 2.0 oder höher | <a href="#">Liste der Variablen</a>         |  |
| v2.0+                  | <a href="#">Version</a>                     |  |

## Verwenden von Befehlen, die DAGs analysieren

Wenn in Ihrer Umgebung Apache Airflow v1.10.12 oder v2.0.2 ausgeführt wird, schlagen CLI-Befehle zum Parsen von DAGs fehl, wenn die DAG Plugins verwendet, die von Paketen abhängen, die über Folgendes installiert wurden: `requirements.txt`

### Apache Airflow v2.0.2

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

Sie können diese CLI-Befehle verwenden, wenn Ihre DAGs keine Plugins verwenden, die von Paketen abhängen, die über `a requirements.txt` installiert wurden.

## Beispiel-Code

Der folgende Abschnitt enthält Beispiele für verschiedene Verwendungsmöglichkeiten der Apache Airflow CLI.

## Eine Apache Airflow v2-Variable festlegen, abrufen oder löschen

Sie können den folgenden Beispielcode verwenden, um eine Variable im Format von `<script><mwaa env name> get | set | delete <variable> <variable value> </variable></variable>` festzulegen, abzurufen oder zu löschen.

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

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

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

else
 echo "Not a valid command"
 exit 1
fi

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

## Fügen Sie beim Auslösen einer DAG eine Konfiguration hinzu

Sie können den folgenden Beispielcode mit Apache Airflow v1 und Apache Airflow v2 verwenden, um beim Auslösen einer DAG eine Konfiguration hinzuzufügen, z. `airflow trigger_dag 'dag_name' -conf '{"key":"value"}`

```
import boto3
import json
import requests
import base64
```

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

client = boto3.client('mwa')

mwa_cli_token = client.create_cli_token(
 Name=mwa_env_name
)

mwa_auth_token = 'Bearer ' + mwa_cli_token['CliToken']
mwa_webserver_hostname = 'https://{0}/aws_mwa/
cli'.format(mwa_cli_token['WebServerHostname'])
raw_data = "trigger_dag {0} -c '{1}'".format(dag_name, conf)

mwa_response = requests.post(
 mwa_webserver_hostname,
 headers={
 'Authorization': mwa_auth_token,
 'Content-Type': 'text/plain'
 },
 data=raw_data
)

mwa_std_err_message = base64.b64decode(mwa_response.json()['stderr']).decode('utf8')
mwa_std_out_message = base64.b64decode(mwa_response.json()['stdout']).decode('utf8')

print(mwa_response.status_code)
print(mwa_std_err_message)
print(mwa_std_out_message)
```

## Führen Sie CLI-Befehle in einem SSH-Tunnel zu einem Bastion-Host aus

Das folgende Beispiel zeigt, wie Airflow CLI-Befehle mithilfe eines SSH-Tunnel-Proxys zu einem Linux-Bastion-Host ausgeführt werden.

Mit curl

1. 

```
ssh -D 8080 -f -C -q -N YOUR_USER@YOUR_BASTION_HOST
```

2. 

```
curl -x socks5h://0:8080 --request POST https://YOUR_HOST_NAME/aws_mwaa/cli --
header YOUR_HEADERS --data-raw YOUR_CLI_COMMAND
```

## Beispiele GitHub und Tutorials AWS

- [Arbeiten mit Parametern und Variablen von Apache Airflow v2.0.2 in Amazon Managed Workflows for Apache Airflow](#)
- [Interaktion mit Apache Airflow v1.10.12 auf Amazon MWAA über die Befehlszeile](#)
- [Interaktive Befehle mit Apache Airflow v1.10.12 auf Amazon MWAA](#) und Bash Operator auf GitHub



# Verbindungen zu Apache Airflow verwalten

In diesem Abschnitt werden die verschiedenen Möglichkeiten zur Konfiguration einer Apache-Airflow for-Apache-Airflow for-Apache-Airflow for-Apache-Airflow beschrieben.

## Themen

- [Überblick über Apache Airflow-Variablen und Verbindungen](#)
- [In Amazon MWAA-Umgebungen installierte Apache Airflow-Anbieterpakete](#)
- [Übersicht über Verbindungsarten](#)
- [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#)

## Überblick über Apache Airflow-Variablen und Verbindungen

In einigen Fällen möchten Sie möglicherweise zusätzliche Verbindungen oder Variablen für eine Umgebung angeben, z. B. ein AWS Profil, oder Ihre Ausführungsrolle in einem Verbindungsobjekt im Apache Airflow-Metastore hinzufügen und dann auf die Verbindung innerhalb einer DAG verweisen.

- **Selbstverwalteter Apache Airflow.** Bei einer selbstverwalteten Apache Airflow-Installation legen Sie die [Apache Airflow-Konfigurationsoptionen in](#) `festairflow.cfg`.

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

- **Apache Airflow auf Amazon MWAA.** Bei Amazon MWAA müssen Sie diese Konfigurationseinstellungen als [Apache Airflow-Konfigurationsoptionen](#) auf der Amazon MWAA-Konsole hinzufügen. Die Apache Airflow-Konfigurationsoptionen werden als Umgebungsvariablen in Ihre Umgebung geschrieben und überschreiben alle anderen vorhandenen Konfigurationen für dieselbe Einstellung.

# In Amazon MWAA-Umgebungen installierte Apache Airflow-Anbieterpakete

Amazon MWAA installiert [Anbieter-Extras](#) für Apache Airflow v2 und höhere Verbindungstypen, wenn Sie eine neue Umgebung erstellen. Durch die Installation von Anbieterpaketen können Sie einen Verbindungstyp in der Apache Airflow-Benutzeroberfläche anzeigen. Dies bedeutet auch, dass Sie diese Pakete nicht als Python-Abhängigkeit in Ihrer `-requirements.txt` Datei angeben müssen. Auf dieser Seite finden Sie die von Amazon MWAA installierten Apache-Airflow-Anbieterpakete für alle Apache-Airflow-v2-Umgebungen.

## Note

Für Apache Airflow v2 und höher installiert Amazon MWAA [Watchtower Version 2.0.1](#) nach der Ausführung von `pip3 install -r requirements.txt`, dass die Kompatibilität mit der CloudWatch Protokollierung nicht durch andere Python-Bibliotheksinstallationen überschrieben wird.

## Inhalt

- [Anbieterpakete für Apache Airflow v2.8.1-Verbindungen](#)
- [Anbieterpakete für Apache Airflow v2.7.2-Verbindungen](#)
- [Anbieterpakete für Apache Airflow v2.6.3-Verbindungen](#)
- [Anbieterpakete für Apache Airflow v2.5.1-Verbindungen](#)
- [Anbieterpakete für Apache Airflow v2.4.3-Verbindungen](#)
- [Anbieterpakete für Apache Airflow v2.2.2-Verbindungen](#)
- [Anbieterpakete für Apache Airflow v2.0.2-Verbindungen](#)
- [Angaben neuerer Anbieterpakete](#)

## Anbieterpakete für Apache Airflow v2.8.1-Verbindungen

Wenn Sie eine Amazon MWAA-Umgebung in Apache Airflow v2.8.1 erstellen, installiert Amazon MWAA die folgenden Anbieterpakete, die für Apache Airflow-Verbindungen verwendet werden.

**Note**

Sie können die neueste unterstützte Version von `apache-airflow-providers-amazon`, um diesen Anbieter zu aktualisieren. Weitere Informationen zur Angabe neuerer Versionen finden Sie unter [the section called “Angeben neuerer Anbieterpakete”](#).

| Verbindungstyp        | Paket                                                               |
|-----------------------|---------------------------------------------------------------------|
| AWS Verbindung        | <a href="#">apache-airflow-providers-amazon[aiobotocore]=8.16.0</a> |
| Postgres-Verbindung   | <a href="#">apache-airflow-providers-postgres==5.10.0</a>           |
| FTP-Verbindung        | <a href="#">apache-airflow-providers-ftp==3.7.0</a>                 |
| Veredelungsverbindung | <a href="#">apache-airflow-providers-celery==3.5.1</a>              |
| HTTP-Verbindung       | <a href="#">apache-airflow-providers-http==4.8.0</a>                |
| IMAP-Verbindung       | <a href="#">apache-airflow-providers-imap==3.5.0</a>                |
| Allgemeines SQL       | <a href="#">apache-airflow-providers-common-sql==1.10.0</a>         |
| SQLite-Verbindung     | <a href="#">apache-airflow-providers-sqlite==3.7.0</a>              |

## Anbieterpakete für Apache Airflow v2.7.2-Verbindungen

Wenn Sie eine Amazon MWAA-Umgebung in Apache Airflow v2.7.2 erstellen, installiert Amazon MWAA die folgenden Anbieterpakete, die für Apache Airflow-Verbindungen verwendet werden.

**Note**

Sie können die neueste unterstützte Version von `apache-airflow-providers-amazon`, um diesen Anbieter zu aktualisieren. Weitere Informationen zur Angabe neuerer Versionen finden Sie unter [the section called “Angeben neuerer Anbieterpakete”](#).

| Verbindungstyp        | Paket                                                              |
|-----------------------|--------------------------------------------------------------------|
| AWS Verbindung        | <a href="#">apache-airflow-providers-amazon[aiobotocore]=8.7.1</a> |
| Postgres-Verbindung   | <a href="#">apache-airflow-providers-postgres==5.6.1</a>           |
| FTP-Verbindung        | <a href="#">apache-airflow-providers-ftp==3.5.2</a>                |
| Veredelungsverbindung | <a href="#">apache-airflow-providers-celery==3.3.4</a>             |
| HTTP-Verbindung       | <a href="#">apache-airflow-providers-http==4.5.2</a>               |
| IMAP-Verbindung       | <a href="#">apache-airflow-providers-imap==3.3.2</a>               |
| Allgemeines SQL       | <a href="#">apache-airflow-providers-common-sql==1.7.2</a>         |
| SQLite-Verbindung     | <a href="#">apache-airflow-providers-sqlite==3.4.3</a>             |

## Anbieterpakete für Apache Airflow v2.6.3-Verbindungen

Wenn Sie eine Amazon MWAA-Umgebung in Apache Airflow v2.6.3 erstellen, installiert Amazon MWAA die folgenden Anbieterpakete, die für Apache Airflow-Verbindungen verwendet werden.

### Note

Sie können die neueste unterstützte Version von `apache-airflow-providers-amazon`, um diesen Anbieter zu aktualisieren. Weitere Informationen zur Angabe neuerer Versionen finden Sie unter [the section called “Angaben neuerer Anbieterpakete”](#).

| Verbindungstyp      | Paket                                                              |
|---------------------|--------------------------------------------------------------------|
| AWS Verbindung      | <a href="#">apache-airflow-providers-amazon[aiobotocore]=8.2.0</a> |
| Postgres-Verbindung | <a href="#">apache-airflow-providers-postgres==5.5.1</a>           |

| Verbindungstyp        | Paket                                                      |
|-----------------------|------------------------------------------------------------|
| FTP-Verbindung        | <a href="#">apache-airflow-providers-ftp==3.4.2</a>        |
| Veredelungsverbindung | <a href="#">apache-airflow-providers-celery==3.2.1</a>     |
| HTTP-Verbindung       | <a href="#">apache-airflow-providers-http==4.4.2</a>       |
| IMAP-Verbindung       | <a href="#">apache-airflow-providers-imap==3.2.2</a>       |
| Allgemeines SQL       | <a href="#">apache-airflow-providers-common-sql==1.5.2</a> |
| SQLite-Verbindung     | <a href="#">apache-airflow-providers-sqlite==3.4.2</a>     |

## Anbieterpakete für Apache Airflow v2.5.1-Verbindungen

Wenn Sie eine Amazon MWAA-Umgebung in Apache Airflow v2.5.1 erstellen, installiert Amazon MWAA die folgenden Anbieterpakete, die für Apache Airflow-Verbindungen verwendet werden.

### Note

Sie können die neueste unterstützte Version von `apache-airflow-providers-amazon`, um diesen Anbieter zu aktualisieren. Weitere Informationen zur Angabe neuerer Versionen finden Sie unter [the section called “Angaben neuerer Anbieterpakete”](#).

| Verbindungstyp        | Paket                                                    |
|-----------------------|----------------------------------------------------------|
| AWS Verbindung        | <a href="#">apache-airflow-providers-amazon==7.1.0</a>   |
| Postgres-Verbindung   | <a href="#">apache-airflow-providers-postgres==5.4.0</a> |
| FTP-Verbindung        | <a href="#">apache-airflow-providers-ftp==3.3.0</a>      |
| Veredelungsverbindung | <a href="#">apache-airflow-providers-celery==3.1.0</a>   |
| HTTP-Verbindung       | <a href="#">apache-airflow-providers-http==4.1.1</a>     |
| IMAP-Verbindung       | <a href="#">apache-airflow-providers-imap==3.1.1</a>     |

| Verbindungstyp    | Paket                                                      |
|-------------------|------------------------------------------------------------|
| Allgemeines SQL   | <a href="#">apache-airflow-providers-common-sql==1.3.3</a> |
| SQLite-Verbindung | <a href="#">apache-airflow-providers-sqlite==3.3.1</a>     |

## Anbieterpakete für Apache Airflow v2.4.3-Verbindungen

Wenn Sie eine Amazon MWAA-Umgebung in Apache Airflow v2.4.3 erstellen, installiert Amazon MWAA die folgenden Anbieterpakete, die für Apache Airflow-Verbindungen verwendet werden.

| Verbindungstyp        | Paket                                                      |
|-----------------------|------------------------------------------------------------|
| AWS Verbindung        | <a href="#">apache-airflow-providers-amazon==6.0.0</a>     |
| Postgres-Verbindung   | <a href="#">apache-airflow-providers-postgres==5.2.2</a>   |
| FTP-Verbindung        | <a href="#">apache-airflow-providers-ftp==3.1.0</a>        |
| Veredelungsverbindung | <a href="#">apache-airflow-providers-celery==3.0.0</a>     |
| HTTP-Verbindung       | <a href="#">apache-airflow-providers-http==4.0.0</a>       |
| IMAP-Verbindung       | <a href="#">apache-airflow-providers-imap==3.0.0</a>       |
| Allgemeines SQL       | <a href="#">apache-airflow-providers-common-sql==1.2.0</a> |
| SQLite-Verbindung     | <a href="#">apache-airflow-providers-sqlite==3.2.1</a>     |

## Anbieterpakete für Apache Airflow v2.2.2-Verbindungen

Wenn Sie eine Amazon MWAA-Umgebung in Apache Airflow v2.2.2 erstellen, installiert Amazon MWAA die folgenden Anbieterpakete, die für Apache Airflow-Verbindungen verwendet werden.

| Verbindungstyp | Paket                                                  |
|----------------|--------------------------------------------------------|
| AWS Verbindung | <a href="#">apache-airflow-providers-amazon==2.4.0</a> |

| Verbindungstyp        | Paket                                                    |
|-----------------------|----------------------------------------------------------|
| Postgres-Verbindung   | <a href="#">apache-airflow-providers-postgres==2.3.0</a> |
| FTP-Verbindung        | <a href="#">apache-airflow-providers-ftp==2.0.1</a>      |
| Veredelungsverbindung | <a href="#">apache-airflow-providers-celery==2.1.0</a>   |
| HTTP-Verbindung       | <a href="#">apache-airflow-providers-http==2.0.1</a>     |
| IMAP-Verbindung       | <a href="#">apache-airflow-providers-imap==2.0.1</a>     |
| SQLite-Verbindung     | <a href="#">apache-airflow-providers-sqlite==2.0.1</a>   |

## Anbieterpakete für Apache Airflow v2.0.2-Verbindungen

Wenn Sie eine Amazon MWAA-Umgebung in Apache Airflow v2.0.2 erstellen, installiert Amazon MWAA die folgenden Anbieterpakete, die für Apache Airflow-Verbindungen verwendet werden.

| Verbindungstyp        | Paket                                                      |
|-----------------------|------------------------------------------------------------|
| Tableau-Verbindung    | <a href="#">apache-airflow-providers-tableau==1.0.0</a>    |
| Databricks-Verbindung | <a href="#">apache-airflow-providers-databricks==1.0.1</a> |
| SSH-Verbindung        | <a href="#">apache-airflow-providers-ssh==1.3.0</a>        |
| Postgres-Verbindung   | <a href="#">apache-airflow-providers-postgres==1.0.2</a>   |
| Docker-Verbindung     | <a href="#">apache-airflow-providers-docker==1.2.0</a>     |
| Oracle-Verbindung     | <a href="#">apache-airflow-providers-oracle==1.1.0</a>     |
| Presto-Verbindung     | <a href="#">apache-airflow-providers-presto==1.0.2</a>     |
| SFTP-Verbindung       | <a href="#">apache-airflow-providers-sftp==1.2.0</a>       |

## Angeben neuerer Anbieterpakete

Ab Apache Airflow v2.7.2 muss Ihre Anforderungsdatei eine `--constraint`-Anweisung enthalten. Wenn Sie keine Einschränkung angeben, gibt Amazon MWAA eine für Sie an, um sicherzustellen, dass die in Ihren Anforderungen aufgeführten Pakete mit der von Ihnen verwendeten Version von Apache Airflow kompatibel sind.

Apache Airflow-Einschränkungsdateien geben die Anbieterversionen an, die zum Zeitpunkt einer Apache Airflow-Version verfügbar sind. In vielen Fällen sind neuere Anbieter jedoch mit dieser Version von Apache Airflow kompatibel. Da Sie Einschränkungen verwenden müssen, können Sie die Bedingungsdatei für eine bestimmte Anbieterversion ändern, um eine neuere Version eines Anbieterpakets anzugeben:

1. Laden Sie die versionsspezifische Einschränkungsdtei von <https://raw.githubusercontent.com/apache/airflow/constraints-2.7.2/constraints-3.11.txt> herunter
2. Ändern Sie die `apache-airflow-providers-amazon` Version in der Einschränkungsdtei in die Version, die Sie verwenden möchten.
3. Speichern Sie die geänderte Einschränkungsdtei im Ordner Amazon S3 Dags Ihrer Amazon MWAA-Umgebung, z. B. als `constraints-3.11-updated.txt`
4. Geben Sie Ihre Anforderungen wie im Folgenden gezeigt an.

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

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

### Note

Wenn Sie einen privaten Webserver verwenden, empfehlen wir Ihnen, [die erforderlichen Bibliotheken mithilfe des lokalen Amazon MWAA-Runners als WHL-Dateien zu verpacken](https://github.com/aws/aws-mwaa-local-runner). <https://github.com/aws/aws-mwaa-local-runner>

## Übersicht über Verbindungsarten

Apache Airflow speichert Verbindungen als Verbindungs-URI-Zeichenfolge. Es stellt eine Verbindungsvorlage in der Apache Airflow-Benutzeroberfläche bereit, um die Verbindungs-URI-Zeichenfolge unabhängig vom Verbindungstyp zu generieren. Wenn eine Verbindungsvorlage in der



Apache Airflow-Benutzeroberfläche nicht verfügbar ist, kann eine alternative Verbindungsvorlage verwendet werden, um diese Verbindungs-URI-Zeichenfolge zu generieren, z. B. mithilfe der HTTP-Verbindungsvorlage. Der Hauptunterschied ist das URI-Präfix, z. B. `dasmy-conn-type://`, das Apache Airflow-Anbieter für eine Verbindung normalerweise ignorieren. Auf dieser Seite wird beschrieben, wie Verbindungsvorlagen in der Apache Airflow-Benutzeroberfläche synonym für verschiedene Verbindungstypen verwendet werden.

### Warning

Überschreiben Sie die `aws_default` Verbindung in Amazon MWAA nicht. Amazon MWAA verwendet diese Verbindung, um eine Vielzahl kritischer Aufgaben auszuführen, z. B. das Sammeln von Aufgabenprotokollen. Das Überschreiben dieser Verbindung kann zu Datenverlust und Störungen der Verfügbarkeit Ihrer Umgebung führen.

## Themen

- [Beispiel einer URI-Zeichenfolge für eine Verbindung](#)
- [Beispiel-Verbindungsvorlage](#)
- [Beispiel für die Verwendung einer HTTP-Verbindungsvorlage für eine Jdbc-Verbindung](#)

## Beispiel einer URI-Zeichenfolge für eine Verbindung

Das folgende Beispiel zeigt eine Verbindungs-URI-Zeichenfolge für den MySQL-Verbindungstyp.

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

## Beispiel-Verbindungsvorlage

Das folgende Beispiel zeigt die Vorlage für die HTTP-Verbindung in der Apache Airflow-Benutzeroberfläche.

### Apache Airflow v2

Das folgende Beispiel zeigt die HTTP-Verbindungsvorlage für Apache Airflow v2 in der Apache Airflow-Benutzeroberfläche.

### Add Connection

|                    |                                                                                                                                               |
|--------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Conn Id *</b>   | <input type="text"/>                                                                                                                          |
| <b>Conn Type *</b> | <input type="text" value="HTTP"/><br><small>Conn Type missing? Make sure you've installed the corresponding Airflow Provider Package.</small> |
| <b>Description</b> | <input type="text"/>                                                                                                                          |
| <b>Host</b>        | <input type="text"/>                                                                                                                          |
| <b>Schema</b>      | <input type="text"/>                                                                                                                          |
| <b>Login</b>       | <input type="text"/>                                                                                                                          |
| <b>Password</b>    | <input type="text"/>                                                                                                                          |
| <b>Port</b>        | <input type="text"/>                                                                                                                          |
| <b>Extra</b>       | <input type="text"/>                                                                                                                          |

## Apache Airflow v1

Das folgende Beispiel zeigt die HTTP-Verbindungsvorlage für Apache Airflow v1 in der Apache Airflow-Benutzeroberfläche.

| Add Connection |                                   |
|----------------|-----------------------------------|
| Conn Id *      | <input type="text"/>              |
| Conn Type      | <input type="text" value="HTTP"/> |
| Host           | <input type="text"/>              |
| Schema         | <input type="text"/>              |
| Login          | <input type="text"/>              |
| Password       | <input type="text"/>              |
| Port           | <input type="text"/>              |
| Extra          | <input type="text"/>              |

## Beispiel für die Verwendung einer HTTP-Verbindungsvorlage für eine Jdbc-Verbindung

Das folgende Beispiel zeigt, wie die HTTP-Verbindungsvorlage für einen Jdbc-Verbindungstyp in Apache Airflow v2.0.2 und dieselben Werte in der Jdbc-Verbindungsvorlage für Apache Airflow v1.10.12 in der Apache Airflow-Benutzeroberfläche verwendet werden.

### Apache Airflow v2

Das folgende Beispiel zeigt die Verbindungs-URI-Zeichenfolge, die von Apache Airflow für das Beispiel in diesem Abschnitt generiert wurde.

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

Das folgende Beispiel zeigt, wie die HTTP-Verbindungsvorlage für eine Jdbc-Verbindung für Apache Airflow v2 in der Apache Airflow-Benutzeroberfläche verwendet wird.

**Add Connection**

|                    |                                                                                                                                                                      |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Conn Id *</b>   | <input type="text" value="my_jdbc_conn"/>                                                                                                                            |
| <b>Conn Type *</b> | <input type="text" value="HTTP"/><br><small>Conn Type missing? Make sure you've installed the corresponding Airflow Provider Package.</small>                        |
| <b>Description</b> | <input style="height: 40px;" type="text"/>                                                                                                                           |
| <b>Host</b>        | <input type="text" value="myconnectionurl/some/path"/>                                                                                                               |
| <b>Schema</b>      | <input type="text"/>                                                                                                                                                 |
| <b>Login</b>       | <input type="text" value="mylogin"/>                                                                                                                                 |
| <b>Password</b>    | <input type="text"/>                                                                                                                                                 |
| <b>Port</b>        | <input type="text"/>                                                                                                                                                 |
| <b>Extra</b>       | <pre>{   "extra__jdbc__drv__path": "/usr/local/airflow/dags/classpath/redshift-jdbc42-2.0.0.1.jar",   "extra__jdbc__drv__clsname": "redshift-jdbc42-2.0.0.1" }</pre> |

## Apache Airflow v1

Das folgende Beispiel zeigt die Verbindungs-URI-Zeichenfolge, die von Apache Airflow für das Beispiel in diesem Abschnitt generiert wurde.

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

Das folgende Beispiel zeigt die Jdbc-Verbindungsvorlage für Apache Airflow v1.10.12 in der Apache Airflow-Benutzeroberfläche.

| Add Connection |                                                                                            |
|----------------|--------------------------------------------------------------------------------------------|
| Conn Id *      | <input type="text" value="my_jdbc_conn"/>                                                  |
| Conn Type      | <input type="text" value="Jdbc Connection"/>                                               |
| Connection URL | <input type="text" value="myconnectionrurl/some/path"/>                                    |
| Login          | <input type="text" value="mylogin"/>                                                       |
| Password       | <input type="password"/>                                                                   |
| Driver Path    | <input type="text" value="/usr/local/airflow/dags/classpath/redshift-jdbc42-2.0.0.1.jar"/> |
| Driver Class   | <input type="text" value="redshift-jdbc42-2.0.0.1"/>                                       |

## Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager

AWS Secrets Manager ist ein unterstütztes alternatives Apache Airflow-Backend in einer Amazon Managed Workflows für Apache Airflow-Umgebung. Diese Anleitung zeigt, wie Sie AWS Secrets Manager Geheimnisse für Apache Airflow-Variablen und eine Apache Airflow-Verbindung auf Amazon Managed Workflows for Apache Airflow sicher speichern können.

### Note

- Die von Ihnen erstellten Geheimnisse werden Ihnen in Rechnung gestellt. Weitere Informationen zu den Preisen von Secrets Manager finden Sie unter [AWS Preise](#).

### Inhalt

- [Schritt eins: Erteilen Sie Amazon MWAA die Erlaubnis, auf geheime Schlüssel von Secrets Manager zuzugreifen](#)

- [Schritt zwei: Erstellen Sie das Secrets Manager Manager-Backend als Apache Airflow-Konfigurationsoption](#)
- [Schritt drei: Generieren Sie eine Apache AWS Airflow-Verbindungs-URI-Zeichenfolge](#)
- [Schritt 4: Fügen Sie die Variablen in Secrets Manager hinzu](#)
- [Schritt fünf: Fügen Sie die Verbindung in Secrets Manager hinzu](#)
- [Beispiel-Code](#)
- [Ressourcen](#)
- [Als nächstes](#)

## Schritt eins: Erteilen Sie Amazon MWAA die Erlaubnis, auf geheime Schlüssel von Secrets Manager zuzugreifen

Die [Ausführungsrolle](#) für Ihre Amazon MWAA-Umgebung benötigt Lesezugriff auf den geheimen Schlüssel in. AWS Secrets Manager Die folgende IAM-Richtlinie ermöglicht Lese- und Schreibzugriff mithilfe der verwalteten Richtlinie. AWS [SecretsManagerReadWrite](#)

Um die Richtlinie Ihrer Ausführungsrolle zuzuordnen

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich „Berechtigungen“ Ihre Ausführungsrolle aus.
4. Wählen Sie Richtlinien anfügen.
5. Geben Sie `SecretsManagerReadWrite` in das Textfeld Filterrichtlinien ein.
6. Wählen Sie Richtlinie anfügen aus.

Wenn Sie keine AWS verwaltete Berechtigungsrichtlinie verwenden möchten, können Sie die Ausführungsrolle Ihrer Umgebung direkt aktualisieren, um eine beliebige Zugriffsebene auf Ihre Secrets Manager Manager-Ressourcen zu ermöglichen. Die folgende Richtlinienanweisung gewährt beispielsweise Lesezugriff auf alle Secrets, die Sie in einer bestimmten AWS Region in Secrets Manager erstellen.

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

```

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

```

## Schritt zwei: Erstellen Sie das Secrets Manager Manager-Backend als Apache Airflow-Konfigurationsoption

Im folgenden Abschnitt wird beschrieben, wie Sie eine Apache Airflow-Konfigurationsoption auf der Amazon MWAA-Konsole für das Backend erstellen. AWS Secrets Manager Wenn Sie in eine gleichnamige Konfigurationseinstellung verwenden, hat die Konfiguration `airflow.cfg`, die Sie in den folgenden Schritten erstellen, Vorrang und überschreibt die Konfigurationseinstellungen.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie Weiter aus.
5. Wählen Sie im Bereich mit den Airflow-Konfigurationsoptionen die Option Benutzerdefinierte Konfiguration hinzufügen aus. Fügen Sie die folgenden Schlüssel-Wert-Paare hinzu:
  - a. **secrets.backend:**  
**airflow.providers.amazon.aws.secrets.secrets\_manager.SecretsManagerBackend**
  - b. **secrets.backend\_kwargs:** `{"connections_prefix" : "airflow/connections", "variables_prefix" : "airflow/variables"}` Dadurch wird Apache Airflow so konfiguriert, dass es unter Pfaden nach Verbindungszeichenfolgen und Variablen sucht. `airflow/connections/* airflow/variables/*`

Sie können ein [Suchmuster](#) verwenden, um die Anzahl der API-Aufrufe zu reduzieren, die Amazon MWAA in Ihrem Namen an Secrets Manager durchführt. Wenn Sie kein Suchmuster angeben, sucht Apache Airflow im konfigurierten Backend nach allen Verbindungen und Variablen. Indem Sie ein Muster angeben, schränken Sie die möglichen Pfade ein, nach denen Apache Airflow sucht. Dies senkt Ihre Kosten, wenn Sie Secrets Manager mit Amazon MWAA verwenden.

Um ein Suchmuster anzugeben, geben Sie die Parameter `connections_lookup_pattern` und `variables_lookup_pattern` an. Diese Parameter akzeptieren eine RegEx Zeichenfolge als Eingabe. Um beispielsweise nach Geheimnissen zu suchen, die mit `beginntest`, geben Sie Folgendes ein `secrets.backend_kwargs`:

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

#### Note

Um `connections_lookup_pattern` und verwenden zu können `variables_lookup_pattern`, müssen Sie `apache-airflow-providers-amazon` Version 7.3.0 oder höher installieren. Weitere Informationen zum Aktualisieren von Anbieterpaketen auf neuere Versionen finden Sie unter [the section called "Angeben neuerer Anbieterpakete"](#)

6. Wählen Sie Speichern.

## Schritt drei: Generieren Sie eine Apache AWS Airflow-Verbindungs-URI-Zeichenfolge

[Um eine Verbindungszeichenfolge zu erstellen, verwenden Sie die Tabulatortaste auf Ihrer Tastatur, um die Schlüssel-Wert-Paare im Connection-Objekt einzurücken.](#) Wir empfehlen außerdem, in Ihrer Shell-Sitzung eine Variable für das `extra` Objekt zu erstellen. Der folgende Abschnitt führt Sie durch



die Schritte zum [Generieren einer Apache Airflow-Verbindungs-URI-Zeichenfolge](#) für eine Amazon MWAA-Umgebung mithilfe von Apache Airflow oder einem Python-Skript.

## Apache Airflow CLI

Die folgende Shell-Sitzung verwendet Ihre lokale Airflow-CLI, um eine Verbindungszeichenfolge zu generieren. Wenn Sie die CLI nicht installiert haben, empfehlen wir die Verwendung des Python-Skripts.

1. Öffnen Sie eine Python-Shell-Sitzung:

```
python3
```

2. Geben Sie den folgenden Befehl ein:

```
>>> import json
```

3. Geben Sie den folgenden Befehl ein:

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

4. Erstellen Sie in Ihrer Shell-Sitzung eine Variable für das extra Objekt. *Ersetzen Sie die Beispielwerte in YOUR\_EXECUTION\_ROLE\_ARN durch den ARN der Ausführungsrolle und die Region in YOUR\_REGION (z. B.). us-east-1*

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

5. Erstellen Sie das Verbindungsobjekt. Ersetzen Sie den Beispielwert myconn durch den Namen der Apache Airflow-Verbindung.

```
>>> myconn = Connection(

```

6. Verwenden Sie die Tabulatortaste auf Ihrer Tastatur, um jedes der folgenden Schlüssel-Wert-Paare in Ihrem Verbindungsobjekt einzurücken. *Ersetzen Sie die Beispielwerte durch Rot.*

- a. Geben Sie den AWS Verbindungstyp an:

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

- b. Geben Sie die Apache Airflow-Datenbankoption an:

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

- c. Geben Sie die Apache Airflow-UI-URL auf Amazon MWAA an:

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

- d. Geben Sie die AWS Zugriffsschlüssel-ID (Benutzername) an, um sich bei Amazon MWAA anzumelden:

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

- e. Geben Sie den AWS geheimen Zugriffsschlüssel (Passwort) an, um sich bei Amazon MWAA anzumelden:

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

- f. Geben Sie die extra Shell-Sitzungsvariable an:

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

- g. Schließt das Verbindungsobjekt.

```
...)
```

7. Drückt die Verbindungs-URI-Zeichenfolge aus:

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

Sie sollten die Verbindungs-URI-Zeichenfolge in der Antwort sehen:

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

## Python script

Das folgende Python-Skript benötigt die Apache Airflow CLI nicht.

1. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `mwa_connection.py`

```
import urllib.parse

conn_type = 'YOUR_DB_OPTION'
host = 'YOUR_MWAA_AIRFLOW_UI_URL'
port = 'YOUR_PORT'
login = 'YOUR_AWS_ACCESS_KEY_ID'
password = 'YOUR_AWS_SECRET_ACCESS_KEY'
role_arn = urllib.parse.quote_plus('YOUR_EXECUTION_ROLE_ARN')
region_name = 'YOUR_REGION'

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

2. Ersetzen Sie die Platzhalter durch *Rot*.
3. Führen Sie das folgende Skript aus, um eine Verbindungszeichenfolge zu generieren.

```
python3 mwa_connection.py
```

## Schritt 4: Fügen Sie die Variablen in Secrets Manager hinzu

Im folgenden Abschnitt wird beschrieben, wie Sie das Geheimnis für eine Variable in Secrets Manager erstellen.

Um das Geheimnis zu erstellen

1. Öffnen Sie die [AWS Secrets Manager -Konsole](#).
2. Wählen Sie Store a new secret (Ein neues Secret speichern).
3. Wählen Sie Andere Art von Geheimnis.
4. Wählen Sie im Bereich Geben Sie die Schlüssel/Wert-Paare an, die in diesem geheimen Schlüssel gespeichert werden sollen, die Option Klartext aus.

- Fügen Sie den Variablenwert als Klartext im folgenden Format hinzu.

```
"YOUR_VARIABLE_VALUE"
```

Um beispielsweise eine Ganzzahl anzugeben:

```
14
```

Um beispielsweise eine Zeichenfolge anzugeben:

```
"mystring"
```

- Wählen Sie für den Verschlüsselungsschlüssel eine AWS KMS Schlüsseloption aus der Dropdownliste aus.
- Geben Sie in das Textfeld für Geheimer Name einen Namen im folgenden Format ein.

```
airflow/variables/YOUR_VARIABLE_NAME
```

Beispielsweise:

```
airflow/variables/test-variable
```

- Wählen Sie Weiter aus.
- Gehen Sie auf der Seite Geheimen Schlüssel konfigurieren im Bereich Geheimer Name und Beschreibung wie folgt vor.
  - Geben Sie unter Geheimer Name einen Namen für Ihr Geheimnis ein.
  - (Optional) Geben Sie unter Beschreibung eine Beschreibung für Ihr Geheimnis ein.

Wählen Sie Weiter aus.

- Behalten Sie unter Rotation konfigurieren — optional die Standardoptionen bei und wählen Sie Weiter.
- Wiederholen Sie diese Schritte in Secrets Manager für alle zusätzlichen Variablen, die Sie hinzufügen möchten.
- Überprüfen Sie auf der Seite „Überprüfen“ Ihr Geheimnis und wählen Sie dann Speichern aus.

## Schritt fünf: Fügen Sie die Verbindung in Secrets Manager hinzu

Im folgenden Abschnitt wird beschrieben, wie Sie das Geheimnis für Ihre Verbindungszeichenfolge-URI in Secrets Manager erstellen.

Um das Geheimnis zu erstellen

1. Öffnen Sie die [AWS Secrets Manager -Konsole](#).
2. Wählen Sie Store a new secret (Ein neues Secret speichern).
3. Wählen Sie Andere Art von Geheimnis.
4. Wählen Sie im Bereich Geben Sie die Schlüssel/Wert-Paare an, die in diesem geheimen Schlüssel gespeichert werden sollen, die Option Klartext aus.
5. Fügen Sie die Verbindungs-URI-Zeichenfolge als Klartext im folgenden Format hinzu.

```
YOUR_CONNECTION_URI_STRING
```

Beispielsweise:

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

### Warning

Apache Airflow analysiert jeden der Werte in der Verbindungszeichenfolge. Sie dürfen weder einfache noch doppelte Anführungszeichen verwenden, da die Verbindung sonst als einzelne Zeichenfolge analysiert wird.

6. Wählen Sie für den Verschlüsselungsschlüssel eine AWS KMS Schlüsseloption aus der Dropdownliste aus.
7. Geben Sie in das Textfeld für Geheimer Name einen Namen im folgenden Format ein.

```
airflow/connections/YOUR_CONNECTION_NAME
```

Beispielsweise:

```
airflow/connections/myconn
```

8. Wählen Sie Weiter aus.
9. Gehen Sie auf der Seite Geheimen Schlüssel konfigurieren im Bereich Geheimer Name und Beschreibung wie folgt vor.
  - a. Geben Sie unter Geheimer Name einen Namen für Ihr Geheimnis ein.
  - b. (Optional) Geben Sie unter Beschreibung eine Beschreibung für Ihr Geheimnis ein.

Wählen Sie Weiter aus.

10. Behalten Sie unter Rotation konfigurieren — optional die Standardoptionen bei und wählen Sie Weiter.
11. Wiederholen Sie diese Schritte in Secrets Manager für alle zusätzlichen Variablen, die Sie hinzufügen möchten.
12. Überprüfen Sie auf der Seite „Überprüfen“ Ihr Geheimnis und wählen Sie dann Speichern aus.

## Beispiel-Code

- Erfahren Sie auf dieser Seite anhand des Beispielcodes unter, wie Sie den geheimen Schlüssel für die Apache Airflow-Verbindung (myconn) verwenden. [Mit einem geheimen Schlüssel inAWS Secrets Managerfür eine Apache Airflow-Verbindung](#)
- Erfahren Sie auf dieser Seite anhand des Beispielcodes unter, wie Sie den geheimen Schlüssel für die Apache Airflow-Variable (test-variable) verwenden. [Mit einem geheimen Schlüssel inAWS Secrets Managerfür eine Apache Airflow-Variable](#)

## Ressourcen

- Weitere Informationen zur Konfiguration von Secrets Manager Manager-Geheimnissen mit der Konsole und dem AWS CLI finden [Sie unter Create a Secret](#) im AWS Secrets Manager Benutzerhandbuch.
- Verwenden Sie ein Python-Skript, um eine große Menge von Apache Airflow-Variablen und -Verbindungen zu Secrets Manager zu migrieren, siehe [Move your Apache Airflow connections and variables](#) to. AWS Secrets Manager

## Als nächstes

- Erfahren Sie unter, wie Sie ein Token für den Zugriff auf die Apache Airflow-Benutzeroberfläche generieren. [Zugreifen auf Apache Airflow](#)

# Verwaltung von Amazon MWAA-Umgebungen

Die Amazon Managed Workflows for Apache Airflow-Konsole enthält integrierte Optionen zur Konfiguration des privaten oder öffentlichen Zugriffs auf die Apache Airflow-Benutzeroberfläche. Sie enthält auch integrierte Optionen zur Konfiguration der Umgebungsgröße und zum Zeitpunkt der Skalierung von Workern sowie Apache Airflow-Konfigurationsoptionen, mit denen Sie Apache Airflow-Konfigurationen überschreiben können, auf die normalerweise nur in zugegriffen werden kann. `airflow.cfg` In diesem Handbuch wird beschrieben, wie Sie diese Konfigurationen auf der Amazon MWAA-Konsole verwenden.

## Themen

- [Konfiguration der Amazon MWAA-Umgebungs-klasse](#)
- [Konfiguration der automatischen Skalierung von Amazon MWAA Worker](#)
- [Konfiguration der automatischen Skalierung des Amazon MWAA-Webservers](#)
- [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#)
- [Aktualisierung der Apache Airflow-Version](#)
- [Verwenden eines Startskripts mit Amazon MWAA](#)

## Konfiguration der Amazon MWAA-Umgebungs-klasse

Die Umgebungs-klasse, die Sie für Ihre Amazon MWAA-Umgebung wählen, bestimmt die Größe der AWS-verwalteten AWS Fargate Container, in denen der [Celery Executor ausgeführt](#) wird, und der -verwalteten Amazon Aurora PostgreSQL-Metadaten-datenbank, in der die AWS Apache Airflow-Scheduler Task-Instances erstellen. Auf dieser Seite werden die einzelnen Amazon MWAA-Umgebungs-klassen und die Schritte zur Aktualisierung der Umgebungs-klasse auf der Amazon MWAA-Konsole beschrieben.

## Sections

- [Funktionen der Umgebung](#)
- [Apache Airflow Scheduler](#)



## Funktionen der Umgebung

Der folgende Abschnitt enthält die standardmäßigen gleichzeitigen Apache Airflow-Aufgaben, Random Access Memory (RAM) und die virtuellen zentralen Verarbeitungseinheiten (vCPUs) für jede Umgebungsklasse. Bei den aufgeführten gleichzeitigen Aufgaben wird davon ausgegangen, dass die Parallelität der Aufgaben die Apache Airflow Worker-Kapazität in der Umgebung nicht überschreitet.

In der folgenden Tabelle bezieht sich die DAG-Kapazität auf DAG-Definitionen, nicht auf Ausführungen, und es wird davon ausgegangen, dass Ihre DAGs [dynamisch](#) in einer einzigen Python-Datei sind und mit den Best Practices von [Apache Airflow](#) geschrieben wurden.

Die Ausführung von Aufgaben hängt davon ab, wie viele gleichzeitig geplant sind. Dabei wird davon ausgegangen, dass die Anzahl der DAG-Ausführungen, die zur gleichen Zeit gestartet werden sollen, die Standardeinstellung nicht überschreitet [max\\_dagruns\\_per\\_loop\\_to\\_schedule](#), ebenso wie die Größe und Anzahl der Worker, wie in diesem Thema beschrieben.

### mw1.small

- Bis zu 50 DAG-Kapazität
- 5 gleichzeitige Aufgaben (standardmäßig)
- 1 vCPUs
- 2 GB RAM

### mw1.medium

- Bis zu 200 DAG-Kapazität
- 10 gleichzeitige Aufgaben (standardmäßig)
- 2 vCPUs
- 4 GB RAM

### mw1.large

- Bis zu 1000 DAG-Kapazität
- 20 gleichzeitige Aufgaben (standardmäßig)
- 4 vCPUs
- 8 GB RAM

## mw1.xlarge

- Bis zu 2000 DAG-Kapazität
- 40 gleichzeitige Aufgaben (standardmäßig)
- 8 vCPUs
- 24 GB RAM

## mw1.2xlarge

- Bis zu 4000 DAG-Kapazität
- 80 gleichzeitige Aufgaben (standardmäßig)
- 16 vCPUs
- 48 GB RAM

Sie können es verwenden `celery.worker_autoscale`, um die Anzahl der Aufgaben pro Mitarbeiter zu erhöhen. Weitere Informationen hierzu finden Sie unter [the section called “Beispiel für einen Hochleistungs-Anwendungsfall”](#).

## Apache Airflow Scheduler

Der folgende Abschnitt enthält die Apache Airflow-Scheduler-Optionen, die auf der Amazon MWAA verfügbar sind, und wie sich die Anzahl der Scheduler auf die Anzahl der Trigger auswirkt.

In Apache Airflow verwaltet ein [Trigger Aufgaben, die er aufschiebt](#), bis bestimmte, mithilfe eines Triggers festgelegte Bedingungen erfüllt sind. In Amazon MWAA wird der Triggerer zusammen mit dem Scheduler für dieselbe Fargate-Aufgabe ausgeführt. Eine Erhöhung der Anzahl der Scheduler erhöht entsprechend die Anzahl der verfügbaren Trigger und optimiert so die Art und Weise, wie die Umgebung verzögerte Aufgaben verwaltet. Dadurch wird eine effiziente Bearbeitung von Aufgaben gewährleistet und sie werden umgehend so geplant, dass sie ausgeführt werden, wenn die Bedingungen erfüllt sind.

### Apache Airflow v2

- v2 — Akzeptiert zwischen 2 bis 5. Standardeinstellung: 2.

# Konfiguration der automatischen Skalierung von Amazon MWAA Worker

Der Auto-Scaling-Mechanismus erhöht automatisch die Anzahl der Apache Airflow-Worker als Reaktion auf laufende und in der Warteschlange stehende Aufgaben in Ihrer Amazon Managed Workflows for Apache Airflow-Umgebung und entsorgt zusätzliche Mitarbeiter, wenn keine Aufgaben mehr in der Warteschlange stehen oder ausgeführt werden. Auf dieser Seite wird beschrieben, wie Sie Auto Scaling konfigurieren können, indem Sie mithilfe der Amazon MWAA-Konsole die maximale Anzahl von Apache Airflow-Workern angeben, die in Ihrer Umgebung ausgeführt werden.

## Note

Amazon MWAA verwendet Apache Airflow-Metriken, um zu ermitteln, wann zusätzliche [Celery Executor-Mitarbeiter](#) benötigt werden, und erhöht bei Bedarf die Anzahl der Fargate-Mitarbeiter auf den von angegebenen Wert. `max-workers` Wenn die zusätzlichen Mitarbeiter ihre Arbeit erledigen und die Arbeitsbelastung abnimmt, entfernt Amazon MWAA sie und reduziert somit wieder den Wert von `min-workers`

Wenn Mitarbeiter beim Herunterskalieren neue Aufgaben übernehmen, behält Amazon MWAA die Fargate-Ressource und entfernt den Mitarbeiter nicht. Weitere Informationen finden Sie unter [So funktioniert Amazon MWAA Auto Scaling](#).

## Sections

- [So funktioniert Worker Scaling](#)
- [Verwenden der Amazon MWAA-Konsole](#)
- [Beispiel für einen Hochleistungs-Anwendungsfall](#)
- [Problembehandlung bei Aufgaben, die im laufenden Zustand hängen geblieben sind](#)
- [Als nächstes](#)

## So funktioniert Worker Scaling

Amazon MWAA verwendet `RunningTasks` `QueuedTasks` [Metriken](#), wobei  $(\text{ausgeführte Aufgaben} + \text{Aufgaben in der Warteschlange}) / (\text{Aufgaben pro Mitarbeiter}) = (\text{erforderliche Mitarbeiter})$ . Wenn die erforderliche Anzahl von Arbeitern die aktuelle Anzahl von Arbeitern übersteigt, fügt Amazon MWAA

Fargate-Arbeitercontainer zu diesem Wert hinzu, bis zu dem von angegebenen Höchstwert. `max-workers`

Da die Arbeitslast sinkt `RunningTasks` und die Summe der `QueuedTasks` Kennzahlen sinkt, fordert Amazon MWAA Fargate auf, die Anzahl der Mitarbeiter für die Umwelt zu reduzieren. Alle Mitarbeiter, die ihre Arbeit noch erledigen, bleiben während des Downscalings geschützt, bis sie ihre Arbeit abgeschlossen haben. Je nach Arbeitslast können Aufgaben in die Warteschlange gestellt werden, während die Mitarbeiter ihre Aufgaben herunterskalieren.

## Verwenden der Amazon MWAA-Konsole

Sie können die maximale Anzahl von Workern, die in Ihrer Umgebung gleichzeitig ausgeführt werden können, auf der Amazon MWAA-Konsole auswählen. Standardmäßig können Sie einen Höchstwert von bis zu 25 angeben.

Um die Anzahl der Arbeiter zu konfigurieren

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie Weiter aus.
5. Geben Sie im Bereich Umgebungsklasse einen Wert in das Feld Maximale Anzahl von Mitarbeitern ein.
6. Wählen Sie Speichern.

### Note

Es kann einige Minuten dauern, bis Änderungen in Ihrer Umgebung wirksam werden.

## Beispiel für einen Hochleistungs-Anwendungsfall

Im folgenden Abschnitt werden die Arten von Konfigurationen beschrieben, die Sie verwenden können, um hohe Leistung und Parallelität in einer Umgebung zu ermöglichen.

## Apache Airflow vor Ort

In der Regel würden Sie auf einer lokalen Apache Airflow-Plattform die Einstellungen für Aufgabenparallelität, auto Skalierung und Parallelität in Ihrer Datei konfigurieren: `airflow.cfg`

- `core.parallelism`— Die maximale Anzahl von Task-Instanzen, die pro Scheduler gleichzeitig ausgeführt werden können.
- `core.dag_concurrency`— Die maximale Parallelität für DAGs (keine Worker).
- `celery.worker_autoscale`— Die maximale und minimale Anzahl von Aufgaben, die gleichzeitig auf einem beliebigen Worker ausgeführt werden können.

Wenn beispielsweise auf eingestellt `core.parallelism` `core.dag_concurrency` war `100` und auf gesetzt wäre `7`, könnten Sie trotzdem nur die Gesamtzahl der `14` Aufgaben gleichzeitig ausführen, wenn Sie `2` DAGs hätten. Angenommen, jede DAG ist so eingestellt, dass sie nur sieben Aufgaben gleichzeitig (`in core.dag_concurrency`) ausführt, obwohl die Gesamtparallelität auf (`in`) eingestellt ist. `100 core.parallelism`

## In einer Amazon MWAA-Umgebung

In einer Amazon MWAA-Umgebung können Sie diese Einstellungen direkt auf der Amazon MWAA-Konsole mithilfe von [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA Konfiguration der Amazon MWAA-Umgebungs-kategorie](#), und dem auto Skalierungsmechanismus Maximum Worker Count konfigurieren. `core.dag_concurrency` Es ist zwar nicht in der Drop-down-Liste als Apache Airflow-Konfigurationsoption auf der Amazon MWAA-Konsole verfügbar, Sie können es jedoch als benutzerdefinierte [Apache Airflow-Konfigurationsoption](#) hinzufügen.

Nehmen wir an, Sie haben beim Erstellen Ihrer Umgebung die folgenden Einstellungen ausgewählt:

1. Die [Umgebungs-kategorie](#) `mw1.small`, die die maximale Anzahl gleichzeitiger Aufgaben steuert, die jeder Worker standardmäßig ausführen kann, sowie die vCPU von Containern.
2. Die Standardeinstellung für Mitarbeiter unter Maximale Anzahl von **10** Mitarbeitern.
3. Eine [Apache Airflow-Konfigurationsoption](#) für `celery.worker_autoscale` `4 5,5` Aufgaben pro Mitarbeiter.

Das bedeutet, dass Sie in Ihrer Umgebung `50` Aufgaben gleichzeitig ausführen können. Alle Aufgaben über `50` werden in die Warteschlange gestellt und warten, bis die laufenden Aufgaben abgeschlossen sind.

Führen Sie mehr Aufgaben gleichzeitig aus. Mithilfe der folgenden Konfigurationen können Sie Ihre Umgebung so ändern, dass mehr Aufgaben gleichzeitig ausgeführt werden:

1. [Erhöhen Sie die maximale Anzahl gleichzeitiger Aufgaben, die jeder Worker standardmäßig ausführen kann, und die vCPU von Containern, indem Sie die `mw1.medium` Umgebungsklasse \(standardmäßig 10 gleichzeitige Aufgaben\) auswählen.](#)
2. `celery.worker_autoscale` als [Apache Airflow-Konfigurationsoption](#) hinzufügen.
3. Erhöhen Sie die maximale Anzahl an Mitarbeitern. In diesem Beispiel 20 würde eine Erhöhung der maximalen Anzahl von Mitarbeitern von 10 auf die doppelte Anzahl gleichzeitiger Aufgaben, die die Umgebung ausführen kann, bedeuten.

Geben Sie die Mindestanzahl an Mitarbeitern an. Sie können auch die Mindest- und Höchstzahl von Apache Airflow Workern angeben, die in Ihrer Umgebung ausgeführt werden, indem Sie AWS Command Line Interface (AWS CLI) verwenden. Beispielsweise:

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

Weitere Informationen finden Sie unter dem Befehl [update-environment](#) im AWS CLI

## Problembehandlung bei Aufgaben, die im laufenden Zustand hängen geblieben sind

In seltenen Fällen geht Apache Airflow möglicherweise davon aus, dass Aufgaben noch ausgeführt werden. Um dieses Problem zu beheben, müssen Sie die nicht mehr benötigte Aufgabe in Ihrer Apache Airflow-Benutzeroberfläche löschen. Weitere Informationen finden Sie im Thema [Ich sehe, dass meine Aufgaben hängen bleiben oder nicht abgeschlossen werden](#) Problembehandlung.

### Als nächstes

- Erfahren Sie mehr über die bewährten Methoden, die wir empfehlen, um die Leistung Ihrer Umgebung zu optimieren [Leistungsoptimierung für Apache Airflow auf Amazon MWAA](#).

# Konfiguration der automatischen Skalierung des Amazon MWAA-Webservers

Für Umgebungen, in denen Apache Airflow v2.2.2 und höher ausgeführt wird, skaliert Amazon MWAA Ihre Webserver dynamisch, um schwankende Arbeitslasten zu bewältigen, wodurch wiederum Leistungsprobleme bei Spitzenlasten vermieden werden. Durch die automatische Skalierung der Anzahl der Webserver auf der Grundlage der CPU-Auslastung und der Anzahl der aktiven Verbindungen stellt Amazon MWAA sicher, dass Ihre Apache Airflow-Umgebung die gestiegene Nachfrage problemlos bewältigen kann, sei es durch REST-API-Anfragen, CLI-Nutzung oder mehr gleichzeitige Benutzer der Apache Airflow-Benutzeroberfläche.

## Sections

- [So funktioniert die Skalierung von Webservern](#)
- [Verwenden der Amazon MWAA-Konsole](#)

## So funktioniert die Skalierung von Webservern

Amazon MWAA verwendet die Container-Metrik und die Load Balancer-Metrik [ActiveConnectionCount](#), um anhand der Menge des Datenverkehrs zu bestimmen, ob eine Skalierung der Webserver erforderlich ist. [CPUUtilization](#) Wenn CPUUtilization höher als 70 oder höher als 15 ActiveConnectionCount ist, fügt Amazon MWAA zusätzliche Fargate-Webserver-Container bis zu dem von angegebenen Höchstwert hinzu. `MaxWebServers`

Wenn der Traffic abnimmt und die ActiveConnectionCount Werte CPUUtilization und sinken, fordert Amazon MWAA Fargate auf, die Webserver-Container für die Umgebung auf den von festgelegten Mindestwert herunterzuskalieren. `MinimumWebServers`


## Verwenden der Amazon MWAA-Konsole

Sie können die Anzahl der Webserver, die gleichzeitig in Ihrer Umgebung ausgeführt werden können, auf der Amazon MWAA-Konsole auswählen. Standardmäßig beträgt die Mindestanzahl von Webservern zwei und die maximale Anzahl von Webservern fünf.

Um die Anzahl der Webserver zu konfigurieren

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.

3. Wählen Sie Bearbeiten aus.
4. Wählen Sie Weiter aus.
5. Geben Sie im Bereich Umgebungsklasse einen Wert in das Feld Maximale Anzahl von Webservern ein.
6. Geben Sie als Nächstes einen Wert in das Feld Minimale Anzahl an Webservern ein.
7. Wählen Sie Speichern.

 Note

Es kann einige Minuten dauern, bis Änderungen in Ihrer Umgebung wirksam werden.

## Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA

Apache Airflow-Konfigurationsoptionen können als Umgebungsvariablen an Ihre Amazon Managed Workflows for Apache Airflow-Umgebung angehängt werden. Sie können aus der vorgeschlagenen Drop-down-Liste wählen oder benutzerdefinierte Konfigurationsoptionen für Ihre Apache Airflow-Version auf der Amazon MWAA-Konsole angeben. Auf dieser Seite werden die verfügbaren Apache Airflow-Konfigurationsoptionen beschrieben und wie Sie diese Optionen verwenden können, um die Apache Airflow-Konfigurationseinstellungen in Ihrer Umgebung zu überschreiben.

### Inhalt

- [Voraussetzungen](#)
- [Funktionsweise](#)
- [Verwenden von Konfigurationsoptionen zum Laden von Plugins in Apache Airflow v2](#)
- [Übersicht über die Konfigurationsoptionen](#)
  - [Apache Airflow-Konfigurationsoptionen](#)
  - [Apache Airflow-Referenz](#)
  - [Verwenden der Amazon MWAA-Konsole](#)
- [Konfigurationsreferenz](#)
  - [E-Mail-Konfigurationen](#)
  - [Aufgabenkonfigurationen](#)



- [Scheduler-Konfigurationen](#)
- [Worker-Konfigurationen](#)
- [Webserver-Konfigurationen](#)
- [Konfigurationen auslösen](#)
- [Beispiele und Beispielcode](#)
  - [Beispiel DAG](#)
  - [Beispiel für Einstellungen für E-Mail-Benachrichtigungen](#)
- [Als nächstes](#)

## Voraussetzungen

Sie benötigen Folgendes, bevor Sie die Schritte auf dieser Seite ausführen können.

- Berechtigungen — Ihr AWS Konto muss von Ihrem Administrator Zugriff auf die [FullConsoleAmazonMWAA-Zugriffskontrollrichtlinie](#) für Ihre Umgebung erhalten haben. Darüber hinaus muss Ihrer Amazon MWAA-Umgebung von Ihrer [Ausführungsrolle](#) der Zugriff auf die von Ihrer Umgebung verwendeten AWS Ressourcen gestattet werden.
- Zugriff — Wenn Sie Zugriff auf öffentliche Repositorys benötigen, um Abhängigkeiten direkt auf dem Webserver zu installieren, muss Ihre Umgebung für den Zugriff auf öffentliche Netzwerk-Webserver konfiguriert sein. Weitere Informationen finden Sie unter [the section called “Apache Airflow-Zugriffsmodi”](#).
- Amazon S3 S3-Konfiguration — Der [Amazon S3 S3-Bucket](#), der zum Speichern Ihrer DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten verwendet wird `plugins.zip`, `requirements.txt` muss mit geblocktem öffentlichem Zugriff und aktivierter Versionierung konfiguriert sein.

## Funktionsweise

Wenn Sie eine Umgebung erstellen, fügt Amazon MWAA die Konfigurationseinstellungen, die Sie auf der Amazon MWAA-Konsole in den Airflow-Konfigurationsoptionen angeben, als Umgebungsvariablen an den Container für Ihre Umgebung an. AWS Fargate Wenn Sie eine Einstellung mit demselben Namen in verwenden, überschreiben die Optionen `airflow.cfg`, die Sie auf der Amazon MWAA-Konsole angeben, die Werte in `airflow.cfg`

Wir stellen die `airflow.cfg` in der Apache Airflow-Benutzeroberfläche einer Amazon MWAA-Umgebung zwar nicht standardmäßig zur Verfügung, aber Sie können die Apache Airflow-Konfigurationsoptionen direkt in der Amazon MWAA-Konsole ändern, einschließlich der Einstellung, dass die Konfigurationen verfügbar gemacht werden. `webserver.expose_config`

## Verwenden von Konfigurationsoptionen zum Laden von Plugins in Apache Airflow v2

Standardmäßig sind Plugins in Apache Airflow v2 so konfiguriert, dass sie mithilfe der Einstellung „träge“ geladen werden. `core.lazy_load_plugins : True` Wenn Sie benutzerdefinierte Plugins in Apache Airflow v2 verwenden, müssen Sie `core.lazy_load_plugins : False` als Apache Airflow-Konfigurationsoption hinzufügen, um Plugins zu Beginn jedes Airflow-Prozesses zu laden, um die Standardeinstellung zu überschreiben.

## Übersicht über die Konfigurationsoptionen

Wenn Sie eine Konfiguration auf der Amazon MWAA-Konsole hinzufügen, schreibt Amazon MWAA die Konfiguration als Umgebungsvariable.

- Aufgelistete Optionen. In der Drop-down-Liste können Sie aus einer der für Ihre Apache Airflow-Version verfügbaren Konfigurationseinstellungen wählen. Zum Beispiel: `dag_concurrency 16` Die Konfigurationseinstellung wird in den Fargate-Container Ihrer Umgebung übersetzt als `AIRFLOW__CORE__DAG_CONCURRENCY : 16`
- Benutzerdefinierte Optionen. Sie können auch Airflow-Konfigurationsoptionen angeben, die für Ihre Apache Airflow-Version nicht in der Drop-down-Liste aufgeführt sind. Zum Beispiel: `foo.user YOUR_USER_NAME` Die Konfigurationseinstellung wird in den Fargate-Container Ihrer Umgebung übersetzt als `AIRFLOW__FOO__USER : YOUR_USER_NAME`

## Apache Airflow-Konfigurationsoptionen

Die folgende Abbildung zeigt, wo Sie die Apache Airflow-Konfigurationsoptionen auf der Amazon MWAA-Konsole anpassen können.

## Airflow configuration options - optional [Info](#)

Modify the default settings for Airflow configuration options. You can select an option from the suggestion list or type one manually.

All Airflow configuration options are using default values.

**Add custom configuration value**

## Apache Airflow-Referenz

Eine Liste der von Apache Airflow unterstützten Konfigurationsoptionen finden Sie unter [Konfigurationsreferenz im Apache Airflow-Referenzhandbuch](#). Um die Optionen für die Version von Apache Airflow anzuzeigen, die Sie auf Amazon MWAA ausführen, wählen Sie die Version aus der Drop-down-Liste aus.

## Verwenden der Amazon MWAA-Konsole

Das folgende Verfahren führt Sie durch die Schritte zum Hinzufügen einer Airflow-Konfigurationsoption zu Ihrer Umgebung.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie Weiter aus.
5. Wählen Sie im Bereich mit den Airflow-Konfigurationsoptionen die Option Benutzerdefinierte Konfiguration hinzufügen aus.
6. Wählen Sie eine Konfiguration aus der Dropdownliste aus und geben Sie einen Wert ein, oder geben Sie eine benutzerdefinierte Konfiguration ein und geben Sie einen Wert ein.
7. Wählen Sie für jede Konfiguration, die Sie hinzufügen möchten, die Option Benutzerdefinierte Konfiguration hinzufügen aus.
8. Wählen Sie Speichern.

## Konfigurationsreferenz

Der folgende Abschnitt enthält die Liste der verfügbaren Apache Airflow-Konfigurationen in der Drop-down-Liste auf der Amazon MWAA-Konsole.

### E-Mail-Konfigurationen

Die folgende Liste zeigt die Konfigurationsoptionen für Airflow-E-Mail-Benachrichtigungen, die auf Amazon MWAA verfügbar sind.

Wir empfehlen die Verwendung von Port 587 für SMTP-Verkehr. AWS Blockiert standardmäßig ausgehenden SMTP-Verkehr auf Port 25 aller Amazon EC2 EC2-Instances. Wenn Sie ausgehenden Datenverkehr auf Port 25 senden möchten, können Sie [beantragen, dass diese Einschränkung aufgehoben wird](#).

#### Apache Airflow v2

| Airflow-Version | Option zur Konfiguration von Airflow | Beschreibung                                                                                                                      | Beispielwert                        |
|-----------------|--------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|-------------------------------------|
| v2              | email.email_backend                  | <a href="#">Das Apache Airflow-Hilfsprogramm, das für E-Mail-Benachrichtigungen in email_backend verwendet wird.</a>              | airflow.utils.email.send_email_smtp |
| v2              | smtp.smtp_host                       | <a href="#">Der Name des Ausgangsservers, der für die E-Mail-Adresse in smtp_host verwendet wird.</a>                             | localhost                           |
| v2              | smtp.smtp_starttls                   | <a href="#">Transport Layer Security (TLS) wird verwendet, um die E-Mail über das Internet in smtp_starttls zu verschlüsseln.</a> | False                               |

| Airflow-Version | Option zur Konfiguration von Airflow | Beschreibung                                                                                                             | Beispielwert       |
|-----------------|--------------------------------------|--------------------------------------------------------------------------------------------------------------------------|--------------------|
| v2              | smtp.smtp_ssl                        | <a href="#">Secure Sockets Layer (SSL) wird verwendet, um den Server und den E-Mail-Client in smtp_ssl zu verbinden.</a> | True               |
| v2              | smtp.smtp_port                       | <a href="#">Der TCP-Port (Transmission Control Protocol), der dem Server in smtp_port zugewiesen wurde.</a>              | 587                |
| v2              | smtp.smtp_mail_von                   | <a href="#">Die ausgehende E-Mail-Adresse in smtp_mail_from.</a>                                                         | myemail@domain.com |

## Aufgabenkonfigurationen

Die folgende Liste zeigt die Konfigurationen, die in der Drop-down-Liste für Airflow-Aufgaben auf Amazon MWAA verfügbar sind.

### Apache Airflow v2

| Airflow-Version | Option zur Konfiguration von Airflow | Beschreibung                                                                                                 | Beispielwert |
|-----------------|--------------------------------------|--------------------------------------------------------------------------------------------------------------|--------------|
| v2              | core.default_task_retries            | <a href="#">Gibt an, wie oft eine Apache Airflow-Aufgabe in default_task_retries wiederholt werden soll.</a> | 3            |

| Airflow-Version | Option zur Konfiguration von Airflow | Beschreibung                                                                                                                                          | Beispielwert |
|-----------------|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| v2              | core.parallelism                     | Die maximale Anzahl von Task-Instanzen, die gleichzeitig in der gesamten Umgebung parallel ausgeführt werden können ( <a href="#">Parallelität</a> ). | 40           |

## Scheduler-Konfigurationen

Die folgende Liste zeigt die Apache Airflow Scheduler-Konfigurationen, die in der Drop-down-Liste auf Amazon MWAA verfügbar sind.

### Apache Airflow v2

| Airflow-Version | Option zur Konfiguration von Airflow      | Beschreibung                                                                                                                                                         | Beispielwert |
|-----------------|-------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| v2              | scheduler.catchup_by_default              | Weist den Scheduler an, einen DAG-Lauf zu erstellen, um das in <a href="#">catchup_by_default</a> angegebene Zeitintervall „catch“.                                  | False        |
| v2              | scheduler.scheduler_zombie_task_threshold | <a href="#">Teilt dem Scheduler mit, ob die Task-Instanz als fehlgeschlagen markiert und die Aufgabe in scheduler_zombie_task_threshold neu geplant werden soll.</a> | 300          |

## Worker-Konfigurationen

Die folgende Liste zeigt die Airflow-Worker-Konfigurationen, die in der Drop-down-Liste auf Amazon MWAA verfügbar sind.


### Apache Airflow v2

| Airflow-Version | Option zur Konfiguration von Airflow | Beschreibung                                                                                                                                                                                                                                                                                    | Beispielwert |
|-----------------|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| v2              | celery.worker_auto_scale             | <p><u>Die maximale und minimale Anzahl von Aufgaben, die gleichzeitig auf jedem Worker ausgeführt werden können, der den Celery Executor in worker_autoscale verwendet</u></p> <p>Der Wert muss in der folgenden Reihenfolge durch Kommas getrennt werden: max_concurrency, min_concurrency</p> | 16,12        |

## Webserver-Konfigurationen

Die folgende Liste zeigt die Airflow-Webserverkonfigurationen, die in der Drop-down-Liste auf Amazon MWAA verfügbar sind.

## Apache Airflow v2

| Airflow-Version | Option zur Konfiguration von Airflow | Beschreibung                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | Beispielwert     |
|-----------------|--------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| v2              | webserver.default_ui_timezone        | <p><a href="#">Die standardmäßige Datetime-Einstellung der Apache Airflow-Benutzeroberfläche in default_ui_timezone.</a></p> <div data-bbox="854 646 1162 1885"><p> <b>Note</b></p><p>Durch die Einstellung dieser default_ui_timezone Option wird die Zeitzone, in der Ihre DAGs ausgeführt werden sollen, nicht geändert. Um die Zeitzone für Ihre DAGs zu ändern, können Sie ein benutzerdefiniertes Plugin verwenden. Weitere</p></div> | Amerika/New_York |



| Airflow-Version | Option zur Konfiguration von Airflow | Beschreibung                                                                                    | Beispielwert |
|-----------------|--------------------------------------|-------------------------------------------------------------------------------------------------|--------------|
|                 |                                      | Informationen finden Sie unter <a href="#">the section called "Zeitzone einer DAG ändern"</a> . |              |

## Konfigurationen auslösen

Die folgende Liste zeigt die Apache [Airflow-Triggerkonfigurationen](#), die auf Amazon MWAA verfügbar sind.

### Apache Airflow v2

| Airflow-Version | Option zur Konfiguration von Airflow  | Beschreibung                                                                                                                                                                                                                                           | Beispielwert |
|-----------------|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| v2.7            | <code>mwaa.triggerer_aktiviert</code> | Wird für die Aktivierung und Deaktivierung des Triggers auf Amazon MWAA verwendet. Standardmäßig ist dieser Wert auf <code>True</code> festgelegt. Wenn auf <code>False</code> gesetzt, startet Amazon MWAA keine auslösenden Prozesse auf Scheduling. | True         |

| Airflow-Version | Option zur Konfiguration von Airflow | Beschreibung                                                                                                                                                                                                                                                                                                                                                                 | Beispielwert |
|-----------------|--------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| v2.7            | triggerer.default_capacity           | Definiert die Anzahl der Trigger, die jeder Trigger parallel ausführen kann. Bei Amazon MWAA wird diese Kapazität für jeden Trigger und für jeden Scheduler festgelegt, da beide Komponenten parallel ausgeführt werden. Die Standardeinstellung pro Scheduler ist auf 60, 125, und 1000 für kleine, 250, 500, mittlere und große, xlarge- und 2xlarge-Instances festgelegt. | 125          |

## Beispiele und Beispielcode

### Beispiel DAG

Sie können die folgende DAG verwenden, um Ihre `email_backend` Apache Airflow-Konfigurationsoptionen zu drucken. Um ihn als Reaktion auf Amazon MWAA-Ereignisse auszuführen, kopieren Sie den Code in den DAGs-Ordner Ihrer Umgebung auf Ihrem Amazon S3 S3-Speicher-Bucket.

```
from airflow.decorators import dag
from datetime import datetime

def print_var(**kwargs):
```

```
email_backend = kwargs['conf'].get(section='email', key='email_backend')
print("email_backend")
return email_backend

@dag(
 dag_id="print_env_variable_example",
 schedule_interval=None,
 start_date=datetime(yyyy, m, d),
 catchup=False,
)
def print_variable_dag():
 email_backend_test = PythonOperator(
 task_id="email_backend_test",
 python_callable=print_var,
 provide_context=True
)

print_variable_test = print_variable_dag()
```

## Beispiel für Einstellungen für E-Mail-Benachrichtigungen

Die folgenden Apache Airflow-Konfigurationsoptionen können für ein Gmail.com-E-Mail-Konto mit einem App-Passwort verwendet werden. Weitere Informationen finden Sie [im Referenzhandbuch der Gmail-Hilfe unter Mit App-Passwörtern anmelden](#).

## Airflow configuration options - optional [Info](#)

Modify the default settings for Airflow configuration options. You can select an option from the suggestion list or type one manually.

| Configuration option                                          | Custom value                                                    |                                       |
|---------------------------------------------------------------|-----------------------------------------------------------------|---------------------------------------|
| <input type="text" value="smtp.smtp_host"/> X                 | <input type="text" value="smtp.gmail.com"/>                     | <input type="button" value="Remove"/> |
| <input type="text" value="smtp.smtp_mail_from"/> X            | <input type="text" value="&lt;your email&gt;@gmail.com"/>       | <input type="button" value="Remove"/> |
| <input type="text" value="smtp.smtp_password"/> X             | <input type="text" value="&lt;your 16 digit app password&gt;"/> | <input type="button" value="Remove"/> |
| <input type="text" value="smtp.smtp_port"/> X                 | <input type="text" value="587"/>                                | <input type="button" value="Remove"/> |
| <input type="text" value="smtp.smtp_ssl"/> X                  | <input type="text" value="False"/>                              | <input type="button" value="Remove"/> |
| <input type="text" value="smtp.smtp_starttls"/> X             | <input type="text" value="True"/>                               | <input type="button" value="Remove"/> |
| <input type="text" value="smtp.smtp_user"/> X                 | <input type="text" value="&lt;your email&gt;@gmail.com"/>       | <input type="button" value="Remove"/> |
| <input type="button" value="Add custom configuration value"/> |                                                                 |                                       |

## Als nächstes

- Erfahren Sie unter, wie Sie Ihren DAG-Ordner in Ihren Amazon S3 S3-Bucket hochladen [Hinzufügen oder Aktualisieren von DAGs](#).

## Aktualisierung der Apache Airflow-Version

Amazon MWAA unterstützt kleinere Versions-Upgrades. Das bedeutet, dass Sie Ihre Umgebung von Version zu Version  $x.4.z$  aktualisieren können.  $x.5.z$  Um ein Upgrade einer Hauptversion durchzuführen, z. B. von Version  $1.y.z$  zu Version  $2.y.z$ , müssen Sie eine neue Umgebung erstellen und Ihre Ressourcen migrieren. Weitere Informationen zum Upgrade auf eine neue Hauptversion von Apache Airflow finden Sie unter [Migration zu einer neuen Amazon MWAA-Umgebung im Amazon MWAA-Migrationshandbuch](#).

Während des Upgrade-Vorgangs erfasst Amazon MWAA einen Snapshot Ihrer Umgebungsmetadaten, aktualisiert die Worker, Scheduler und den Webserver auf die neue Apache Airflow-Version und stellt schließlich die Metadaten-Datenbank mithilfe des Snapshots wieder her.

#### Note

Sie können die Apache Airflow-Version für Ihre Umgebung nicht herunterstufen.

Stellen Sie vor dem Upgrade sicher, dass Ihre DAGs und andere Workflow-Ressourcen mit der neuen Apache Airflow-Version kompatibel sind, auf die Sie aktualisieren. Wenn Sie a `requirements.txt` zur Verwaltung von Abhängigkeiten verwenden, müssen Sie auch sicherstellen, dass die Abhängigkeiten, die Sie in Ihren Anforderungen angeben, mit der neuen Version kompatibel sind.

#### Themen

- [Aktualisieren Sie Ihre Workflow-Ressourcen](#)
- [Geben Sie die neue Version an](#)

## Aktualisieren Sie Ihre Workflow-Ressourcen

Wenn Sie Apache Airflow-Versionen ändern, stellen Sie sicher, dass Sie in Ihrer `requirements.txt` Version [auf die richtige `--constraint` URL](#) verweisen.

#### Warning

Wenn Sie während eines Upgrades Anforderungen angeben, die mit Ihrer Apache Airflow-Zielversion nicht kompatibel sind, kann dies zu einem langwierigen Rollback-Prozess auf die vorherige Version von Apache Airflow mit der vorherigen Anforderungsversion führen.

#### Um Ihre Workflow-Ressourcen zu migrieren

1. Erstellen Sie einen Fork des [aws-mwaa-local-runner](#) Repositorys und klonen Sie eine Kopie des lokalen Amazon MWAA-Runners.
2. Gehen Sie zu dem Zweig des `aws-mwaa-local-runner` Repositorys, der der Version entspricht, auf die Sie ein Upgrade durchführen.

3. Verwenden Sie das Amazon MWAA Local Runner CLI-Tool, um das Docker-Image zu erstellen und Apache Airflow lokal auszuführen. Weitere Informationen finden Sie in der [README-Datei](#) für den lokalen Runner im Repository. GitHub
4. Um Ihre zu aktualisieren `requirements.txt`, folgen Sie den bewährten Methoden, die wir unter [Verwaltung von Python-Abhängigkeiten](#) im Amazon MWAA-Benutzerhandbuch empfehlen.
5. (Optional) Um den Upgrade-Prozess zu beschleunigen, [bereinigen Sie die Metadaten-Datenbank der Umgebung](#). Das Upgrade von Umgebungen mit einer großen Menge an Metadaten kann erheblich länger dauern.
6. Nachdem Sie Ihre Workflow-Ressourcen erfolgreich getestet haben, kopieren Sie Ihre DAGs und Plugins in den Amazon S3 S3-Bucket Ihrer Umgebung. `requirements.txt`

Sie sind jetzt bereit, die Umgebung zu bearbeiten, eine neue Apache Airflow-Version anzugeben und den Aktualisierungsvorgang zu starten.

## Geben Sie die neue Version an

Nachdem Sie die Aktualisierung Ihrer Workflow-Ressourcen abgeschlossen haben, um die Kompatibilität mit der neuen Apache Airflow-Version sicherzustellen, gehen Sie wie folgt vor, um die Umgebungsdetails zu bearbeiten und die Version von Apache Airflow anzugeben, auf die Sie aktualisieren möchten.

### Note

Wenn Sie ein Upgrade durchführen, werden alle Aufgaben, die derzeit in der Umgebung ausgeführt werden, während des Vorgangs beendet. Der Aktualisierungsvorgang kann bis zu zwei Stunden dauern. Während dieser Zeit ist Ihre Umgebung nicht verfügbar.

Um eine neue Version mithilfe der Konsole anzugeben

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie aus der Liste Umgebungen die Umgebung aus, für die Sie ein Upgrade durchführen möchten.
3. Wählen Sie auf der Umgebungsseite Bearbeiten aus, um die Umgebung zu bearbeiten.
4. Wählen Sie im Abschnitt Umgebungsdetails für die Airflow-Version die neue Apache Airflow-Versionsnummer, auf die Sie die Umgebung aktualisieren möchten, aus der Dropdownliste aus.

5. Wählen Sie Weiter, bis Sie auf der Seite Überprüfen und speichern sind.
6. Überprüfen Sie auf der Seite Überprüfen und speichern Ihre Änderungen und wählen Sie dann Speichern.

Wenn Sie die Änderungen übernehmen, beginnt Ihre Umgebung mit dem Upgrade-Vorgang. Während dieses Zeitraums gibt der [Status](#) Ihrer Umgebung an, welche Maßnahmen Amazon MWAA ergreift und ob das Verfahren erfolgreich ist.

In einem erfolgreichen Upgrade-Szenario wird der Status angezeigt UPDATING, CREATING\_SNAPSHOT während Amazon MWAA eine Sicherungskopie Ihrer Metadaten erfasst. Schließlich kehrt der Status zuerst zu und dann zu dem AVAILABLE Zeitpunkt zurück UPDATING, an dem der Vorgang abgeschlossen ist.

Wenn die Umgebung nicht aktualisiert werden kann, wird Ihr Umgebungsstatus angezeigt ROLLING\_BACK. Wenn das Rollback erfolgreich ist, wird zunächst der Status angezeigt, was darauf hinweist UPDATE\_FAILED, dass das Update fehlgeschlagen ist, die Umgebung jedoch verfügbar ist. Wenn das Rollback fehlschlägt, wird der Status angezeigt, was darauf hinweist UNAVAILABLE, dass Sie nicht auf die Umgebung zugreifen können.

## Verwenden eines Startskripts mit Amazon MWAA

Ein Startskript ist ein Shell (.sh) -Skript, das Sie im Amazon S3 S3-Bucket Ihrer Umgebung hosten, ähnlich Ihren DAGs, Anforderungen und Plugins. Amazon MWAA führt dieses Skript beim Start auf jeder einzelnen Apache Airflow-Komponente (Worker, Scheduler und Webserver) aus, bevor die Anforderungen installiert und der Apache Airflow-Prozess initialisiert wird. Verwenden Sie ein Startskript, um Folgendes zu tun:

- Laufzeiten installieren — Installieren Sie Linux-Laufzeiten, die für Ihre Workflows und Verbindungen erforderlich sind.
- Umgebungsvariablen konfigurieren — Legen Sie Umgebungsvariablen für jede Apache Airflow-Komponente fest. Überschreiben Sie allgemeine Variablen wie PATHPYTHONPATH, und LD\_LIBRARY\_PATH
- Schlüssel und Token verwalten — Übergeben Sie Zugriffstoken für benutzerdefinierte Repositorys an requirements.txt und konfigurieren Sie Sicherheitsschlüssel.

In den folgenden Themen wird beschrieben, wie Sie ein Startskript konfigurieren, um Linux-Laufzeiten zu installieren, Umgebungsvariablen festzulegen und verwandte Probleme mithilfe von CloudWatch Protokollen zu beheben.

## Themen

- [Konfigurieren Sie ein Startskript](#)
- [Installieren Sie Linux-Laufzeiten mithilfe eines Startskripts](#)
- [Legen Sie Umgebungsvariablen mithilfe eines Startskripts fest](#)

## Konfigurieren Sie ein Startskript

Um ein Startskript mit Ihrer bestehenden Amazon MWAA-Umgebung zu verwenden, laden Sie eine `.sh` Datei in den Amazon S3 S3-Bucket Ihrer Umgebung hoch. Um das Skript dann der Umgebung zuzuordnen, geben Sie in Ihren Umgebungsdetails Folgendes an:

- Der Amazon S3 S3-URL-Pfad zum Skript — Der relative Pfad zu dem in Ihrem Bucket gehosteten Skript, zum Beispiel `s3://mwa-environment/startup.sh`
- Die Amazon S3 S3-Versions-ID des Skripts — Die Version des Start-Shell-Skripts in Ihrem Amazon S3 S3-Bucket. Sie müssen die [Versions-ID](#) angeben, die Amazon S3 der Datei bei jeder Aktualisierung des Skripts zuweist. Versions-IDs sind Unicode-kodierte, UTF-8-kodierte, URL-fähige, undurchsichtige Zeichenketten, die beispielsweise nicht länger als 1.024 Byte sind.  
`3sL4kqtJlcpXroDTdmJ+rmSpXd3dIbrHY+MTRCxf3vjVBH40Nr8X8gdRQBpUMLUo`

Verwenden Sie das folgende Beispielskript, um die Schritte in diesem Abschnitt auszuführen. Das Skript gibt den Wert aus, der zugewiesen wurde `MWAA_AIRFLOW_COMPONENT`. Diese Umgebungsvariable identifiziert jede Apache Airflow-Komponente, auf der das Skript ausgeführt wird.

Kopieren Sie den Code und speichern Sie ihn lokal unter `startup.sh`

```
#!/bin/sh

echo "Printing Apache Airflow component"
echo $MWAA_AIRFLOW_COMPONENT
```

Laden Sie als Nächstes das Skript in Ihren Amazon S3 S3-Bucket hoch.



## AWS Management Console

Um ein Shell-Skript (Konsole) hochzuladen

1. Melden Sie sich bei der AWS Management Console an und öffnen Sie die Amazon S3-Konsole unter <https://console.aws.amazon.com/s3/>.
2. Wählen Sie aus der Bucket-Liste den Namen des Buckets aus, der Ihrer Umgebung zugeordnet ist.
3. Wählen Sie auf der Registerkarte Objekte die Option Upload aus.
4. Ziehen Sie das von Ihnen erstellte Shell-Skript per Drag-and-Drop auf die Upload-Seite.
5. Klicken Sie auf Hochladen.

Das Skript wird in der Liste der Objekte angezeigt. Amazon S3 erstellt eine neue Versions-ID für die Datei. Wenn Sie das Skript aktualisieren und es erneut mit demselben Dateinamen hochladen, wird der Datei eine neue Versions-ID zugewiesen.

## AWS CLI

Um ein Shell-Skript (CLI) zu erstellen und hochzuladen

1. Öffnen Sie eine neue Eingabeaufforderung und führen Sie den Amazon S3 `ls` S3-Befehl aus, um den Bucket aufzulisten und zu identifizieren, der mit Ihrer Umgebung verknüpft ist.

```
$ aws s3 ls
```

2. Navigieren Sie zu dem Ordner, in dem Sie das Shell-Skript gespeichert haben. Verwenden Sie `cp` in einem neuen Eingabeaufforderungsfenster, um das Skript in Ihren Bucket hochzuladen. Ersetzen Sie *your-s3-bucket* durch Ihre Informationen.

```
$ aws s3 cp startup.sh s3://your-s3-bucket/startup.sh
```

Bei Erfolg gibt Amazon S3 den URL-Pfad zum Objekt aus:

```
upload: ./startup.sh to s3://your-s3-bucket/startup.sh
```

3. Verwenden Sie den folgenden Befehl, um die neueste Versions-ID für das Skript abzurufen.

```
$ aws s3api list-object-versions --bucket your-s3-bucket --prefix startup --query 'Versions[?IsLatest].[VersionId]' --output text
```

```
BbdVMmBRjtestta1EsVnbybZp1Wqh1J4
```

Sie geben diese Versions-ID an, wenn Sie das Skript einer Umgebung zuordnen.

Ordnen Sie das Skript nun Ihrer Umgebung zu.

## AWS Management Console

Um das Skript einer Umgebung (Konsole) zuzuordnen

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie die Zeile für die Umgebung aus, die Sie aktualisieren möchten, und klicken Sie dann auf Bearbeiten.
3. Geben Sie auf der Seite Details angeben für Startskriptdatei — optional die Amazon S3 S3-URL für das Skript ein, zum Beispiel:`s3://your-mwaa-bucket/startup-sh..`
4. Wählen Sie die neueste Version aus der Drop-down-Liste aus oder suchen Sie nach dem Skript, S3 um nach dem Skript zu suchen.
5. Wählen Sie Weiter und fahren Sie dann mit der Seite Überprüfen und speichern fort.
6. Überprüfen Sie die Änderungen und wählen Sie dann Speichern.

Aktualisierungen der Umgebung können zwischen 10 und 30 Minuten dauern. Amazon MWAA führt das Startskript aus, wenn jede Komponente in Ihrer Umgebung neu gestartet wird.

## AWS CLI

So verknüpfen Sie das Skript mit einer Umgebung (CLI)

- Öffnen Sie eine Befehlszeile und geben Sie `update-environment` damit die Amazon S3 S3-URL und die Versions-ID für das Skript an.

```
$ aws mwaa update-environment \
 --name your-mwaa-environment \
 --startup-script-s3-path startup.sh \
 --version-id BbdVMmBRjtestta1EsVnbybZp1Wqh1J4
```

```
--startup-script-s3-object-version BbdVMmBRjtestta1EsVnbybZp1Wqh1J4
```

Bei Erfolg gibt Amazon MWAA den Amazon Resource Name (ARN) für die Umgebung zurück:

```
arn:aws::airflow:us-west-2:123456789012:environment/your-mwaa-environment
```

Die Aktualisierung der Umgebung kann zwischen 10 und 30 Minuten dauern. Amazon MWAA führt das Startskript aus, wenn jede Komponente in Ihrer Umgebung neu gestartet wird.

Rufen Sie abschließend Protokollereignisse ab, um zu überprüfen, ob das Skript wie erwartet funktioniert. Wenn Sie die Protokollierung für jede Apache Airflow-Komponente aktivieren, erstellt Amazon MWAA eine neue Protokollgruppe und einen neuen Protokollstream. Weitere Informationen finden Sie unter [Apache Airflow-Protokolltypen](#).

## AWS Management Console

So überprüfen Sie den Apache Airflow-Protokollstream (Konsole)

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie Ihre Umgebung aus.
3. Wählen Sie im Bereich Überwachung die Protokollgruppe aus, für die Sie Protokolle anzeigen möchten, z. B. die Airflow Scheduler-Protokollgruppe.
4. Wählen Sie in der CloudWatch Konsole aus der Liste Protokollstreams einen Stream mit dem folgenden Präfix aus: `startup_script_exection_ip`
5. Im Bereich Ereignisse protokollieren sehen Sie die Ausgabe des Befehls, der den Wert für `ausgibtMWAA_AIRFLOW_COMPONENT`. Bei Scheduler-Protokollen gehen Sie beispielsweise wie folgt vor:

```
Printing Apache Airflow component
scheduler
Finished running startup script. Execution time: 0.004s.
Running verification
Verification completed
```

Sie können die vorherigen Schritte wiederholen, um Worker- und Webserver-Protokolle anzuzeigen.

## Installieren Sie Linux-Laufzeiten mithilfe eines Startskripts

Verwenden Sie ein Startskript, um das Betriebssystem einer Apache Airflow-Komponente zu aktualisieren, und installieren Sie zusätzliche Laufzeitbibliotheken, die Sie mit Ihren Workflows verwenden können. Das folgende Skript wird beispielsweise ausgeführt, `yum update` um das Betriebssystem zu aktualisieren.

Bei `yum update` der Ausführung in einem Startskript müssen Sie Python ausschließen, indem Sie, `--exclude=python*` wie im Beispiel gezeigt, verwenden. Damit Ihre Umgebung ausgeführt werden kann, installiert Amazon MWAA eine bestimmte Version von Python, die mit Ihrer Umgebung kompatibel ist. Daher können Sie die Python-Version der Umgebung nicht mit einem Startskript aktualisieren.

```
#!/bin/sh

echo "Updating operating system"
sudo yum update -y --exclude=python*
```

Um Runtimes auf einer bestimmten Apache Airflow-Komponente zu installieren, verwenden Sie `MWAA_AIRFLOW_COMPONENT` und `if` und `fi` bedingte Anweisungen. In diesem Beispiel wird ein einziger Befehl ausgeführt, um die `libaio` Bibliothek auf dem Scheduler und Worker zu installieren, aber nicht auf dem Webserver.

### Important

- Wenn Sie einen [privaten Webserver](#) konfiguriert haben, müssen Sie entweder die folgende Bedingung verwenden oder alle Installationsdateien lokal bereitstellen, um Zeitüberschreitungen bei der Installation zu vermeiden.
- Wird verwendet `sudo`, um Operationen auszuführen, für die Administratorrechte erforderlich sind.

```
#!/bin/sh
```

```
if [["${MWA_AIRFLOW_COMPONENT}" != "webserver"]]
then
 sudo yum -y install libaio
fi
```

Sie können ein Startskript verwenden, um die Python-Version zu überprüfen.

```
#!/bin/sh

export PYTHON_VERSION_CHECK=`python -c 'import sys; version=sys.version_info[:3];
print("{0}.{1}.{2}".format(*version))'`
echo "Python version is $PYTHON_VERSION_CHECK"
```

Amazon MWAA unterstützt das Überschreiben der Standard-Python-Version nicht, da dies zu Inkompatibilitäten mit den installierten Apache Airflow-Bibliotheken führen kann.

## Legen Sie Umgebungsvariablen mithilfe eines Startskripts fest

Verwenden Sie Startskripts, um Umgebungsvariablen festzulegen und Apache Airflow-Konfigurationen zu ändern. Im Folgenden wird eine neue Variable definiert, `ENVIRONMENT_STAGE`. Sie können in einer DAG oder in Ihren benutzerdefinierten Modulen auf diese Variable verweisen.

```
#!/bin/sh

export ENVIRONMENT_STAGE="development"
echo "$ENVIRONMENT_STAGE"
```

Verwenden Sie Startskripts, um gängige Apache Airflow- oder Systemvariablen zu überschreiben. Sie legen beispielsweise fest, `LD_LIBRARY_PATH` dass Python in dem von Ihnen angegebenen Pfad nach Binärdateien suchen soll. [Auf diese Weise können Sie mithilfe von Plugins benutzerdefinierte Binärdateien für Ihre Workflows bereitstellen:](#)

```
#!/bin/sh

export LD_LIBRARY_PATH=/usr/local/airflow/plugins/your-custom-binary
```

## Reservierte Umgebungsvariablen

Amazon MWAA reserviert eine Reihe kritischer Umgebungsvariablen. Wenn Sie eine reservierte Variable überschreiben, setzt Amazon MWAA sie auf ihre Standardwerte zurück. Im Folgenden sind die reservierten Variablen aufgeführt:

- `MWAA__AIRFLOW__COMPONENT`— Wird verwendet, um die Apache Airflow-Komponente mit einem der folgenden Werte zu identifizieren: `scheduler`, `worker`, oder `webserver`.
- `AIRFLOW__WEBSERVER__SECRET_KEY`— Der geheime Schlüssel, der zum sicheren Signieren von Sitzungscookies auf dem Apache Airflow-Webserver verwendet wird.
- `AIRFLOW__CORE__FERNET_KEY`— Der Schlüssel, der für die Verschlüsselung und Entschlüsselung sensibler Daten verwendet wird, die in der Metadaten-Datenbank gespeichert sind, z. B. Verbindungskennwörter.
- `AIRFLOW_HOME`— Der Pfad zum Apache Airflow-Home-Verzeichnis, in dem Konfigurationsdateien und DAG-Dateien lokal gespeichert werden.
- `AIRFLOW__CELERY__BROKER_URL`— Die URL des Message Brokers, der für die Kommunikation zwischen dem Apache Airflow Scheduler und den Celery Worker Nodes verwendet wird.
- `AIRFLOW__CELERY__RESULT_BACKEND`— Die URL der Datenbank, die zum Speichern der Ergebnisse von Celery-Aufgaben verwendet wird.
- `AIRFLOW__CORE__EXECUTOR`— Die Executor-Klasse, die Apache Airflow verwenden soll. In Amazon MWAA ist dies ein `CeleryExecutor`.
- `AIRFLOW__CORE__LOAD_EXAMPLES`— Wird verwendet, um das Laden von Beispiel-DAGs zu aktivieren oder zu deaktivieren.
- `AIRFLOW__METRICS__METRICS_BLOCK_LIST`— Wird verwendet, um zu verwalten, welche Apache Airflow-Metriken von Amazon MWAA ausgegeben und erfasst werden. CloudWatch
- `SQL_ALCHEMY_CONN`— Die Verbindungszeichenfolge für die Datenbank RDS for PostgreSQL, die zum Speichern von Apache Airflow-Metadaten in Amazon MWAA verwendet wird.
- `AIRFLOW__CORE__SQL_ALCHEMY_CONN`— Wird für denselben Zweck verwendet wie `SQL_ALCHEMY_CONN`, folgt jedoch der neuen Apache Airflow-Namenskonvention.
- `AIRFLOW__CELERY__DEFAULT_QUEUE`— Die Standardwarteschlange für Celery-Aufgaben in Apache Airflow.
- `AIRFLOW__OPERATORS__DEFAULT_QUEUE`— Die Standardwarteschlange für Aufgaben, die bestimmte Apache Airflow-Operatoren verwenden.
- `AIRFLOW_VERSION`— Die in der Amazon MWAA-Umgebung installierte Apache Airflow-Version.

- `AIRFLOW_CONN_AWS_DEFAULT`— Die AWS Standardanmeldedaten, die für die Integration mit anderen AWS Diensten in verwendet werden.
- `AWS_DEFAULT_REGION`— Legt die AWS Standardregion fest, die mit Standardanmeldedaten für die Integration mit anderen AWS Diensten verwendet wird.
- `AWS_REGION`— Falls definiert, überschreibt diese Umgebungsvariable die Werte in der Umgebungsvariablen `AWS_DEFAULT_REGION` und im Bereich der Profileinstellungen.
- `PYTHONUNBUFFERED`— Wird zum Senden `stdout` und `stderr` Streamen von Container-Logs verwendet.
- `AIRFLOW__METRICS__STATSD_ALLOW_LIST`— Wird verwendet, um eine Zulassungsliste mit kommasetrennten Präfixen zu konfigurieren, um die Metriken zu senden, die mit den Elementen der Liste beginnen.
- `AIRFLOW__METRICS__STATSD_ON`— Aktiviert das Senden von Metriken an `StatsD`.
- `AIRFLOW__METRICS__STATSD_HOST`— Wird verwendet, um eine Verbindung zum `StatSD` Daemon herzustellen.
- `AIRFLOW__METRICS__STATSD_PORT`— Wird verwendet, um eine Verbindung zum `StatSD` Daemon herzustellen.
- `AIRFLOW__METRICS__STATSD_PREFIX`— Wird verwendet, um eine Verbindung zum `StatSD` Daemon herzustellen.
- `AIRFLOW__CELERY__WORKER_AUTOSCALE`— Legt die maximale und minimale Parallelität fest.
- `AIRFLOW__CORE__DAG_CONCURRENCY`— Legt die Anzahl der Task-Instanzen fest, die vom Scheduler gleichzeitig in einer DAG ausgeführt werden können.
- `AIRFLOW__CORE__MAX_ACTIVE_TASKS_PER_DAG`— Legt die maximale Anzahl aktiver Aufgaben pro DAG fest.
- `AIRFLOW__CORE__PARALLELISM`— Definiert die maximale Anzahl von Task-Instanzen, die gleichzeitig ausgeführt werden können.
- `AIRFLOW__SCHEDULER__PARSING_PROCESSES`— Legt die maximale Anzahl von Prozessen fest, die vom Scheduler analysiert werden, um DAGs zu planen.
- `AIRFLOW__CELERY_BROKER_TRANSPORT_OPTIONS__VISIBILITY_TIMEOUT`— Definiert die Anzahl von Sekunden, die ein Worker auf die Bestätigung der Aufgabe wartet, bevor die Nachricht erneut an einen anderen Worker zugestellt wird.
- `AIRFLOW__CELERY_BROKER_TRANSPORT_OPTIONS__REGION`— Legt die AWS Region für den zugrunde liegenden Celerietransport fest.

- `AIRFLOW__CELERY_BROKER_TRANSPORT_OPTIONS__PREDEFINED_QUEUES`— Legt die Warteschlange für den zugrunde liegenden Celerietransport fest.
- `AIRFLOW_SCHEDULER_ALLOWED_RUN_ID_PATTERN`— Wird verwendet, um die Gültigkeit Ihrer Eingabe für den `run_id` Parameter beim Auslösen einer DAG zu überprüfen.
- `AIRFLOW__WEBSERVER__BASE_URL`— Die URL des Webservers, auf dem die Apache Airflow-Benutzeroberfläche gehostet wird.

## Nicht reservierte Umgebungsvariablen

Sie können ein Startskript verwenden, um nicht reservierte Umgebungsvariablen zu überschreiben. Im Folgenden sind einige dieser allgemeinen Variablen aufgeführt:

- `PATH`— Gibt eine Liste von Verzeichnissen an, in denen das Betriebssystem nach ausführbaren Dateien und Skripten sucht. Wenn ein Befehl in der Befehlszeile ausgeführt wird, überprüft das System die Verzeichnisse, um den Befehl zu finden und auszuführen. `PATH` Wenn Sie benutzerdefinierte Operatoren oder Aufgaben in Apache Airflow erstellen, müssen Sie sich möglicherweise auf externe Skripts oder ausführbare Dateien verlassen. Wenn sich die Verzeichnisse, die diese Dateien enthalten, nicht in den in der `PATH` Variablen angegebenen Verzeichnissen befinden, können die Aufgaben nicht ausgeführt werden, wenn das System sie nicht finden kann. Durch Hinzufügen der entsprechenden Verzeichnisse können Apache Airflow-Aufgaben die erforderlichen ausführbaren Dateien finden und ausführen. `PATH`
- `PYTHONPATH`— Wird vom Python-Interpreter verwendet, um zu bestimmen, in welchen Verzeichnissen nach importierten Modulen und Paketen gesucht werden soll. Es ist eine Liste von Verzeichnissen, die Sie dem Standardsuchpfad hinzufügen können. Dadurch kann der Interpreter Python-Bibliotheken finden und laden, die nicht in der Standardbibliothek enthalten oder in Systemverzeichnissen installiert sind. Verwenden Sie diese Variable, um Ihre Module und benutzerdefinierten Python-Pakete hinzuzufügen und sie mit Ihren DAGs zu verwenden.
- `LD_LIBRARY_PATH`— Eine Umgebungsvariable, die vom dynamischen Linker und Loader unter Linux verwendet wird, um gemeinsam genutzte Bibliotheken zu finden und zu laden. Sie spezifiziert eine Liste von Verzeichnissen, die gemeinsam genutzte Bibliotheken enthalten, die vor den Standardverzeichnissen der Systembibliotheken durchsucht werden. Verwenden Sie diese Variable, um Ihre benutzerdefinierten Binärdateien anzugeben.
- `CLASSPATH`— Wird vom Java Runtime Environment (JRE) und dem Java Development Kit (JDK) verwendet, um Java-Klassen, -Bibliotheken und -Ressourcen zur Laufzeit zu finden und zu laden. Es ist eine Liste von Verzeichnissen, JAR-Dateien und ZIP-Archiven, die kompilierten Java-Code enthalten.



# Arbeiten mit DAGs auf Amazon MWAA

Um Directed Acyclic Graphs (DAGs) in einer Amazon Managed Workflows for Apache Airflow-Umgebung auszuführen, kopieren Sie Ihre Dateien in den Amazon S3 S3-Speicher-Bucket, der mit Ihrer Umgebung verbunden ist, und teilen Amazon MWAA dann mit, wo sich Ihre DAGs und unterstützenden Dateien auf der Amazon MWAA-Konsole befinden. Amazon MWAA kümmert sich um die Synchronisation der DAGs zwischen Workern, Schemulern und dem Webserver. In diesem Handbuch wird beschrieben, wie Sie Ihre DAGs hinzufügen oder aktualisieren und benutzerdefinierte Plugins und Python-Abhängigkeiten in einer Amazon MWAA-Umgebung installieren.

## Themen

- [Überblick über den Amazon S3 S3-Bucket](#)
- [Hinzufügen oder Aktualisieren von DAGs](#)
- [Installation benutzerdefinierter Plugins](#)
- [Python-Abhängigkeiten installieren](#)
- [Löschen von Dateien auf Amazon S3](#)

## Überblick über den Amazon S3 S3-Bucket

Bei einem Amazon S3 S3-Bucket für eine Amazon MWAA-Umgebung muss der öffentliche Zugriff gesperrt sein. Standardmäßig sind alle Amazon S3 S3-Ressourcen — Buckets, Objekte und zugehörige Unterressourcen (z. B. Lebenszykluskonfiguration) — privat.

- Nur der Eigentümer der Ressource, das AWS Konto, das den Bucket erstellt hat, kann auf die Ressource zugreifen. Der Ressourcenbesitzer (z. B. Ihr Administrator) kann anderen Zugriffsberechtigungen gewähren, indem er eine Zugriffskontrollrichtlinie verfasst.
- Die von Ihnen eingerichtete Zugriffsrichtlinie muss berechtigt sein, DAGs, benutzerdefinierte Plugins `plugins.zip` und Python-Abhängigkeiten `requirements.txt` zu Ihrem Amazon S3 S3-Bucket hinzuzufügen. Eine Beispielrichtlinie, die die erforderlichen Berechtigungen enthält, finden Sie unter [FullConsoleAccessAmazonMWAA](#).

Für einen Amazon S3 S3-Bucket für eine Amazon MWAA-Umgebung muss Versioning aktiviert sein. Wenn die Amazon S3 S3-Bucket-Versionierung aktiviert ist, wird bei jeder Erstellung einer neuen Version eine neue Kopie erstellt.

- Die Versionierung ist für die benutzerdefinierten Plug-ins in a `plugins.zip` und die Python-Abhängigkeiten in a in Ihrem `requirements.txt` Amazon S3 S3-Bucket aktiviert.
- Sie müssen jedes Mal `plugins.zip`, wenn diese Dateien in Ihrem Amazon S3-Bucket aktualisiert werden, die Version von und `requirements.txt` auf der Amazon MWAA-Konsole angeben.

## Hinzufügen oder Aktualisieren von DAGs

Directed Acyclic Graphs (DAGs) werden in einer Python-Datei definiert, die die Struktur der DAG als Code definiert. Sie können die oder die Amazon S3 S3-Konsole verwenden AWS CLI, um DAGs in Ihre Umgebung hochzuladen. Auf dieser Seite werden die Schritte beschrieben, um Apache Airflow-DAGs in Ihrer Amazon Managed Workflows for Apache Airflow-Umgebung mithilfe des `aws` Ordners in Ihrem Amazon S3 S3-Bucket hinzuzufügen oder zu aktualisieren.

### Abschnitte

- [Voraussetzungen](#)
- [Funktionsweise](#)
- [Was hat sich in Version 2 geändert](#)
- [Testen von DAGs mit dem Amazon MWAA CLI Utility](#)
- [Hochladen von DAG-Code in Amazon S3](#)
- [Angabe des Pfads zu Ihrem DAG-Ordner auf der Amazon MWAA-Konsole \(das erste Mal\)](#)
- [Anzeigen von Änderungen in Ihrer Apache Airflow-Benutzeroberfläche](#)
- [Als nächstes](#)

## Voraussetzungen

Sie benötigen Folgendes, um die Schritte auf dieser Seite ausführen zu können.

- Berechtigungen — Ihrem AWS Konto muss von Ihrem Administrator Zugriff auf die [FullConsoleAccessAmazonMWAA-Zugriffskontrollrichtlinie](#) für Ihre Umgebung gewährt worden sein. Darüber hinaus muss Ihrer Amazon MWAA-Umgebung von Ihrer [Ausführungsrolle](#) der Zugriff auf die von Ihrer Umgebung verwendeten AWS Ressourcen gestattet sein.
- Zugriff — Wenn Sie Zugriff auf öffentliche Repositorys benötigen, um Abhängigkeiten direkt auf dem Webserver zu installieren, muss Ihre Umgebung für den Zugriff auf öffentliche Netzwerkwebserver konfiguriert sein. Weitere Informationen finden Sie unter [the section called "Apache Airflow-Zugriffsmodi"](#).

- Amazon S3 S3-Konfiguration — Der [Amazon S3 S3-Bucket](#), in dem Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten gespeichert werden `plugins.zip,requirements.txt` muss so konfiguriert sein, dass der öffentliche Zugriff blockiert und die Versionierung aktiviert ist.

## Funktionsweise

Ein Directed Acyclic Graph (DAG) wird in einer einzigen Python-Datei definiert, die die Struktur der DAG als Code definiert. Sie besteht aus Folgendem:

- Eine [DAG-Definition](#).
- [Operatoren](#), die beschreiben, wie die DAG ausgeführt wird und [welche Aufgaben](#) ausgeführt werden.
- [Operatorbeziehungen](#), die die Reihenfolge beschreiben, in der die Aufgaben ausgeführt werden.

Um eine Apache Airflow-Plattform in einer Amazon MWAA-Umgebung auszuführen, müssen Sie Ihre DAG-Definition in den `dags` Ordner in Ihrem Speicher-Bucket kopieren. Der DAG-Ordner in Ihrem Speicher-Bucket könnte beispielsweise so aussehen:

Example DAG-Ordner

```
dags/
dag_def.py
```

Amazon MWAA synchronisiert automatisch alle 30 Sekunden neue und geänderte Objekte aus Ihrem Amazon S3 S3-Bucket mit dem Amazon MWAA-Scheduler und dem `/usr/local/airflow/dags` Ordner der Worker-Container, wobei die Dateihierarchie der Amazon S3 S3-Quelle unabhängig vom Dateityp beibehalten wird. Die Zeit, die neue DAGs benötigen, um in Ihrer Apache Airflow-Benutzeroberfläche zu erscheinen, wird von gesteuert [scheduler.dag\\_dir\\_list\\_interval](#). Änderungen an bestehenden DAGs werden in der nächsten [DAG-Verarbeitungsschleife](#) übernommen.

### Note

Sie müssen die `airflow.cfg` Konfigurationsdatei nicht in Ihren DAG-Ordner aufnehmen. Sie können die standardmäßigen Apache Airflow-Konfigurationen von der Amazon MWAA-Konsole aus überschreiben. Weitere Informationen finden Sie unter [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#).

## Was hat sich in Version 2 geändert

- Neu: Operatoren, Hooks und Executors. Die Importanweisungen in Ihren DAGs und die benutzerdefinierten Plugins, die Sie in einer `plugins.zip` Amazon MWAA angeben, haben sich zwischen Apache Airflow v1 und Apache Airflow v2 geändert. Zum Beispiel wurde `from airflow.contrib.hooks.aws_hook import AwsHook` in Apache Airflow v1 durch `from airflow.providers.amazon.aws.hooks.base_aws import AwsBaseHook` in Apache Airflow v2 geändert. Weitere Informationen finden Sie unter [Python-API-Referenz](#) im Apache Airflow-Referenzhandbuch.

## Testen von DAGs mit dem Amazon MWAA CLI Utility

- Das Befehlszeileninterface (CLI) repliziert eine Amazon Managed-Workflows-for-Apache-Airflow-Umgebung lokal.
- Die CLI erstellt lokal ein Docker-Container-Image, das einem Amazon MWAA-Produktionsimage ähnelt. Auf diese Weise können Sie eine lokale Apache Airflow-Umgebung ausführen, um DAGs, benutzerdefinierte Plugins und Abhängigkeiten zu entwickeln und zu testen, bevor Sie sie auf Amazon MWAA bereitstellen.
- Informationen zum Ausführen der CLI finden Sie [aws-mwaa-local-runner](#) unter GitHub.

## Hochladen von DAG-Code in Amazon S3

Sie können die Amazon S3 S3-Konsole oder die AWS Command Line Interface (AWS CLI) verwenden, um DAG-Code in Ihren Amazon S3 S3-Bucket hochzuladen. Bei den folgenden Schritten wird davon ausgegangen, dass Sie Code ( `.py` ) in einen Ordner hochladen, der tags in Ihrem Amazon S3 S3-Bucket benannt ist.

### Verwendung der AWS CLI

Die AWS Command Line Interface (AWS CLI) ist ein Open-Source-Tool, mit dem Sie über Befehle in Ihrer Befehlszeilen-Shell mit den AWS-Services interagieren können. Um die Schritte auf dieser Seite abzuschließen, benötigen Sie Folgendes:

- [AWS CLI— Installieren Sie Version 2.](#)
- [AWS CLI— Schnelle Konfiguration mit `aws configure`.](#)

## Zum Hochladen mit dem AWS CLI

1. Verwenden Sie den folgenden Befehl, um alle Ihre Amazon S3 S3-Buckets aufzulisten.

```
aws s3 ls
```

2. Verwenden Sie den folgenden Befehl, um die Dateien und Ordner im Amazon S3 S3-Bucket für Ihre Umgebung aufzulisten.

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

3. Der folgende Befehl lädt ein `dag_def.py` Datei in ein `days` Ordner hoch.

```
aws s3 cp dag_def.py s3://YOUR_S3_BUCKET_NAME/days/
```

Wenn ein benannter Ordner in Ihrem Amazon S3 S3-Bucket noch nicht `days` existiert, erstellt dieser Befehl den `days` Ordner und lädt die benannte Datei `dag_def.py` in den neuen Ordner hoch.

## Verwenden der Amazon S3-Konsole

Die Amazon S3 S3-Konsole ist eine webbasierte Benutzeroberfläche, mit der Sie die Ressourcen in Ihrem Amazon S3 S3-Bucket erstellen und verwalten können. Die folgenden Schritte setzen voraus, dass Sie einen DAG-Ordner benannt `days`.

So uploaden Sie mit der Amazon S3 S3-Konsole

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung.
3. Wählen Sie den S3-Bucket-Link im DAG-Code im S3-Bereich aus, um Ihren Speicher-Bucket auf der Amazon S3 S3-Konsole zu öffnen.
4. Wählen Sie den Ordner `days` aus.
5. Klicken Sie auf Upload.
6. Wählen Sie Datei hinzufügen.
7. Wählen Sie die lokale Kopie Ihres `dag_def.py` und wählen Sie Hochladen.

## Angabe des Pfads zu Ihrem DAG-Ordner auf der Amazon MWAA-Konsole (das erste Mal)

Die folgenden Schritte gehen davon aus, dass Sie den Pfad zu einem Ordner in Ihrem Amazon S3 S3-Bucket mit dem Namen angehendags.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie die Umgebung aus, in der Sie DAGs ausführen möchten.
3. Wählen Sie Edit (Bearbeiten) aus.
4. Wählen Sie im Bereich DAG-Code im Bereich Amazon S3 neben dem Feld DAG-Ordner die Option S3 durchsuchen aus.
5. Wählen Sie Ihrendags Ordner aus.
6. Wählen Sie Choose (Auswählen) aus.
7. Wählen Sie Weiter, Umgebung aktualisieren.

## Anzeigen von Änderungen in Ihrer Apache Airflow-Benutzeroberfläche

### Bei Apache Airflow einloggen

Sie benötigen [Zugriffsrichtlinie für die Apache Airflow-Benutzeroberfläche: AmazonMWAA Access WebServer](#) Berechtigungen für Ihr AWS Konto in AWS Identity and Access Management (IAM), um Ihre Apache Airflow-Benutzeroberfläche anzeigen zu können.

So greifen Sie auf Ihre Apache Airflow-Benutzeroberfläche zu

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung.
3. Wählen Sie Open Airflow UI.

### Als nächstes

- Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.

# Installation benutzerdefinierter Plugins

Amazon Managed Workflows for Apache Airflow unterstützt den integrierten Plugin-Manager von Apache Airflow, sodass Sie benutzerdefinierte Apache Airflow-Operatoren, Hooks, Sensoren oder Schnittstellen verwenden können. Auf dieser Seite werden die Schritte zur Installation [benutzerdefinierter Apache Airflow-Plugins](#) in Ihrer Amazon MWAA-Umgebung mithilfe einer Datei beschrieben. `plugins.zip`

## Inhalt

- [Voraussetzungen](#)
- [Funktionsweise](#)
- [Was hat sich in Version 2 geändert](#)
- [Übersicht über benutzerdefinierte Plugins](#)
  - [Verzeichnis- und Größenbeschränkungen für benutzerdefinierte Plugins](#)
- [Beispiele für benutzerdefinierte Plugins](#)
  - [Beispiel für die Verwendung einer flachen Verzeichnisstruktur in `plugins.zip`](#)
  - [Beispiel für die Verwendung einer verschachtelten Verzeichnisstruktur in `plugins.zip`](#)
- [Eine Datei vom Typ `plugins.zip` erstellen](#)
  - [Schritt eins: Testen Sie benutzerdefinierte Plugins mit dem Amazon MWAA CLI-Hilfsprogramm](#)
  - [Schritt zwei: Erstellen Sie die Datei `plugins.zip`](#)
- [Auf Amazon `plugins.zip` S3 hochladen](#)
  - [Verwenden der AWS CLI](#)
  - [Verwenden der Amazon S3-Konsole](#)
- [Installation benutzerdefinierter Plugins in Ihrer Umgebung](#)
  - [Angabe des Pfads zu `plugins.zip` auf der Amazon MWAA-Konsole \(beim ersten Mal\)](#)
  - [Angabe der `plugins.zip` Version auf der Amazon MWAA-Konsole](#)
- [Beispielhafte Anwendungsfälle für `plugins.zip`](#)
- [Als nächstes](#)

## Voraussetzungen

Sie benötigen Folgendes, bevor Sie die Schritte auf dieser Seite abschließen können.

- Berechtigungen — Ihr AWS Konto muss von Ihrem Administrator Zugriff auf die [FullConsoleAccessAmazonMWAA-Zugriffskontrollrichtlinie](#) für Ihre Umgebung erhalten haben. Darüber hinaus muss Ihrer Amazon MWAA-Umgebung von Ihrer [Ausführungsrolle](#) der Zugriff auf die von Ihrer Umgebung verwendeten AWS Ressourcen gestattet werden.
- Zugriff — Wenn Sie Zugriff auf öffentliche Repositorys benötigen, um Abhängigkeiten direkt auf dem Webserver zu installieren, muss Ihre Umgebung für den Zugriff auf öffentliche Netzwerk-Webserver konfiguriert sein. Weitere Informationen finden Sie unter [the section called “Apache Airflow-Zugriffsmodi”](#).
- Amazon S3 S3-Konfiguration — Der [Amazon S3 S3-Bucket](#), der zum Speichern Ihrer DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten verwendet wird `plugins.zip`, `requirements.txt` muss mit geblocktem öffentlichen Zugriff und aktivierter Versionierung konfiguriert sein.

## Funktionsweise

Um benutzerdefinierte Plugins in Ihrer Umgebung auszuführen, müssen Sie drei Dinge tun:

1. Erstellen Sie lokal eine `plugins.zip` Datei.
2. Laden Sie die lokale `plugins.zip` Datei in Ihren Amazon S3 S3-Bucket hoch.
3. Geben Sie die Version dieser Datei im Feld Plugins-Datei auf der Amazon MWAA-Konsole an.

### Note

Wenn Sie zum ersten Mal einen `plugins.zip` in Ihren Amazon S3 S3-Bucket hochladen, müssen Sie auch den Pfad zu der Datei auf der Amazon MWAA-Konsole angeben. Sie müssen diesen Schritt nur einmal ausführen.

## Was hat sich in Version 2 geändert

- Neu: Operatoren, Hooks und Executors. Die Import-Anweisungen in Ihren DAGs und die benutzerdefinierten Plugins, die Sie in einem MWAA `plugins.zip` auf Amazon angeben, haben sich zwischen Apache Airflow v1 und Apache Airflow v2 geändert. Beispielsweise wurde `from airflow.contrib.hooks.aws_hook import AwsHook` in Apache Airflow v1 zu Apache Airflow v2 geändert. `from airflow.providers.amazon.aws.hooks.base_aws import`



`AwsBaseHook` Weitere Informationen finden Sie in der [Python-API-Referenz](#) im Apache Airflow-Referenzhandbuch.

- Neu: Importe in Plugins. Das Importieren von Operatoren, Sensoren und Hooks, die mithilfe von Plugins hinzugefügt wurden, `airflow.{operators,sensors,hooks}.<plugin_name>` wird nicht mehr unterstützt. Diese Erweiterungen sollten als reguläre Python-Module importiert werden. In Version 2 und höher besteht der empfohlene Ansatz darin, sie im DAG-Verzeichnis zu platzieren und eine `.airflowignore`-Datei zu erstellen und zu verwenden, um sie von der Analyse als DAGs auszuschließen. Weitere Informationen finden Sie unter [Modulverwaltung](#) und [Erstellen eines benutzerdefinierten Operators](#) im Apache Airflow-Referenzhandbuch.

## Übersicht über benutzerdefinierte Plugins

Der integrierte Plugin-Manager von Apache Airflow kann externe Funktionen in seinen Kern integrieren, indem er Dateien einfach in einem `$AIRFLOW_HOME/plugins` Ordner ablegt. Es ermöglicht Ihnen, benutzerdefinierte Apache Airflow-Operatoren, Hooks, Sensoren oder Schnittstellen zu verwenden. Der folgende Abschnitt enthält ein Beispiel für flache und verschachtelte Verzeichnisstrukturen in einer lokalen Entwicklungsumgebung und die daraus resultierenden Importanweisungen, die die Verzeichnisstruktur innerhalb einer `plugins.zip` bestimmen.

### Verzeichnis- und Größenbeschränkungen für benutzerdefinierte Plugins

Der Apache Airflow Scheduler und die Workers suchen beim Start auf dem AWS -verwalteten Fargate-Container für Ihre Umgebung unter nach benutzerdefinierten Plugins. `/usr/local/airflow/plugins/*`

- Verzeichnisstruktur. Die Verzeichnisstruktur (`at/*`) basiert auf dem Inhalt Ihrer `plugins.zip` Datei. Wenn Ihr Verzeichnis beispielsweise als `operators` Verzeichnis der obersten Ebene `plugins.zip` enthält, wird das Verzeichnis in Ihre Umgebung extrahiert. `/usr/local/airflow/plugins/operators`
- Größenbeschränkung. Wir empfehlen eine `plugins.zip` Datei mit weniger als 1 GB. Je größer eine `plugins.zip` Datei, desto länger ist die Startzeit in einer Umgebung. Amazon MWAA begrenzt die Größe einer `plugins.zip` Datei zwar nicht explizit, aber wenn Abhängigkeiten nicht innerhalb von zehn Minuten installiert werden können, führt der Fargate-Service zu einem Timeout und versucht, die Umgebung auf einen stabilen Zustand zurückzusetzen.

**Note**

Für Umgebungen, die Apache Airflow v1.10.12 oder Apache Airflow v2.0.2 verwenden, begrenzt Amazon MWAA den ausgehenden Datenverkehr auf dem Apache Airflow-Webserver und erlaubt Ihnen nicht, Plugins oder Python-Abhängigkeiten direkt auf dem Webserver zu installieren. Ab Apache Airflow v2.2.2 kann Amazon MWAA Plugins und Abhängigkeiten direkt auf dem Webserver installieren.

## Beispiele für benutzerdefinierte Plugins

Im folgenden Abschnitt wird anhand von Beispielcode aus dem Apache Airflow-Referenzhandbuch gezeigt, wie Sie Ihre lokale Entwicklungsumgebung strukturieren können.

### Beispiel für die Verwendung einer flachen Verzeichnisstruktur in plugins.zip

#### Apache Airflow v2

Das folgende Beispiel zeigt eine `plugins.zip` Datei mit einer flachen Verzeichnisstruktur für Apache Airflow v2.

Example flaches Verzeichnis mit `plugins.zip` PythonVirtualenvOperator

Das folgende Beispiel zeigt die oberste Baumstruktur einer Datei `plugins.zip` für das PythonVirtualenvOperator benutzerdefinierte Plugin in [Ein benutzerdefiniertes Plugin für Apache Airflow erstellen](#) PythonVirtualenvOperator.

```
virtual_python_plugin.py
```

Example `plugins/virtual_python_plugin.py`

Das folgende Beispiel zeigt das PythonVirtualenvOperator benutzerdefinierte Plugin.

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

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

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

```
from airflow.plugins_manager import AirflowPlugin
import airflow.utils.python_virtualenv
from typing import List

def _generate_virtualenv_cmd(tmp_dir: str, python_bin: str, system_site_packages:
bool) -> List[str]:
 cmd = ['python3', '/usr/local/airflow/.local/lib/python3.7/site-packages/
virtualenv', tmp_dir]
 if system_site_packages:
 cmd.append('--system-site-packages')
 if python_bin is not None:
 cmd.append(f'--python={python_bin}')
 return cmd

airflow.utils.python_virtualenv._generate_virtualenv_cmd=_generate_virtualenv_cmd

class VirtualPythonPlugin(AirflowPlugin):
 name = 'virtual_python_plugin'
```

## Apache Airflow v1

Das folgende Beispiel zeigt eine `plugins.zip` Datei mit einer flachen Verzeichnisstruktur für Apache Airflow v1.

Example flaches Verzeichnis mit `plugins.zip` PythonVirtualenvOperator

Das folgende Beispiel zeigt die oberste Baumstruktur einer Datei `plugins.zip` für das PythonVirtualenvOperator benutzerdefinierte Plugin in [Ein benutzerdefiniertes Plugin für Apache Airflow erstellen](#) PythonVirtualenvOperator.

```
virtual_python_plugin.py
```

Example `plugins/virtual_python_plugin.py`

Das folgende Beispiel zeigt das PythonVirtualenvOperator benutzerdefinierte Plugin.

```

from airflow.plugins_manager import AirflowPlugin
from airflow.operators.python_operator import PythonVirtualenvOperator

def _generate_virtualenv_cmd(self, tmp_dir):
 cmd = ['python3', '/usr/local/airflow/.local/lib/python3.7/site-packages/
virtualenv', tmp_dir]
 if self.system_site_packages:
 cmd.append('--system-site-packages')
 if self.python_version is not None:
 cmd.append('--python=python{}'.format(self.python_version))
 return cmd
PythonVirtualenvOperator._generate_virtualenv_cmd=_generate_virtualenv_cmd

class EnvVarPlugin(AirflowPlugin):
 name = 'virtual_python_plugin'

```

## Beispiel für die Verwendung einer verschachtelten Verzeichnisstruktur in plugins.zip

### Apache Airflow v2

Das folgende Beispiel zeigt eine `plugins.zip` Datei mit separaten Verzeichnissen für `hooksoperators`, und einem `sensors` Verzeichnis für Apache Airflow v2.

### Example plugins.zip

```

__init__.py
my_airflow_plugin.py
hooks/
|-- __init__.py
|-- my_airflow_hook.py
operators/
|-- __init__.py
|-- my_airflow_operator.py
|-- hello_operator.py
sensors/
|-- __init__.py
|-- my_airflow_sensor.py

```

Das folgende Beispiel zeigt die Import-Anweisungen in der DAG ([DAGs-Ordner](#)), die die benutzerdefinierten Plugins verwendet.

## Example dags/your\_dag.py

```
from airflow import DAG
from datetime import datetime, timedelta
from operators.my_airflow_operator import MyOperator
from sensors.my_airflow_sensor import MySensor
from operators.hello_operator import HelloOperator

default_args = {
 'owner': 'airflow',
 'depends_on_past': False,
 'start_date': datetime(2018, 1, 1),
 'email_on_failure': False,
 'email_on_retry': False,
 'retries': 1,
 'retry_delay': timedelta(minutes=5),
}

with DAG('customdag',
 max_active_runs=3,
 schedule_interval='@once',
 default_args=default_args) as dag:

 sens = MySensor(
 task_id='taskA'
)

 op = MyOperator(
 task_id='taskB',
 my_field='some text'
)

 hello_task = HelloOperator(task_id='sample-task', name='foo_bar')

 sens >> op >> hello_task
```

## Example plugins/my\_airflow\_plugin.py

```
from airflow.plugins_manager import AirflowPlugin
from hooks.my_airflow_hook import *
```

```
from operators.my_airflow_operator import *

class PluginName(AirflowPlugin):

 name = 'my_airflow_plugin'

 hooks = [MyHook]
 operators = [MyOperator]
 sensors = [MySensor]
```

Die folgenden Beispiele zeigen die einzelnen Importanweisungen, die in den benutzerdefinierten Plugin-Dateien benötigt werden.

#### Example hooks/my\_airflow\_hook.py

```
from airflow.hooks.base import BaseHook

class MyHook(BaseHook):

 def my_method(self):
 print("Hello World")
```

#### Example sensors/my\_airflow\_sensor.py

```
from airflow.sensors.base import BaseSensorOperator
from airflow.utils.decorators import apply_defaults

class MySensor(BaseSensorOperator):

 @apply_defaults
 def __init__(self,
 *args,
 **kwargs):
 super(MySensor, self).__init__(*args, **kwargs)

 def poke(self, context):
 return True
```

## Example operators/my\_airflow\_operator.py

```
from airflow.operators.bash import BaseOperator
from airflow.utils.decorators import apply_defaults
from hooks.my_airflow_hook import MyHook

class MyOperator(BaseOperator):

 @apply_defaults
 def __init__(self,
 my_field,
 *args,
 **kwargs):
 super(MyOperator, self).__init__(*args, **kwargs)
 self.my_field = my_field

 def execute(self, context):
 hook = MyHook('my_conn')
 hook.my_method()
```

## Example operators/hello\_operator.py

```
from airflow.models.baseoperator import BaseOperator
from airflow.utils.decorators import apply_defaults

class HelloOperator(BaseOperator):

 @apply_defaults
 def __init__(
 self,
 name: str,
 **kwargs) -> None:
 super().__init__(**kwargs)
 self.name = name

 def execute(self, context):
 message = "Hello {}".format(self.name)
 print(message)
 return message
```

Folgen Sie den Schritten unter [Testen benutzerdefinierter Plug-ins mit dem Amazon MWAA-CLI-Hilfsprogramm](#) und dann [Erstellen einer Datei plugins.zip](#), um den Inhalt in Ihrem plugins Verzeichnis zu komprimieren. Beispiel: `cd plugins`

## Apache Airflow v1

Das folgende Beispiel zeigt eine `plugins.zip` Datei mit separaten Verzeichnissen für `hooksoperators`, und einem `sensors` Verzeichnis für Apache Airflow v1.10.12.

### Example plugins.zip

```
__init__.py
my_airflow_plugin.py
hooks/
 |-- __init__.py
 |-- my_airflow_hook.py
operators/
 |-- __init__.py
 |-- my_airflow_operator.py
 |-- hello_operator.py
sensors/
 |-- __init__.py
 |-- my_airflow_sensor.py
```

Das folgende Beispiel zeigt die Import-Anweisungen in der DAG ([DAGs-Ordner](#)), die die benutzerdefinierten Plugins verwendet.

### Example dags/your\_dag.py

```
from airflow import DAG
from datetime import datetime, timedelta
from operators.my_operator import MyOperator
from sensors.my_sensor import MySensor
from operators.hello_operator import HelloOperator

default_args = {
 'owner': 'airflow',
 'depends_on_past': False,
 'start_date': datetime(2018, 1, 1),
 'email_on_failure': False,
 'email_on_retry': False,
 'retries': 1,
```



```
'retry_delay': timedelta(minutes=5),
}

with DAG('customdag',
 max_active_runs=3,
 schedule_interval='@once',
 default_args=default_args) as dag:

 sens = MySensor(
 task_id='taskA'
)

 op = MyOperator(
 task_id='taskB',
 my_field='some text'
)

 hello_task = HelloOperator(task_id='sample-task', name='foo_bar')

sens >> op >> hello_task
```

### Example plugins/my\_airflow\_plugin.py

```
from airflow.plugins_manager import AirflowPlugin
from hooks.my_airflow_hook import *
from operators.my_airflow_operator import *
from utils.my_utils import *

class PluginName(AirflowPlugin):

 name = 'my_airflow_plugin'

 hooks = [MyHook]
 operators = [MyOperator]
 sensors = [MySensor]
```

Die folgenden Beispiele zeigen die einzelnen Importanweisungen, die in den benutzerdefinierten Plugin-Dateien benötigt werden.

## Example hooks/my\_airflow\_hook.py

```
from airflow.hooks.base_hook import BaseHook

class MyHook(BaseHook):

 def my_method(self):
 print("Hello World")
```

## Example sensors/my\_airflow\_sensor.py

```
from airflow.sensors.base_sensor_operator import BaseSensorOperator
from airflow.utils.decorators import apply_defaults

class MySensor(BaseSensorOperator):

 @apply_defaults
 def __init__(self,
 *args,
 **kwargs):
 super(MySensor, self).__init__(*args, **kwargs)

 def poke(self, context):
 return True
```

## Example operators/my\_airflow\_operator.py

```
from airflow.operators.bash_operator import BaseOperator
from airflow.utils.decorators import apply_defaults
from hooks.my_hook import MyHook

class MyOperator(BaseOperator):

 @apply_defaults
 def __init__(self,
 my_field,
 *args,
 **kwargs):
 super(MyOperator, self).__init__(*args, **kwargs)
```

```
self.my_field = my_field

def execute(self, context):
 hook = MyHook('my_conn')
 hook.my_method()
```

### Example operators/hello\_operator.py

```
from airflow.models.baseoperator import BaseOperator
from airflow.utils.decorators import apply_defaults

class HelloOperator(BaseOperator):

 @apply_defaults
 def __init__(
 self,
 name: str,
 **kwargs) -> None:
 super().__init__(**kwargs)
 self.name = name

 def execute(self, context):
 message = "Hello {}".format(self.name)
 print(message)
 return message
```

Folgen Sie den Schritten unter [Testen benutzerdefinierter Plug-ins mit dem Amazon MWAA-CLI-Hilfsprogramm](#) und dann [Erstellen einer Datei plugins.zip](#), um den Inhalt in Ihrem plugins Verzeichnis zu komprimieren. Beispiel: `cd plugins`

## Eine Datei vom Typ plugins.zip erstellen

In den folgenden Schritten werden die Schritte beschrieben, die wir empfehlen, um eine Datei plugins.zip lokal zu erstellen.

### Schritt eins: Testen Sie benutzerdefinierte Plugins mit dem Amazon MWAA CLI-Hilfsprogramm

- Das Befehlszeilenschnittstellenprogramm (CLI) repliziert eine Amazon Managed Workflows for Apache Airflow-Umgebung lokal.

- Die CLI erstellt lokal ein Docker-Container-Image, das einem Amazon MWAA-Produktionsimage ähnelt. Auf diese Weise können Sie eine lokale Apache Airflow-Umgebung ausführen, um DAGs, benutzerdefinierte Plugins und Abhängigkeiten zu entwickeln und zu testen, bevor Sie sie auf Amazon MWAA bereitstellen.
- Informationen zum Ausführen der CLI finden Sie [aws-mwaa-local-runner](#) unter GitHub.

## Schritt zwei: Erstellen Sie die Datei `plugins.zip`

Sie können ein integriertes ZIP-Archivierungsprogramm oder ein anderes ZIP-Hilfsprogramm (z. B. [7zip](#)) verwenden, um eine ZIP-Datei zu erstellen.

### Note

Das integrierte ZIP-Hilfsprogramm für Windows OS fügt möglicherweise Unterordner hinzu, wenn Sie eine ZIP-Datei erstellen. Wir empfehlen, den Inhalt der Datei `plugins.zip` vor dem Hochladen in Ihren Amazon S3 S3-Bucket zu überprüfen, um sicherzustellen, dass keine zusätzlichen Verzeichnisse hinzugefügt wurden.

1. Wechseln Sie zu den Verzeichnissen in Ihr lokales Airflow-Plugin-Verzeichnis. Beispiel:

```
myproject$ cd plugins
```

2. Führen Sie den folgenden Befehl aus, um sicherzustellen, dass der Inhalt über Ausführungsberechtigungen verfügt (nur macOS und Linux).

```
plugins$ chmod -R 755 .
```

3. Komprimieren Sie den Inhalt Ihres `plugins` Ordners.

```
plugins$ zip -r plugins.zip .
```

## Auf Amazon `plugins.zip` S3 hochladen

Sie können die Amazon S3 S3-Konsole oder die AWS Command Line Interface (AWS CLI) verwenden, um eine `plugins.zip` Datei in Ihren Amazon S3 S3-Bucket hochzuladen.

## Verwenden der AWS CLI

Die AWS Command Line Interface (AWS CLI) ist ein Open-Source-Tool, mit dem Sie über Befehle in Ihrer Befehlszeilen-Shell mit den AWS-Services interagieren können. Um die Schritte auf dieser Seite abzuschließen, benötigen Sie Folgendes:

- [AWS CLI— Installieren Sie Version 2.](#)
- [AWS CLI— Schnelle Konfiguration mit `aws configure`.](#)

### Zum Hochladen mit dem AWS CLI

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihre `plugins.zip` Datei gespeichert ist. Beispiel:

```
cd plugins
```

2. Verwenden Sie den folgenden Befehl, um alle Ihre Amazon S3 S3-Buckets aufzulisten.

```
aws s3 ls
```

3. Verwenden Sie den folgenden Befehl, um die Dateien und Ordner im Amazon S3 S3-Bucket für Ihre Umgebung aufzulisten.

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

4. Verwenden Sie den folgenden Befehl, um die `plugins.zip` Datei in den Amazon S3 S3-Bucket für Ihre Umgebung hochzuladen.

```
aws s3 cp plugins.zip s3://YOUR_S3_BUCKET_NAME/plugins.zip
```

## Verwenden der Amazon S3-Konsole

Die Amazon S3 S3-Konsole ist eine webbasierte Benutzeroberfläche, mit der Sie die Ressourcen in Ihrem Amazon S3 S3-Bucket erstellen und verwalten können.

Um mit der Amazon S3 S3-Konsole hochzuladen

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.

2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich DAG-Code im Bereich S3 den Link S3-Bucket aus, um Ihren Speicher-Bucket auf der Amazon S3 S3-Konsole zu öffnen.
4. Klicken Sie auf Hochladen.
5. Wählen Sie Datei hinzufügen.
6. Wählen Sie die lokale Kopie Ihres `plugins.zip` und wählen Sie Hochladen.

## Installation benutzerdefinierter Plugins in Ihrer Umgebung

In diesem Abschnitt wird beschrieben, wie Sie die benutzerdefinierten Plugins, die Sie in Ihren Amazon S3 S3-Bucket hochgeladen haben, installieren, indem Sie bei jeder Aktualisierung der ZIP-Datei den Pfad zur Datei `plugins.zip` und die Version der Datei `plugins.zip` angeben.

### Angabe des Pfads zu **plugins.zip** auf der Amazon MWAA-Konsole (beim ersten Mal)

Wenn Sie zum ersten Mal einen `plugins.zip` in Ihren Amazon S3 S3-Bucket hochladen, müssen Sie auch den Pfad zu der Datei auf der Amazon MWAA-Konsole angeben. Sie müssen diesen Schritt nur einmal ausführen.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie im Bereich DAG-Code in Amazon S3 neben dem Feld Plugins-Datei — optional die Option S3 durchsuchen aus.
5. Wählen Sie die `plugins.zip` Datei in Ihrem Amazon S3 S3-Bucket aus.
6. Wählen Sie Choose (Auswählen) aus.
7. Wählen Sie Weiter, Umgebung aktualisieren.

### Angabe der **plugins.zip** Version auf der Amazon MWAA-Konsole

Sie müssen die Version Ihrer `plugins.zip` Datei auf der Amazon MWAA-Konsole jedes Mal angeben, wenn Sie eine neue Version Ihrer Datei `plugins.zip` in Ihren Amazon S3 S3-Bucket hochladen.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie im Bereich DAG-Code in Amazon S3 eine `plugins.zip` Version aus der Dropdownliste aus.
5. Wählen Sie Weiter.

## Beispielhafte Anwendungsfälle für `plugins.zip`

- Erfahren Sie in, wie Sie ein benutzerdefiniertes Plugin erstellen [Benutzerdefiniertes Plugin mit Apache Hive und Hadoop](#).
- Erfahren Sie in, wie Sie ein benutzerdefiniertes Plugin erstellen [Benutzerdefiniertes Plugin zum PatchenPythonVirtualenvOperator](#).
- Erfahren Sie in, wie Sie ein benutzerdefiniertes Plugin erstellen [Benutzerdefiniertes Plugin mit Oracle](#).
- Erfahren Sie in, wie Sie ein benutzerdefiniertes Plugin erstellen [the section called “Zeitzone einer DAG ändern”](#).

## Als nächstes

- Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.

## Python-Abhängigkeiten installieren

Eine Python-Abhängigkeit ist jedes Paket oder jede Distribution, die nicht in der Apache Airflow-Basisinstallation für Ihre Apache Airflow-Version in Ihrer Amazon Managed Workflows for Apache Airflow-Umgebung enthalten ist. Auf dieser Seite werden die Schritte zur Installation von Apache Airflow Python-Abhängigkeiten in Ihrer Amazon MWAA-Umgebung mithilfe einer `requirements.txt` Datei in Ihrem Amazon S3 S3-Bucket beschrieben.

### Inhalt

- [Voraussetzungen](#)
- [Funktionsweise](#)

- [Überblick über die Python-Abhängigkeiten](#)
  - [Standort- und Größenbeschränkungen für Python-Abhängigkeiten](#)
- [Eine Datei requirements.txt erstellen](#)
  - [Schritt eins: Testen Sie Python-Abhängigkeiten mit dem Amazon MWAA CLI-Hilfsprogramm](#)
  - [Schritt zwei: Erstellen Sie das requirements.txt](#)
- [Auf Amazon requirements.txt S3 hochladen](#)
  - [Mit dem AWS CLI](#)
  - [Verwenden der Amazon S3-Konsole](#)
- [Installation von Python-Abhängigkeiten in Ihrer Umgebung](#)
  - [Angaben des Pfads zu requirements.txt auf der Amazon MWAA-Konsole \(beim ersten Mal\)](#)
  - [Angabe der requirements.txt Version auf der Amazon MWAA-Konsole](#)
- [Logs für Ihr anzeigen requirements.txt](#)
- [Als nächstes](#)

## Voraussetzungen

Sie benötigen Folgendes, bevor Sie die Schritte auf dieser Seite abschließen können.

- Berechtigungen — Ihr AWS Konto muss von Ihrem Administrator Zugriff auf die [FullConsoleAmazonMWAA-Zugriffskontrollrichtlinie](#) für Ihre Umgebung erhalten haben. Darüber hinaus muss Ihrer Amazon MWAA-Umgebung von Ihrer [Ausführungsrolle](#) der Zugriff auf die von Ihrer Umgebung verwendeten AWS Ressourcen gestattet werden.
- Zugriff — Wenn Sie Zugriff auf öffentliche Repositorys benötigen, um Abhängigkeiten direkt auf dem Webserver zu installieren, muss Ihre Umgebung für den Zugriff auf öffentliche Netzwerk-Webserver konfiguriert sein. Weitere Informationen finden Sie unter [the section called “Apache Airflow-Zugriffsmodi”](#).
- Amazon S3 S3-Konfiguration — Der [Amazon S3 S3-Bucket](#), der zum Speichern Ihrer DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten verwendet wird `plugins.zip`, `requirements.txt` muss mit geblocktem öffentlichen Zugriff und aktivierter Versionierung konfiguriert sein.



## Funktionsweise

Auf Amazon MWAA installieren Sie alle Python-Abhängigkeiten, indem Sie eine `requirements.txt` Datei in Ihren Amazon S3 S3-Bucket hochladen und dann bei jeder Aktualisierung der Datei die Version der Datei auf der Amazon MWAA-Konsole angeben. Amazon MWAA wird ausgeführt `pip3 install -r requirements.txt`, um die Python-Abhängigkeiten auf dem Apache Airflow Scheduler und jedem der Worker zu installieren.

Um Python-Abhängigkeiten in Ihrer Umgebung auszuführen, müssen Sie drei Dinge tun:

1. Erstellen Sie lokal eine `requirements.txt` Datei.
2. Laden Sie das Lokale `requirements.txt` in Ihren Amazon S3 S3-Bucket hoch.
3. Geben Sie die Version dieser Datei im Feld Anforderungsdatei auf der Amazon MWAA-Konsole an.

### Note

Wenn Sie zum ersten Mal einen `requirements.txt` Bucket erstellen und in Ihren Amazon S3 S3-Bucket hochladen, müssen Sie auch den Pfad zu der Datei auf der Amazon MWAA-Konsole angeben. Sie müssen diesen Schritt nur einmal ausführen.

## Überblick über die Python-Abhängigkeiten

Sie können Apache Airflow-Extras und andere Python-Abhängigkeiten aus dem Python Package Index (PyPi.org), Python Wheels (`.whl`) oder Python-Abhängigkeiten installieren, die auf einem privaten PyPi /PEP-503-konformen Repo in Ihrer Umgebung gehostet werden.

## Standort- und Größenbeschränkungen für Python-Abhängigkeiten

Der Apache Airflow Scheduler und die Workers suchen beim Start auf dem AWS-verwalteten Fargate-Container für Ihre Umgebung unter nach benutzerdefinierten Plugins. `/usr/local/airflow/plugins`

- Größenbeschränkung. Wir empfehlen eine `requirements.txt` Datei, die auf Bibliotheken verweist, deren Gesamtgröße weniger als 1 GB beträgt. Je mehr Bibliotheken Amazon MWAA installieren muss, desto länger dauert die Startzeit in einer Umgebung. Amazon MWAA begrenzt

zwar nicht explizit die Größe der installierten Bibliotheken, aber wenn Abhängigkeiten nicht innerhalb von zehn Minuten installiert werden können, führt der Fargate-Service zu einem Timeout und versucht, die Umgebung auf einen stabilen Zustand zurückzusetzen.

## Eine Datei requirements.txt erstellen

In den folgenden Schritten werden die Schritte beschrieben, die wir empfehlen, um eine Datei requirements.txt lokal zu erstellen.

### Schritt eins: Testen Sie Python-Abhängigkeiten mit dem Amazon MWAA CLI-Hilfsprogramm

- Das Befehlszeilenschnittstellenprogramm (CLI) repliziert eine Amazon Managed Workflows for Apache Airflow-Umgebung lokal.
- Die CLI erstellt lokal ein Docker-Container-Image, das einem Amazon MWAA-Produktionsimage ähnelt. Auf diese Weise können Sie eine lokale Apache Airflow-Umgebung ausführen, um DAGs, benutzerdefinierte Plugins und Abhängigkeiten zu entwickeln und zu testen, bevor Sie sie auf Amazon MWAA bereitstellen.
- Informationen zum Ausführen der CLI finden Sie unter [aws-mwaa-local-runner](#) on GitHub

### Schritt zwei: Erstellen Sie das **requirements.txt**

Im folgenden Abschnitt wird beschrieben, wie Python-Abhängigkeiten aus dem [Python-Paketindex](#) in einer requirements.txt Datei angegeben werden.

#### Apache Airflow v2

1. Testen Sie lokal. Fügen Sie iterativ weitere Bibliotheken hinzu, um die richtige Kombination von Paketen und ihren Versionen zu finden, bevor Sie eine requirements.txt Datei erstellen. Informationen zum Ausführen des Amazon MWAA-CLI-Dienstprogramms finden Sie unter [aws-mwaa-local-runner](#) on GitHub
2. Sehen Sie sich die Extras des Apache Airflow-Pakets an. Eine Liste der für Apache Airflow v2 auf Amazon MWAA installierten Pakete finden Sie auf der Website unter [Amazon MWAA Local Runner](#). requirements.txt GitHub
3. Fügen Sie eine Beschränkungsanweisung hinzu. Fügen Sie die Einschränkungsdatei für Ihre Apache Airflow v2-Umgebung am Anfang Ihrer requirements.txt Datei hinzu. Apache

Airflow-Einschränkungsdateien spezifizieren die Provider-Versionen, die zum Zeitpunkt einer Apache Airflow-Veröffentlichung verfügbar waren.

Ab Apache Airflow v2.7.2 muss Ihre Anforderungsdatei eine Erklärung enthalten. -- `constraint` Wenn Sie keine Einschränkung angeben, gibt Amazon MWAA eine für Sie an, um sicherzustellen, dass die in Ihren Anforderungen aufgeführten Pakete mit der Version von Apache Airflow kompatibel sind, die Sie verwenden.

Ersetzen Sie im folgenden Beispiel `{environment-version}` durch die Versionsnummer Ihrer Umgebung und `{Python-version}` durch die Version von Python, die mit Ihrer Umgebung kompatibel ist.

Informationen zu der Version von Python, die mit Ihrer Apache Airflow-Umgebung kompatibel ist, finden Sie unter [Apache Airflow-Versionen](#).

```
--constraint "https://raw.githubusercontent.com/apache/airflow/
constraints-{Airflow-version}/constraints-{Python-version}.txt"
```

Wenn die Einschränkungseinstellung feststellt, dass das `xyz==1.0` Paket nicht mit anderen Paketen in Ihrer Umgebung kompatibel ist, kann nicht verhindert werden, dass inkompatible Bibliotheken in Ihrer Umgebung installiert werden. Wenn die Installation eines Pakets fehlschlägt, können Sie die Fehlerprotokolle für jede Apache Airflow-Komponente (den Scheduler, den Worker und den Webserver) im entsprechenden Protokollstream unter CloudWatch Logs einsehen. Weitere Informationen zu Protokolltypen finden Sie unter [the section called "Airflow-Protokolle anzeigen"](#)

4. Apache Airflow-Pakete. Fügen Sie die [Paket-Extras](#) und die Version (`==`) hinzu. Dies hilft zu verhindern, dass Pakete mit demselben Namen, aber unterschiedlicher Version in Ihrer Umgebung installiert werden.

```
apache-airflow[package-extra]==2.5.1
```

5. Python-Bibliotheken. Fügen Sie den Paketnamen und die Version (`==`) in Ihre `requirements.txt` Datei ein. Auf diese Weise wird verhindert, dass ein future aktuelles Update von [PyPi.org](#) automatisch angewendet wird.

```
library == version
```

## Example Boto3 und psycopg2-binary

Dieses Beispiel dient zu Demonstrationszwecken. Die Bibliotheken boto und psycopg2-binary sind in der Apache Airflow v2-Basisinstallation enthalten und müssen nicht in einer Datei angegeben werden. `requirements.txt`

```
boto3==1.17.54
boto==2.49.0
botocore==1.20.54
psycopg2-binary==2.8.6
```

[Wenn ein Paket ohne Version angegeben wird, installiert Amazon MWAA die neueste Version des Pakets von PyPi.org.](#) Diese Version kann zu Konflikten mit anderen Paketen in Ihrem führen. `requirements.txt`

## Apache Airflow v1

1. Testen Sie lokal. Fügen Sie iterativ weitere Bibliotheken hinzu, um die richtige Kombination von Paketen und ihren Versionen zu finden, bevor Sie eine `requirements.txt` Datei erstellen. Informationen zum Ausführen des Amazon MWAA-CLI-Dienstprogramms finden Sie unter [aws-mwaa-local-runner](#) on. GitHub
2. Sehen Sie sich die Extras des Airflow-Pakets an. [Sehen Sie sich die Liste der für Apache Airflow v1.10.12 verfügbaren Pakete unter https://raw.githubusercontent.com/apache/airflow/constraints-1.10.12/constraints-3.7.txt an.](#)
3. Fügen Sie die Beschränkungsdatei hinzu. Fügen Sie die Einschränkungdatei für Apache Airflow v1.10.12 am Anfang Ihrer Datei hinzu. `requirements.txt` Wenn die Einschränkungdatei feststellt, dass das `xyz==1.0` Paket nicht mit anderen Paketen in Ihrer Umgebung kompatibel ist, kann sie nicht verhindern, dass inkompatible Bibliotheken in Ihrer Umgebung installiert werden. `pip3 install`

```
--constraint "https://raw.githubusercontent.com/apache/airflow/
constraints-1.10.12/constraints-3.7.txt"
```

4. Apache Airflow v1.10.12-Pakete. Fügen Sie die [Airflow-Paket-Extras und die Apache Airflow v1.10.12-Version](#) () hinzu. `==` Dadurch wird verhindert, dass Pakete mit demselben Namen, aber unterschiedlicher Version in Ihrer Umgebung installiert werden.

```
apache-airflow[package]==1.10.12
```

### Example Secure Shell (SSH)

Die folgende `requirements.txt` Beispieldatei installiert SSH für Apache Airflow v1.10.12.

```
apache-airflow[ssh]==1.10.12
```

5. Python-Bibliotheken. Fügen Sie den Paketnamen und die Version (==) in Ihre `requirements.txt` Datei ein. Auf diese Weise wird verhindert, dass ein future aktuelles Update von [PyPi.org](https://pypi.org) automatisch angewendet wird.

```
library == version
```

### Example Boto3

Die folgende `requirements.txt` Beispieldatei installiert die Boto3-Bibliothek für Apache Airflow v1.10.12.

```
boto3 == 1.17.4
```

[Wenn ein Paket ohne Version angegeben wird, installiert Amazon MWAA die neueste Version des Pakets von PyPi.org.](#) Diese Version kann zu Konflikten mit anderen Paketen in Ihrem `requirements.txt` führen.

## Auf Amazon **requirements.txt** S3 hochladen

Sie können die Amazon S3 S3-Konsole oder die AWS Command Line Interface (AWS CLI) verwenden, um eine `requirements.txt` Datei in Ihren Amazon S3 S3-Bucket hochzuladen.

### Mit dem AWS CLI

The AWS Command Line Interface (AWS CLI) ist ein Open-Source-Tool, mit dem Sie mithilfe von Befehlen in Ihrer Befehlszeilen-Shell mit AWS Diensten interagieren können. Um die Schritte auf dieser Seite abzuschließen, benötigen Sie Folgendes:

- [AWS CLI — Installieren Sie Version 2.](#)

- [AWS CLI — Schnelle Konfiguration mit `aws configure`](#).

## Zum Hochladen mit dem AWS CLI

1. Verwenden Sie den folgenden Befehl, um alle Ihre Amazon S3 S3-Buckets aufzulisten.

```
aws s3 ls
```

2. Verwenden Sie den folgenden Befehl, um die Dateien und Ordner im Amazon S3 S3-Bucket für Ihre Umgebung aufzulisten.

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

3. Der folgende Befehl lädt eine `requirements.txt` Datei in einen Amazon S3 S3-Bucket hoch.

```
aws s3 cp requirements.txt s3://YOUR_S3_BUCKET_NAME/requirements.txt
```

## Verwenden der Amazon S3-Konsole

Die Amazon S3 S3-Konsole ist eine webbasierte Benutzeroberfläche, mit der Sie die Ressourcen in Ihrem Amazon S3 S3-Bucket erstellen und verwalten können.

Um mit der Amazon S3 S3-Konsole hochzuladen

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich DAG-Code im Bereich S3 den Link S3-Bucket aus, um Ihren Speicher-Bucket auf der Amazon S3 S3-Konsole zu öffnen.
4. Klicken Sie auf Hochladen.
5. Wählen Sie Datei hinzufügen.
6. Wählen Sie die lokale Kopie Ihres `requirements.txt` und wählen Sie Hochladen.

## Installation von Python-Abhängigkeiten in Ihrer Umgebung

In diesem Abschnitt wird beschrieben, wie Sie die Abhängigkeiten, die Sie in Ihren Amazon S3 S3-Bucket hochgeladen haben, installieren, indem Sie den Pfad zur Datei `requirements.txt` und bei jeder Aktualisierung die Version der Datei `requirements.txt` angeben.

## Angeben des Pfads zu **requirements.txt** auf der Amazon MWAA-Konsole (beim ersten Mal)

Wenn Sie zum ersten Mal einen `requirements.txt` Bucket erstellen und in Ihren Amazon S3 S3-Bucket hochladen, müssen Sie auch den Pfad zu der Datei auf der Amazon MWAA-Konsole angeben. Sie müssen diesen Schritt nur einmal ausführen.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie im Bereich DAG-Code in Amazon S3 neben dem Feld Anforderungsdatei — optional die Option S3 durchsuchen aus.
5. Wählen Sie die `requirements.txt` Datei in Ihrem Amazon S3 S3-Bucket aus.
6. Wählen Sie Choose (Auswählen) aus.
7. Wählen Sie Weiter, Umgebung aktualisieren.

Sie können sofort nach Abschluss der Aktualisierung Ihrer Umgebung mit der Verwendung der neuen Pakete beginnen.

## Angabe der **requirements.txt** Version auf der Amazon MWAA-Konsole

Sie müssen die Version Ihrer `requirements.txt` Datei auf der Amazon MWAA-Konsole jedes Mal angeben, wenn Sie eine neue Version Ihrer Datei `requirements.txt` in Ihren Amazon S3 S3-Bucket hochladen.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie im Bereich DAG-Code in Amazon S3 eine `requirements.txt` Version aus der Dropdownliste aus.
5. Wählen Sie Weiter, Umgebung aktualisieren.

Sie können sofort nach Abschluss der Aktualisierung Ihrer Umgebung mit der Verwendung der neuen Pakete beginnen.

## Logs für Ihr anzeigen **requirements.txt**

Sie können Apache Airflow-Protokolle für den Scheduler einsehen, Ihre Workflows planen und Ihren Ordner analysieren. dags In den folgenden Schritten wird beschrieben, wie Sie die Protokollgruppe für den Scheduler auf der Amazon MWAA-Konsole öffnen und Apache Airflow-Protokolle in der Logs-Konsole anzeigen. CloudWatch

Um Protokolle für ein anzuzeigen **requirements.txt**

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich Überwachung die Protokollgruppe Airflow Scheduler aus.
4. Wählen Sie unter **requirements\_install\_ip** Log-Streams die Option Log Streams aus.
5. Sie sollten die Liste der Pakete, die in der Umgebung installiert wurden, unter finden `/usr/local/airflow/.local/bin`. Beispielsweise:

```
Collecting appdirs==1.4.4 (from -r /usr/local/airflow/.local/bin (line 1))
Downloading https://files.pythonhosted.org/
packages/3b/00/2344469e2084fb28kjdsfiuyweb47389789vxbmnbjhsdgf5463acd6cf5e3db69324/
appdirs-1.4.4-py2.py3-none-any.whl
Collecting astroid==2.4.2 (from -r /usr/local/airflow/.local/bin (line 2))
```

6. Überprüfen Sie die Liste der Pakete und ob bei der Installation eines dieser Pakete ein Fehler aufgetreten ist. Wenn etwas schief gelaufen ist, wird möglicherweise ein Fehler ähnlich dem folgenden angezeigt:

```
2021-03-05T14:34:42.731-07:00
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
```

## Als nächstes

- Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on. GitHub



# Löschen von Dateien auf Amazon S3

Auf dieser Seite wird beschrieben, wie die Versionierung in einem Amazon S3 S3-Bucket für eine Amazon Managed Workflows for Apache Airflow-Umgebung funktioniert und welche Schritte zum Löschen einer `DAGplugins.zip`, oder `requirements.txt` Datei erforderlich sind.

## Inhalt

- [Voraussetzungen](#)
- [Überblick über die Versionierung](#)
- [Funktionsweise](#)
- [Löschen einer DAG auf Amazon S3](#)
- [Eine „aktuelle“ Datei requirements.txt oder plugins.zip aus einer Umgebung entfernen](#)
- [Löschen einer „nicht aktuellen“ \(vorherigen\) Version von requirements.txt oder plugins.zip](#)
- [Verwenden von Lebenszyklen zum automatischen Löschen „nicht aktueller“ \(früherer\) Versionen und zum automatischen Löschen von Markierungen](#)
- [Beispiel für eine Lebenszyklusrichtlinie zum automatischen Löschen von „nicht aktuellen“ Versionen von requirements.txt und zum automatischen Löschen von Markierungen](#)
- [Als nächstes](#)

## Voraussetzungen

Sie benötigen Folgendes, bevor Sie die Schritte auf dieser Seite ausführen können.

- Berechtigungen — Ihr AWS Konto muss von Ihrem Administrator Zugriff auf die [FullConsoleAccessAmazonMWAA-Zugriffskontrollrichtlinie](#) für Ihre Umgebung erhalten haben. Darüber hinaus muss Ihrer Amazon MWAA-Umgebung von Ihrer [Ausführungsrolle](#) der Zugriff auf die von Ihrer Umgebung genutzten AWS Ressourcen gestattet werden.
- Zugriff — Wenn Sie Zugriff auf öffentliche Repositorys benötigen, um Abhängigkeiten direkt auf dem Webserver zu installieren, muss Ihre Umgebung für den Zugriff auf öffentliche Netzwerk-Webserver konfiguriert sein. Weitere Informationen finden Sie unter [the section called “Apache Airflow-Zugriffsmodi”](#).
- Amazon S3 S3-Konfiguration — Der [Amazon S3 S3-Bucket](#), der zum Speichern Ihrer DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten verwendet wird `plugins.zip`, `requirements.txt` muss mit geblocktem öffentlichen Zugriff und aktivierter Versionierung konfiguriert sein.

## Überblick über die Versionierung

Die `requirements.txt` und `plugins.zip` in Ihrem Amazon S3 S3-Bucket sind versioniert. Wenn die Amazon S3 S3-Bucket-Versionierung für ein Objekt aktiviert ist und ein Artefakt (z. B. `plugins.zip`) aus einem Amazon S3 S3-Bucket gelöscht wird, wird die Datei nicht vollständig gelöscht. Jedes Mal, wenn ein Artefakt auf Amazon S3 gelöscht wird, wird eine neue Kopie der Datei erstellt, bei der es sich um eine 404-Fehler/0k-Datei (Objekt nicht gefunden) mit der Aufschrift „Ich bin nicht hier“ handelt. Amazon S3 nennt dies eine Löschmarkierung. Eine Löschmarke ist eine „Null“-Version der Datei mit einem Schlüsselnamen (oder Schlüssel) und einer Versions-ID wie jedes andere Objekt.

Wir empfehlen, Dateiversionen und Markierungen regelmäßig zu löschen, um die Speicherkosten für Ihren Amazon S3 S3-Bucket zu senken. Um „nicht aktuelle“ (vorherige) Dateiversionen vollständig zu löschen, müssen Sie die Versionen der Datei (en) und dann die Löschmarkierung für die Version löschen.

## Funktionsweise

Amazon MWAA führt alle dreißig Sekunden einen Synchronisierungsvorgang für Ihren Amazon S3 S3-Bucket durch. Dadurch werden alle DAG-Löschungen in einem Amazon S3 S3-Bucket mit dem Airflow-Image Ihres Fargate-Containers synchronisiert.

Für `plugins.zip` `requirements.txt` und-Dateien treten Änderungen erst nach einem Umgebungsupdate auf, wenn Amazon MWAA ein neues Airflow-Image Ihres Fargate-Containers mit den benutzerdefinierten Plugins und Python-Abhängigkeiten erstellt. Wenn Sie die aktuelle Version einer `requirements.txt` `plugins.zip` OR-Datei löschen und dann Ihre Umgebung aktualisieren, ohne eine neue Version für die gelöschte Datei bereitzustellen, schlägt das Update fehl und es wird eine Fehlermeldung angezeigt, z. B. „Version {version} der Datei kann nicht gelesen werden“.  
{file}

## Löschen einer DAG auf Amazon S3

Eine DAG-Datei (`.py`) ist nicht versioniert und kann direkt auf der Amazon S3 S3-Konsole gelöscht werden. In den folgenden Schritten wird beschrieben, wie Sie eine DAG in Ihrem Amazon S3 S3-Bucket löschen.

Um eine DAG zu löschen

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.

3. Wählen Sie im Bereich DAG-Code im Bereich S3 den Link S3-Bucket aus, um Ihren Speicher-Bucket auf der Amazon S3 S3-Konsole zu öffnen.
4. Wählen Sie den Ordner dags aus.
5. Wählen Sie die DAG aus und klicken Sie auf Löschen.
6. Unter Objekte löschen? , geben Sie `eindelete`.
7. Wählen Sie Delete objects (Objekte löschen).

#### Note

Apache Airflow bewahrt historische DAG-Läufe. Nachdem eine DAG in Apache Airflow ausgeführt wurde, verbleibt sie unabhängig vom Dateistatus in der Airflow-DAG-Liste, bis Sie sie in Apache Airflow löschen. Um eine DAG in Apache Airflow zu löschen, wählen Sie die rote Schaltfläche „Löschen“ unter der Spalte Links.

## Eine „aktuelle“ Datei requirements.txt oder plugins.zip aus einer Umgebung entfernen

Derzeit gibt es keine Möglichkeit, plugins.zip oder requirements.txt aus einer Umgebung zu entfernen, nachdem sie hinzugefügt wurden, aber wir arbeiten an dem Problem. In der Zwischenzeit können Sie das Problem umgehen, indem Sie auf eine leere Text- bzw. ZIP-Datei verweisen.

## Löschen einer „nicht aktuellen“ (vorherigen) Version von requirements.txt oder plugins.zip

Die plugins.zip Dateien requirements.txt und in Ihrem Amazon S3 S3-Bucket sind auf Amazon MWAA versioniert. Wenn Sie diese Dateien in Ihrem Amazon S3 S3-Bucket vollständig löschen möchten, müssen Sie die aktuelle Version (121212) des Objekts (z. B. plugins.zip) abrufen, die Version löschen und dann die Löschmarkierung für die Dateiversion (en) entfernen.

Sie können auch „nicht aktuelle“ (frühere) Dateiversionen auf der Amazon S3 S3-Konsole löschen. Sie müssen die Löschmarkierung jedoch trotzdem mit einer der folgenden Optionen löschen.

- Informationen zum Abrufen der Objektversion finden Sie unter [Objektversionen aus einem Bucket mit aktivierter Versionierung abrufen](#) im Amazon S3 S3-Handbuch.

- Informationen zum Löschen der Objektversion finden Sie unter [Löschen von Objektversionen aus einem Bucket mit aktivierter Versionierung](#) im Amazon S3 S3-Handbuch.
- Informationen zum Entfernen einer Löschmarkierung finden Sie unter [Verwaltung von Löschmarkierungen](#) im Amazon S3 S3-Handbuch.

## Verwenden von Lebenszyklen zum automatischen Löschen „nicht aktueller“ (früherer) Versionen und zum automatischen Löschen von Markierungen

Sie können eine Lebenszyklusrichtlinie für Ihren Amazon S3 S3-Bucket konfigurieren, um „nicht aktuelle“ (frühere) Versionen der Dateien `plugins.zip` und `requirements.txt` in Ihrem Amazon S3 S3-Bucket nach einer bestimmten Anzahl von Tagen zu löschen oder um die Löschmarkierung eines abgelaufenen Objekts zu entfernen.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie in Amazon S3 unter DAG-Code Ihren Amazon S3 S3-Bucket aus.
4. Wählen Sie Lebenszyklusregel erstellen aus.

## Beispiel für eine Lebenszyklusrichtlinie zum automatischen Löschen von „nicht aktuellen“ Versionen von `requirements.txt` und zum automatischen Löschen von Markierungen

Das folgende Beispiel zeigt, wie eine Lebenszyklusregel erstellt wird, die „nicht aktuelle“ Versionen einer Datei `requirements.txt` und deren Löschmarkierungen nach dreißig Tagen dauerhaft löscht.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie in Amazon S3 unter DAG-Code Ihren Amazon S3 S3-Bucket aus.
4. Wählen Sie Lebenszyklusregel erstellen aus.
5. Geben Sie im Feld Name der Lebenszyklusregel den Wert ein `Delete previous requirements.txt versions and delete markers after thirty days`.
6. Im Feld Präfix die Anforderungen.

7. Wählen Sie unter Aktionen für Lebenszyklusregeln die Optionen Frühere Versionen von Objekten dauerhaft löschen und Abgelaufene Löschemarken oder unvollständige mehrteilige Uploads löschen aus.
8. Geben Sie im Feld Anzahl der Tage, nachdem Objekte frühere Versionen geworden sind, den Wert ein. 30
9. Wählen Sie unter Markierungen zum Löschen abgelaufener Objekte die Option Markierungen zum Löschen abgelaufener Objekte löschen aus. Objekte werden nach 30 Tagen dauerhaft gelöscht.

## Als nächstes

- Weitere Informationen zu Amazon S3 S3-Löschemarkierungen finden Sie [unter Löschen von Markierungen verwalten](#).
- Weitere Informationen zu den Lebenszyklen von Amazon S3 finden Sie unter [Ablaufende](#) Objekte.

# Netzwerk

In diesem Handbuch wird die Amazon-VPC-Netzwerkeinrichtung beschrieben, die Sie für eine Amazon-MWAA-Umgebung benötigen.

## Sections

- [Informationen zu Netzwerken in Amazon MWAA](#)
- [Sicherheit in Ihrer VPC auf Amazon MWAA](#)
- [Verwaltung des Zugriffs auf servicespezifische Amazon VPC-Endpunkte auf Amazon MWAA](#)
- [Erstellung der erforderlichen VPC-Serviceendpunkte in einer Amazon VPC mit privatem Routing](#)
- [Verwalten Ihrer eigenen Amazon-VPC-Endpunkte auf Amazon MWAA](#)

## Informationen zu Netzwerken in Amazon MWAA

Eine Amazon VPC ist ein virtuelles Netzwerk, das mit Ihrem - AWS Konto verknüpft ist. Es bietet Ihnen Cloud-Sicherheit und die Möglichkeit, dynamisch zu skalieren, indem Sie eine differenzierte Kontrolle über Ihre virtuelle Infrastruktur und die Segmentierung des Netzwerkverkehrs bieten. Auf dieser Seite wird die Amazon-VPC-Infrastruktur mit öffentlichem oder privatem Routing beschrieben, die zur Unterstützung einer Amazon-Managed-Workflows-for-Apache-Airflow-Umgebung benötigt wird.

## Inhalt

- [Bedingungen](#)
- [Was wird unterstützt](#)
- [Übersicht über die VPC-Infrastruktur](#)
  - [Öffentliches Routing über das Internet](#)
  - [Privates Routing ohne Internetzugang](#)
- [Beispielanwendungsfälle für einen Amazon-VPC- und Apache-Airflow-Zugriffsmodus](#)
  - [Internetzugang ist zulässig – neues Amazon-VPC-Netzwerk](#)
  - [Internetzugang ist nicht zulässig – neues Amazon-VPC-Netzwerk](#)
  - [Internetzugang ist nicht zulässig – vorhandenes Amazon-VPC-Netzwerk](#)

## Bedingungen

### Öffentliches Routing

Ein Amazon VPC-Netzwerk, das Zugriff auf das Internet hat.

### Privates Routing

Ein Amazon-VPC-Netzwerk ohne Zugriff auf das Internet.

## Was wird unterstützt

In der folgenden Tabelle werden die Arten von Amazon VPCs beschrieben, die Amazon MWAA unterstützt.

| Amazon-VPC-Typen                                                                             | Unterstützt |  |
|----------------------------------------------------------------------------------------------|-------------|--|
| Eine Amazon VPC im Besitz des -Kontos, das versucht, die Umgebung zu erstellen.              | Ja          |  |
| Eine gemeinsam genutzte Amazon VPC, in der mehrere AWS Konten ihre AWS Ressourcen erstellen. | Ja          |  |

## Übersicht über die VPC-Infrastruktur

Wenn Sie eine Amazon MWAA-Umgebung erstellen, erstellt Amazon MWAA basierend auf dem Apache Airflow-Zugriffsmodus, den Sie für Ihre Umgebung ausgewählt haben, zwischen einem und zwei VPC-Endpunkten für Ihre Umgebung. Diese Endpunkte werden als Elastic Network Interfaces (ENIs) mit privaten IPs in Ihrer Amazon VPC angezeigt. Nachdem diese Endpunkte erstellt wurden, wird jeglicher Datenverkehr, der an diese IPs gerichtet ist, privat oder öffentlich an die entsprechenden - AWS Services weitergeleitet, die von Ihrer Umgebung verwendet werden.

Im folgenden Abschnitt wird die Amazon-VPC-Infrastruktur beschrieben, die erforderlich ist, um Datenverkehr öffentlich über das Internet oder privat innerhalb Ihrer Amazon VPC weiterzuleiten.

## Öffentliches Routing über das Internet

In diesem Abschnitt wird die Amazon-VPC-Infrastruktur einer Umgebung mit öffentlichem Routing beschrieben. Sie benötigen die folgende VPC-Infrastruktur:

- Eine VPC-Sicherheitsgruppe . Eine VPC-Sicherheitsgruppe dient als virtuelle Firewall zur Steuerung des eingehenden (eingehenden) und ausgehenden (ausgehenden) Netzwerkverkehrs auf einer Instance.
  - Es können bis zu 5 Sicherheitsgruppen angegeben werden.
  - Die Sicherheitsgruppe muss eine selbstreferenzierende eingehende Regel für sich selbst angeben.
  - Die Sicherheitsgruppe muss eine Regel für ausgehenden Datenverkehr für den gesamten Datenverkehr angeben ( $0.0.0.0/0$ ).
  - Die Sicherheitsgruppe muss den gesamten Datenverkehr in der Selbstreferenzierungsregel zulassen. Beispiel: [\(Empfohlen\) Beispiel für eine selbstreferenzierende Sicherheitsgruppe für alle Zugriffe](#)
  - Die Sicherheitsgruppe kann optional den Datenverkehr weiter einschränken, indem sie den Portbereich für den HTTPS-Portbereich 443 und einen TCP-Portbereich angibt 5432. Beispiel: [\(Optional\) Beispiel für eine Sicherheitsgruppe, die den eingehenden Zugriff auf Port 5432 einschränkt](#) und [\(Optional\) Beispiel für eine Sicherheitsgruppe, die den eingehenden Zugriff auf Port 443 einschränkt](#).
- Zwei öffentliche Subnetze. Ein öffentliches Subnetz ist ein Subnetz, das einer Routing-Tabelle zugeordnet ist, die über eine Route zu einem Internet-Gateway verfügt.
  - Es sind zwei öffentliche Subnetze erforderlich. Auf diese Weise kann Amazon MWAA ein neues Container-Image für Ihre Umgebung in Ihrer anderen Availability Zone erstellen, wenn ein Container ausfällt.
  - Diese Subnetze müssen zu verschiedenen Availability-Zonen gehören. Zum Beispiel us-east-1a, us-east-1b.
  - Die Subnetze müssen an ein NAT-Gateway (oder eine NAT-Instance) mit einer Elastic IP-Adresse (EIP) weitergeleitet werden.
  - Die Subnetze müssen über eine Routing-Tabelle verfügen, die den Internetdatenverkehr an ein Internet-Gateway weiterleitet.
- Zwei private Subnetze . Ein privates Subnetz ist ein Subnetz, das keiner Routing-Tabelle zugeordnet ist, die über eine Route zu einem Internet-Gateway verfügt.



- Zwei private Subnetze sind erforderlich. Auf diese Weise kann Amazon MWAA ein neues Container-Image für Ihre Umgebung in Ihrer anderen Availability Zone erstellen, wenn ein Container ausfällt.
- Diese Subnetze müssen zu verschiedenen Availability-Zonen gehören. Zum Beispiel us-east-1a, us-east-1b.
- Die Subnetze müssen über eine Routing-Tabelle zu einem NAT-Gerät (Gateway oder Instance) verfügen.
- Die Subnetze dürfen nicht an ein Internet-Gateway weitergeleitet werden.
- Eine Netzwerk-Zugriffssteuerungsliste (ACL). Eine NACL verwaltet (nach Regeln zum Zulassen oder Verweigern) ein- und ausgehenden Datenverkehr auf Subnetzebene.
  - Die NACL muss über eine eingehende Regel verfügen, die den gesamten Datenverkehr zulässt (0.0.0.0/0).
  - Die NACL muss über eine Regel für ausgehenden Datenverkehr verfügen, die den gesamten Datenverkehr verweigert (0.0.0.0/0).
  - Beispiel: [\(Empfohlen\) Beispiel-ACLs](#)
- Zwei NAT-Gateways (oder NAT-Instances). Ein NAT-Gerät leitet den Datenverkehr von den Instances im privaten Subnetz an das Internet oder andere - AWS Services weiter und leitet die Antwort dann zurück an die Instances.
  - Das NAT-Gerät muss an ein öffentliches Subnetz angehängt werden. (Ein NAT-Gerät pro öffentlichem Subnetz.)
  - Dem NAT-Gerät muss eine Elastic IPv4-Adresse (EIP) an jedes öffentliche Subnetz angefügt sein.
- Ein Internet-Gateway. Ein Internet-Gateway verbindet eine Amazon VPC mit dem Internet und anderen - AWS Services.
  - Ein Internet-Gateway muss an die Amazon VPC angehängt sein.

## Privates Routing ohne Internetzugang

In diesem Abschnitt wird die Amazon-VPC-Infrastruktur einer Umgebung mit privatem Routing beschrieben. Sie benötigen die folgende VPC-Infrastruktur:

- Eine VPC-Sicherheitsgruppe . Eine VPC-Sicherheitsgruppe dient als virtuelle Firewall zur Steuerung des eingehenden (eingehenden) und ausgehenden (ausgehenden) Netzwerkverkehrs auf einer Instance.

- Es können bis zu 5 Sicherheitsgruppen angegeben werden.
- Die Sicherheitsgruppe muss eine selbstreferenzierende eingehende Regel für sich selbst angeben.
- Die Sicherheitsgruppe muss eine Regel für ausgehenden Datenverkehr für den gesamten Datenverkehr angeben (0.0.0.0/0).
- Die Sicherheitsgruppe muss den gesamten Datenverkehr in der Selbstreferenzierungsregel zulassen. Beispiel: [\(Empfohlen\) Beispiel für eine selbstreferenzierende Sicherheitsgruppe für alle Zugriffe](#)
- Die Sicherheitsgruppe kann optional den Datenverkehr weiter einschränken, indem sie den Portbereich für den HTTPS-Portbereich 443 und einen TCP-Portbereich angibt 5432. Beispiel: [\(Optional\) Beispiel für eine Sicherheitsgruppe, die den eingehenden Zugriff auf Port 5432 einschränkt](#) und [\(Optional\) Beispiel für eine Sicherheitsgruppe, die den eingehenden Zugriff auf Port 443 einschränkt](#).
- Zwei private Subnetze . Ein privates Subnetz ist ein Subnetz, das keiner Routing-Tabelle zugeordnet ist, die über eine Route zu einem Internet-Gateway verfügt.
  - Zwei private Subnetze sind erforderlich. Auf diese Weise kann Amazon MWAA ein neues Container-Image für Ihre Umgebung in Ihrer anderen Availability Zone erstellen, wenn ein Container ausfällt.
  - Diese Subnetze müssen zu verschiedenen Availability-Zonen gehören. Zum Beispiel us-east-1a, us-east-1b.
  - Die Subnetze müssen über eine Routing-Tabelle zu Ihren VPC-Endpunkten verfügen.
  - Die Subnetze dürfen keine Routing-Tabelle zu einem NAT-Gerät (Gateway oder Instance) oder einem Internet-Gateway haben.
- Eine Netzwerk-Zugriffssteuerungsliste (ACL). Eine NACL verwaltet (nach Regeln zum Zulassen oder Verweigern) ein- und ausgehenden Datenverkehr auf Subnetzebene.
  - Die NACL muss über eine eingehende Regel verfügen, die den gesamten Datenverkehr zulässt (0.0.0.0/0).
  - Die NACL muss über eine Regel für ausgehenden Datenverkehr verfügen, die den gesamten Datenverkehr verweigert (0.0.0.0/0).
  - Beispiel: [\(Empfohlen\) Beispiel-ACLs](#)
- Eine lokale Routing-Tabelle . Eine lokale Routing-Tabelle ist eine Standardroute für die Kommunikation innerhalb der VPC.
  - Die lokale Routing-Tabelle muss Ihren privaten Subnetzen zugeordnet sein.

- Die lokale Routing-Tabelle muss Instances in Ihrer VPC die Kommunikation mit Ihrem eigenen Netzwerk ermöglichen. Wenn Sie beispielsweise einen verwenden, AWS Client VPN um auf den VPC-Schnittstellenendpunkt für Ihren Apache Airflow Web Server zuzugreifen, muss die Routing-Tabelle an den VPC-Endpunkt weitergeleitet werden.
- VPC-Endpunkte für jeden AWS Service, der von Ihrer Umgebung verwendet wird, und Apache Airflow VPC-Endpunkte in derselben AWS Region und Amazon VPC wie Ihre Amazon MWAA-Umgebung.
  - Ein VPC-Endpunkt für jeden AWS Service, der von der Umgebung verwendet wird, und VPC-Endpunkte für Apache Airflow. Beispiel: [\(Erforderlich\) VPC-Endpunkte](#)
  - Für die VPC-Endpunkte muss privates DNS aktiviert sein.
  - Die VPC-Endpunkte müssen den beiden privaten Subnetzen Ihrer Umgebung zugeordnet sein.
  - Die VPC-Endpunkte müssen der Sicherheitsgruppe Ihrer Umgebung zugeordnet sein.
  - Die VPC-Endpunktrichtlinie für jeden Endpunkt sollte so konfiguriert werden, dass der Zugriff auf von der Umgebung verwendete AWS Services ermöglicht wird. Beispiel: [\(Empfohlen\) Beispiel für eine VPC-Endpunktrichtlinie, um allen Zugriff zuzulassen](#)
  - Eine VPC-Endpunktrichtlinie für Amazon S3 sollte so konfiguriert werden, dass der Bucket-Zugriff zugelassen wird. Beispiel: [\(Empfohlen\) Beispiel für eine Amazon S3 S3-Gateway-Endpunktrichtlinie zum Zulassen des Bucket-Zugriffs](#)

## Beispielanwendungsfälle für einen Amazon-VPC- und Apache-Airflow-Zugriffsmodus

In diesem Abschnitt werden die verschiedenen Anwendungsfälle für den Netzwerkzugriff in Ihrer Amazon VPC und der Zugriffsmodus des Apache-Airflow-Webserver beschrieben, den Sie in der Amazon-MWAA-Konsole auswählen sollten.

### Internetzugang ist zulässig – neues Amazon-VPC-Netzwerk

Wenn Ihr Unternehmen Internetzugang in Ihrer VPC zulässt und Sie möchten, dass Benutzer über das Internet auf Ihren Apache-Airflow-Webserver zugreifen:

1. Erstellen Sie ein Amazon-VPC-Netzwerk mit Internetzugriff .
2. Erstellen Sie eine Umgebung mit dem Modus Öffentlicher Netzwerkzugriff für Ihren Apache-Airflow-Webserver.

3. Was wir empfehlen: Wir empfehlen, gleichzeitig die AWS CloudFormation Schnellstartvorlage zu verwenden, die die Amazon-VPC-Infrastruktur, einen Amazon S3-Bucket und eine Amazon-MWAA-Umgebung erstellt. Weitere Informationen hierzu finden Sie unter [Schnellstartanleitung für Amazon Managed Workflows für Apache Airflow](#).

Wenn Ihr Unternehmen Internetzugang in Ihrer VPC zulässt und Sie den Zugriff auf den Apache Airflow-Webserver auf Benutzer in Ihrer VPC beschränken möchten:

1. Erstellen Sie ein Amazon-VPC-Netzwerk mit Internetzugriff .
2. Erstellen Sie einen Mechanismus, um von Ihrem Computer aus auf den VPC-Schnittstellenendpunkt für Ihren Apache-Airflow-Webserver zuzugreifen.
3. Erstellen Sie eine Umgebung mit dem privaten Netzwerkzugriffsmodus für Ihren Apache Airflow Webserver.
4. Was wir empfehlen:
  - a. Wir empfehlen [Option eins: Erstellen des VPC-Netzwerks auf der Amazon MWAA-Konsole](#), die Amazon MWAA-Konsole in oder die AWS CloudFormation Vorlage in zu verwenden [Option zwei: Erstellen eines Amazon VPC-Netzwerks mit Internetzugang](#).
  - b. Wir empfehlen, den Zugriff mit einem AWS Client VPN auf Ihren Apache Airflow Web Server in zu konfigurieren [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem AWS Client VPN](#).

## Internetzugang ist nicht zulässig – neues Amazon-VPC-Netzwerk

Wenn der Internetzugang in Ihrer VPC von Ihrer Organisation nicht zugelassen wird:

1. Erstellen Sie ein Amazon-VPC-Netzwerk ohne Internetzugang.
2. Erstellen Sie einen Mechanismus, um von Ihrem Computer aus auf den VPC-Schnittstellenendpunkt für Ihren Apache-Airflow-Webserver zuzugreifen.
3. Erstellen Sie VPC-Endpunkte für jeden AWS Service, der von Ihrer Umgebung verwendet wird.
4. Erstellen Sie eine Umgebung mit dem privaten Netzwerkzugriffsmodus für Ihren Apache Airflow Webserver.
5. Was wir empfehlen:
  - a. Wir empfehlen, die AWS CloudFormation Vorlage zu verwenden, um eine Amazon VPC ohne Internetzugang und die VPC-Endpunkte für jeden AWS Service zu erstellen,

der von Amazon MWAA in verwendet wird [Option drei: Erstellen eines Amazon VPC-NetzwerksohnelInternetzugang](#).

- b. Wir empfehlen, den Zugriff mit einem AWS Client VPN auf Ihren Apache Airflow Web Server in zu konfigurieren [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem AWS Client VPN](#).

## Internetzugang ist nicht zulässig – vorhandenes Amazon-VPC-Netzwerk

Wenn der Internetzugang in Ihrer VPC von Ihrer Organisation nicht zugelassen wird und Sie bereits über das erforderliche Amazon-VPC-Netzwerk ohne Internetzugang verfügen:

1. Erstellen Sie VPC-Endpunkte für jeden AWS Service, der von Ihrer Umgebung verwendet wird.
2. Erstellen Sie VPC-Endpunkte für Apache Airflow.
3. Erstellen Sie einen Mechanismus, um von Ihrem Computer aus auf den VPC-Schnittstellenendpunkt für Ihren Apache-Airflow-Webserver zuzugreifen.
4. Erstellen Sie eine Umgebung mit dem privaten Netzwerkzugriffsmodus für Ihren Apache-Airflow-Webserver.
5. Was wir empfehlen:
  - a. Wir empfehlen, die VPC-Endpunkte zu erstellen und anzuhängen, die für jeden von Amazon MWAA verwendeten AWS Service erforderlich sind, sowie die VPC-Endpunkte, die für Apache Airflow in erforderlich sind [Erstellung der erforderlichen VPC-Serviceendpunkte in einer Amazon VPC mit privatem Routing](#).
  - b. Wir empfehlen, den Zugriff mit einem AWS Client VPN auf Ihren Apache Airflow Web Server in zu konfigurieren [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem AWS Client VPN](#).

## Sicherheit in Ihrer VPC auf Amazon MWAA

Auf dieser Seite werden die Amazon VPC-Komponenten beschrieben, die zur Sicherung Ihrer Amazon Managed Workflows for Apache Airflow-Umgebung verwendet werden, sowie die für diese Komponenten erforderlichen Konfigurationen.

### Inhalt

- [Bedingungen](#)

- [Überblick über die Sicherheit](#)
- [Netzwerk-Zugriffskontrolllisten \(ACLs\)](#)
  - [\(Empfohlen\) Beispiel-ACLs](#)
- [VPC-Sicherheitsgruppen](#)
  - [\(Empfohlen\) Beispiel für eine selbstreferenzierende Sicherheitsgruppe für alle Zugriffe](#)
  - [\(Optional\) Beispiel für eine Sicherheitsgruppe, die den eingehenden Zugriff auf Port 5432 einschränkt](#)
  - [\(Optional\) Beispiel für eine Sicherheitsgruppe, die den eingehenden Zugriff auf Port 443 einschränkt](#)
- [VPC-Endpunktrichtlinien \(nur privates Routing\)](#)
  - [\(Empfohlen\) Beispiel für eine VPC-Endpunktrichtlinie, um allen Zugriff zuzulassen](#)
  - [\(Empfohlen\) Beispiel für eine Amazon S3 S3-Gateway-Endpunktrichtlinie zum Zulassen des Bucket-Zugriffs](#)

## Bedingungen

### Öffentliches Routing

Ein Amazon VPC-Netzwerk, das Zugang zum Internet hat.

### Privates Routing

Ein Amazon VPC-Netzwerk ohne Zugang zum Internet.

## Überblick über die Sicherheit

Sicherheitsgruppen und Zugriffskontrolllisten (ACLs) bieten Möglichkeiten, den Netzwerkverkehr in den Subnetzen und Instances in Ihrer Amazon VPC mithilfe von Regeln, die Sie angeben, zu kontrollieren.

- Der Netzwerkverkehr zu und von einem Subnetz kann durch Zugriffskontrolllisten (ACLs) gesteuert werden. Sie benötigen nur eine ACL, und dieselbe ACL kann in mehreren Umgebungen verwendet werden.
- Der Netzwerkverkehr zu und von einer Instance kann von einer Amazon VPC-Sicherheitsgruppe gesteuert werden. Sie können zwischen einer und fünf Sicherheitsgruppen pro Umgebung verwenden.

- Der Netzwerkverkehr zu und von einer Instance kann auch durch VPC-Endpunktrichtlinien gesteuert werden. Wenn Ihr Unternehmen den Internetzugang innerhalb Ihrer Amazon VPC nicht erlaubt und Sie ein Amazon VPC-Netzwerk mit privatem Routing verwenden, ist eine VPC-Endpunktrichtlinie für die [VPC-Endpunkte und Apache Airflow AWS VPC-Endpunkte](#) erforderlich.

## Netzwerk-Zugriffskontrolllisten (ACLs)

Eine [Network Access Control List \(ACL\)](#) kann eingehenden und ausgehenden Datenverkehr auf Subnetzebene verwalten (nach Regeln zum Zulassen oder Verweigern). Eine ACL ist zustandslos, was bedeutet, dass Regeln für eingehenden und ausgehenden Datenverkehr getrennt und explizit angegeben werden müssen. Es wird verwendet, um die Arten von Netzwerkverkehr anzugeben, die zu den Instances in einem VPC-Netzwerk ein- oder ausgehen dürfen.

Jede Amazon VPC hat eine Standard-ACL, die den gesamten eingehenden und ausgehenden Datenverkehr zulässt. Sie können die Standard-ACL-Regeln bearbeiten oder eine benutzerdefinierte ACL erstellen und sie an Ihre Subnetze anhängen. Einem Subnetz kann immer nur eine ACL zugeordnet sein, aber eine ACL kann mehreren Subnetzen zugeordnet werden.

### (Empfohlen) Beispiel-ACLs

Das folgende Beispiel zeigt die ACL-Regeln für eingehenden und ausgehenden Datenverkehr, die für eine Amazon-VPC für eine Amazon-VPC mit öffentlichem Routing oder privatem Routing verwendet werden können.

| Nummer der Regel | Typ                        | Protocol (Protokoll) | Port-Bereich | Source    | Erlauben/Verweigern                                                      |
|------------------|----------------------------|----------------------|--------------|-----------|--------------------------------------------------------------------------|
| 100              | Gesamter IPv4-Datenverkehr | Alle                 | Alle         | 0.0.0.0/0 | Sobald Sie die Details auf dieser Seite überprüft haben, klicken Sie auf |

| Nummer der Regel | Typ                        | Protocol (Protokoll) | Port-Bereich | Source    | Erlauben/Verweigern |
|------------------|----------------------------|----------------------|--------------|-----------|---------------------|
| *                | Gesamter IPv4-Datenverkehr | Alle                 | Alle         | 0.0.0.0/0 | Deny                |

## VPC-Sicherheitsgruppen

Eine [VPC-Sicherheitsgruppe](#) fungiert als virtuelle Firewall, die den Netzwerkverkehr auf Instanzebene steuert. Eine Sicherheitsgruppe ist statusbehaftet, was bedeutet, dass sie antworten darf, wenn eine eingehende Verbindung zugelassen ist. Es wird verwendet, um die Arten von Netzwerkverkehr anzugeben, die von den Instances in einem VPC-Netzwerk empfangen werden dürfen.

Jede Amazon VPC hat eine Standardsicherheitsgruppe. Standardmäßig gibt es keine Regeln für eingehenden Datenverkehr. Es gibt eine ausgehende Regel, die den gesamten ausgehenden Verkehr zulässt. Sie können die Standardregeln für Sicherheitsgruppen bearbeiten oder eine benutzerdefinierte Sicherheitsgruppe erstellen und sie an Ihre Amazon VPC anhängen. Auf Amazon MWAA müssen Sie Regeln für eingehenden und ausgehenden Datenverkehr konfigurieren, um den Datenverkehr auf Ihre NAT-Gateways zu leiten.

### (Empfohlen) Beispiel für eine selbstreferenzierende Sicherheitsgruppe für alle Zugriffe

Das folgende Beispiel zeigt die Regeln für eingehende Sicherheitsgruppen, die den gesamten Datenverkehr für eine Amazon-VPC für eine Amazon-VPC mit öffentlichem Routing oder privatem Routing zulassen. Die Sicherheitsgruppe in diesem Beispiel ist eine Regel, die sich selbst auf sich selbst bezieht.

| Typ                   | Protokoll | Art der Quelle | Quelle                                      |
|-----------------------|-----------|----------------|---------------------------------------------|
| Gesamter Datenverkehr | Alle      | Alle           | sg-0909e8e81919/-group my-mwaa-vpc-security |

Das folgende Beispiel zeigt die Regeln für ausgehende Sicherheitsgruppen.



| Typ                   | Protokoll | Art der Quelle | Quelle    |  |  |
|-----------------------|-----------|----------------|-----------|--|--|
| Gesamter Datenverkehr | Alle      | Alle           | 0.0.0.0/0 |  |  |

### (Optional) Beispiel für eine Sicherheitsgruppe, die den eingehenden Zugriff auf Port 5432 einschränkt

Das folgende Beispiel zeigt die Regeln für eingehende Sicherheitsgruppen, die den gesamten HTTPS-Verkehr auf Port 5432 für die Amazon Aurora PostgreSQL-Metadatendatenbank (im Besitz von Amazon MWAA) für Ihre Umgebung zulassen.

#### Note

Wenn Sie den Datenverkehr mithilfe dieser Regel einschränken möchten, müssen Sie eine weitere Regel hinzufügen, um TCP-Verkehr auf Port 443 zuzulassen.

| Typ        | Protocol (Protokoll) | Port-Bereich | Source type (Quellentyp) | Quelle                                      |  |
|------------|----------------------|--------------|--------------------------|---------------------------------------------|--|
| Custom TCP | TCP                  | 5432         | Benutzerdefiniert        | sg-0909e8e81919/-group my-mwaa-vpc-security |  |

### (Optional) Beispiel für eine Sicherheitsgruppe, die den eingehenden Zugriff auf Port 443 einschränkt

Das folgende Beispiel zeigt die Regeln für eingehende Sicherheitsgruppen, die den gesamten TCP-Verkehr auf Port 443 für den Apache Airflow-Webserver zulassen.

| Typ   | Protocol (Protokoll) | Port-Bereich | Source type (Quellentyp) | Quelle                                      |  |
|-------|----------------------|--------------|--------------------------|---------------------------------------------|--|
| HTTPS | TCP                  | 443          | Benutzerdefiniert        | sg-0909e8e81919/-group my-mwaa-vpc-security |  |

## VPC-Endpunktrichtlinien (nur privates Routing)

Eine [VPC-Endpunktrichtlinie \(AWS PrivateLink\)](#) steuert den Zugriff auf AWS Dienste von Ihrem privaten Subnetz aus. Eine VPC-Endpunktrichtlinie ist eine IAM-Ressourcenrichtlinie, die Sie Ihrem VPC-Gateway oder Schnittstellenendpunkt zuordnen. In diesem Abschnitt werden die Berechtigungen beschrieben, die für die VPC-Endpunktrichtlinien für jeden VPC-Endpunkt erforderlich sind.

Wir empfehlen, für jeden der von Ihnen erstellten VPC-Endpoints eine VPC-Schnittstellen-Endpunktrichtlinie zu verwenden, die vollen Zugriff auf alle AWS Dienste ermöglicht, und Ihre Ausführungsrolle ausschließlich für Berechtigungen zu verwenden. AWS

(Empfohlen) Beispiel für eine VPC-Endpunktrichtlinie, um allen Zugriff zuzulassen

Das folgende Beispiel zeigt eine VPC-Schnittstellen-Endpunktrichtlinie für eine Amazon-VPC mit privatem Routing.

```
{
 "Statement": [
 {
 "Action": "*",
 "Effect": "Allow",
 "Resource": "*",
 "Principal": "*"
 }
]
}
```

## (Empfohlen) Beispiel für eine Amazon S3 S3-Gateway-Endpunktrichtlinie zum Zulassen des Bucket-Zugriffs

Das folgende Beispiel zeigt eine VPC-Gateway-Endpunktrichtlinie, die den Zugriff auf die Amazon S3 S3-Buckets ermöglicht, die für Amazon ECR-Operationen für eine Amazon VPC mit privatem Routing erforderlich sind. Dies ist erforderlich, damit Ihr Amazon ECR-Image abgerufen werden kann, zusätzlich zu dem Bucket, in dem Ihre DAGs und unterstützenden Dateien gespeichert sind.

```
{
 "Statement": [
 {
 "Sid": "Access-to-specific-bucket-only",
 "Principal": "*",
 "Action": [
 "s3:GetObject"
],
 "Effect": "Allow",
 "Resource": ["arn:aws:s3:::prod-region-starport-layer-bucket/*"]
 }
]
}
```

## Verwaltung des Zugriffs auf servicespezifische Amazon VPC-Endpunkte auf Amazon MWAA

Ein VPC-Endpunkt (AWS PrivateLink) ermöglicht es Ihnen, Ihre VPC privat mit Diensten zu verbinden, auf denen gehostet wird, AWS ohne dass ein Internet-Gateway, ein NAT-Gerät, VPN oder Firewall-Proxys erforderlich sind. Bei diesen Endpunkten handelt es sich um horizontal skalierbare und hochverfügbare virtuelle Geräte, die die Kommunikation zwischen Instances in Ihrer VPC und AWS Services ermöglichen. Auf dieser Seite werden die von Amazon MWAA erstellten VPC-Endpoints beschrieben und wie Sie auf den VPC-Endpunkt für Ihren Apache Airflow-Webserver zugreifen können, wenn Sie den privaten Netzwerkzugriffsmodus in Amazon Managed Workflows for Apache Airflow gewählt haben.

### Inhalt

- [Preisgestaltung](#)
- [Übersicht über VPC-Endpunkte](#)
  - [Öffentlicher Netzwerkzugriffsmodus](#)

- [Privater Netzwerkzugriffsmodus](#)
- [Erlaubnis zur Nutzung anderer Dienste AWS](#)
- [VPC-Endpoints anzeigen](#)
  - [VPC-Endpunkte auf der Amazon VPC-Konsole anzeigen](#)
  - [Identifizieren der privaten IP-Adressen Ihres Apache Airflow-Webserver und seines VPC-Endpunkts](#)
- [Zugreifen auf den VPC-Endpunkt für Ihren Apache Airflow Webserver \(privater Netzwerkzugriff\)](#)
  - [Verwenden Sie ein AWS Client VPN](#)
  - [Verwenden eines Linux-Bastion-Hosts](#)
  - [Verwenden eines Load Balancer \(fortgeschritten\)](#)

## Preisgestaltung

- [AWS PrivateLink Preise](#)

## Übersicht über VPC-Endpunkte

Wenn Sie eine Amazon MWAA-Umgebung erstellen, erstellt Amazon MWAA zwischen einem und zwei VPC-Endpunkten für Ihre Umgebung. Diese Endpunkte werden als Elastic Network Interfaces (ENIs) mit privaten IPs in Ihrer Amazon VPC angezeigt. Nachdem diese Endpunkte erstellt wurden, wird jeglicher Datenverkehr, der für diese IPs bestimmt ist, privat oder öffentlich an die entsprechenden AWS Dienste weitergeleitet, die von Ihrer Umgebung verwendet werden.

## Öffentlicher Netzwerkzugriffsmodus

Wenn Sie den Modus Öffentlicher Netzwerkzugriff für Ihren Apache Airflow-Webserver gewählt haben, wird der Netzwerkverkehr öffentlich über das Internet geleitet.

- Amazon MWAA erstellt einen VPC-Schnittstellenendpunkt für Ihre Amazon Aurora PostgreSQL-Metadatenbank. Der Endpunkt wird in den Availability Zones erstellt, die Ihren privaten Subnetzen zugeordnet sind, und ist unabhängig von anderen Konten. AWS
- Amazon MWAA bindet dann eine IP-Adresse aus Ihren privaten Subnetzen an die Schnittstellenendpunkte. Dies soll die bewährte Methode unterstützen, eine einzelne IP aus jeder Availability Zone der Amazon VPC zu binden.

## Privater Netzwerkzugriffsmodus

Wenn Sie den privaten Netzwerkzugriffsmodus für Ihren Apache Airflow-Webserver gewählt haben, wird der Netzwerkverkehr innerhalb Ihrer Amazon VPC privat weitergeleitet.

- Amazon MWAA erstellt einen VPC-Schnittstellenendpunkt für Ihren Apache Airflow-Webserver und einen Schnittstellenendpunkt für Ihre Amazon Aurora PostgreSQL-Metadatendatenbank. Die Endpunkte werden in den Availability Zones erstellt, die Ihren privaten Subnetzen zugeordnet sind, und sind unabhängig von anderen Konten. AWS
- Amazon MWAA bindet dann eine IP-Adresse aus Ihren privaten Subnetzen an die Schnittstellenendpunkte. Dies soll die bewährte Methode unterstützen, eine einzelne IP aus jeder Availability Zone der Amazon VPC zu binden.

## Erlaubnis zur Nutzung anderer Dienste AWS

Die Schnittstellenendpunkte verwenden die Ausführungsrolle für Ihre Umgebung in AWS Identity and Access Management (IAM), um die Berechtigungen für AWS Ressourcen zu verwalten, die von Ihrer Umgebung verwendet werden. Da mehr AWS Dienste für eine Umgebung aktiviert sind, müssen Sie für jeden Dienst Berechtigungen mithilfe der Ausführungsrolle Ihrer Umgebung konfigurieren. Informationen zum Hinzufügen von Berechtigungen finden Sie unter [Amazon MWAA-Ausführungsrolle](#).

Wenn Sie den privaten Netzwerkzugriffsmodus für Ihren Apache Airflow-Webserver ausgewählt haben, müssen Sie auch in der VPC-Endpunktrichtlinie für jeden Endpunkt eine Genehmigung zulassen. Weitere Informationen hierzu finden Sie unter [the section called “VPC-Endpunktrichtlinien \(nur privates Routing\)”](#).

## VPC-Endpoints anzeigen

In diesem Abschnitt wird beschrieben, wie Sie die von Amazon MWAA erstellten VPC-Endpoints anzeigen und wie Sie die privaten IP-Adressen für Ihren Apache Airflow VPC-Endpoint identifizieren.

### VPC-Endpoints auf der Amazon VPC-Konsole anzeigen

Der folgende Abschnitt zeigt die Schritte zum Anzeigen der VPC-Endpoints, die von Amazon MWAA erstellt wurden, und aller VPC-Endpoints, die Sie möglicherweise erstellt haben, wenn Sie privates Routing für Ihre Amazon VPC verwenden.

## So zeigen Sie die VPC-Endpunkte an

1. Öffnen Sie die [Seite Endpoints](#) in der Amazon VPC-Konsole.
2. Verwenden Sie die AWS Regionsauswahl, um Ihre Region auszuwählen.
3. Sie sollten die von Amazon MWAA erstellten VPC-Schnittstellenendpunkte sowie alle VPC-Endpunkte sehen, die Sie möglicherweise erstellt haben, wenn Sie privates Routing in Ihrer Amazon VPC verwenden.

Weitere Informationen zu den VPC-Service-Endpunkten, die für eine Amazon VPC mit privatem Routing erforderlich sind, finden Sie unter [Erstellung der erforderlichen VPC-Serviceendpunkte in einer Amazon VPC mit privatem Routing](#)

## Identifizieren der privaten IP-Adressen Ihres Apache Airflow-Webserver und seines VPC-Endpunkts

In den folgenden Schritten wird beschrieben, wie Sie den Hostnamen Ihres Apache Airflow-Webserver und dessen VPC-Schnittstellenendpunkt sowie deren private IP-Adressen abrufen.

1. Verwenden Sie den folgenden Befehl AWS Command Line Interface (AWS CLI), um den Hostnamen für Ihren Apache Airflow-Webserver abzurufen.

```
aws mwaa get-environment --name YOUR_ENVIRONMENT_NAME --query
'Environment.WebserverUrl'
```

Sie sollten etwas Ähnliches wie die folgende Antwort sehen:

```
"99aa99aa-55aa-44a1-a91f-f4552cf4e2f5-vpce.c10.us-west-2.airflow.amazonaws.com"
```

2. Führen Sie einen Dig-Befehl für den Hostnamen aus, der in der Antwort des vorherigen Befehls zurückgegeben wurde. Beispielsweise:

```
dig CNAME +short 99aa99aa-55aa-44a1-a91f-f4552cf4e2f5-vpce.c10.us-
west-2.airflow.amazonaws.com
```

Sie sollten etwas Ähnliches wie die folgende Antwort sehen:

```
vpce-0699aa333a0a0a0-bf90xjtr.vpce-svc-00bb7c2ca2213bc37.us-
west-2.vpce.amazonaws.com.
```

3. Verwenden Sie den folgenden AWS Command Line Interface (AWS CLI) Befehl, um den DNS-Namen des VPC-Endpunkts abzurufen, der in der Antwort auf den vorherigen Befehl zurückgegeben wurde. Beispielsweise:

```
aws ec2 describe-vpc-endpoints | grep vpce-0699aa333a0a0a0-bf90xjtr.vpce-
svc-00bb7c2ca2213bc37.us-west-2.vpce.amazonaws.com.
```

Sie sollten etwas Ähnliches wie die folgende Antwort sehen:

```
"DnsName": "vpce-066777a0a0a0-bf90xjtr.vpce-svc-00bb7c2ca2213bc37.us-
west-2.vpce.amazonaws.com",
```

4. Führen Sie entweder den Befehl nslookup oder dig für Ihren Apache Airflow-Hostnamen und den DNS-Namen des VPC-Endpunkts aus, um die IP-Adressen abzurufen. Beispielsweise:

```
dig +short YOUR_AIRFLOW_HOST_NAME YOUR_AIRFLOW_VPC_ENDPOINT_DNS
```

Sie sollten etwas Ähnliches wie die folgende Antwort sehen:

```
192.0.5.1
192.0.6.1
```

## Zugreifen auf den VPC-Endpunkt für Ihren Apache Airflow Webserver (privater Netzwerkzugriff)

Wenn Sie den privaten Netzwerkzugriffsmodus für Ihren Apache Airflow-Webserver ausgewählt haben, müssen Sie einen Mechanismus für den Zugriff auf den VPC-Schnittstellenendpunkt für Ihren Apache Airflow-Webserver erstellen. Sie müssen für diese Ressourcen dieselbe Amazon VPC, VPC-Sicherheitsgruppe und dieselben privaten Subnetze wie Ihre Amazon MWAA-Umgebung verwenden.

### Verwenden Sie ein AWS Client VPN

AWS Client VPN ist ein verwalteter clientbasierter VPN-Dienst, mit dem Sie sicher auf Ihre AWS Ressourcen und Ressourcen in Ihrem lokalen Netzwerk zugreifen können. Es bietet mithilfe des OpenVPN-Clients eine sichere TLS-Verbindung von jedem Standort aus.

Wir empfehlen, dem Amazon MWAA-Tutorial zu folgen, um ein Client VPN zu konfigurieren.: [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem AWS Client VPN](#)

## Verwenden eines Linux-Bastion-Hosts

Ein Bastion-Host ist ein Server, dessen Zweck darin besteht, den Zugriff auf ein privates Netzwerk von einem externen Netzwerk aus zu ermöglichen, z. B. über das Internet von Ihrem Computer aus. Linux-Instances befinden sich in einem öffentlichen Subnetz und sind mit einer Sicherheitsgruppe eingerichtet, die den SSH-Zugriff von der Sicherheitsgruppe aus ermöglicht, die mit der zugrunde liegenden Amazon EC2 EC2-Instance verbunden ist, auf der der Bastion-Host ausgeführt wird.

Wir empfehlen, dem Amazon MWAA-Tutorial zu folgen, um einen Linux Bastion Host zu konfigurieren.: [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem Linux Bastion Host](#)

## Verwenden eines Load Balancer (fortgeschritten)

Der folgende Abschnitt zeigt die Konfigurationen, die Sie auf einen [Application Load Balancer](#) anwenden müssen.

1. Zielgruppen. Sie müssen Zielgruppen verwenden, die auf die privaten IP-Adressen Ihres Apache Airflow-Webserver und dessen VPC-Schnittstellenendpunkt verweisen. Wir empfehlen, beide privaten IP-Adressen als Ihre registrierten Ziele anzugeben, da die Verwendung von nur einer Adresse die Verfügbarkeit verringern kann. Weitere Informationen zur Identifizierung der privaten IP-Adressen finden Sie unter [the section called “Identifizieren der privaten IP-Adressen Ihres Apache Airflow-Webserver und seines VPC-Endpunkts”](#).
2. Statuscodes. Wir empfehlen 200, 302 Statuscodes in Ihren Zielgruppeneinstellungen zu verwenden. Andernfalls werden die Ziele möglicherweise als fehlerhaft gekennzeichnet, wenn der VPC-Endpunkt für den Apache Airflow-Webserver mit einem Fehler reagiert. 302 Redirect
3. HTTPS-Listener. Sie müssen den Zielport für den Apache Airflow-Webserver angeben. Beispielsweise:

| Protokoll | Port |
|-----------|------|
| HTTPS     | 443  |

4. Neue ACM-Domäne. Wenn Sie ein SSL/TLS-Zertifikat zuordnen möchten AWS Certificate Manager, müssen Sie eine neue Domain für den HTTPS-Listener für Ihren Load Balancer erstellen.
5. Region des ACM-Zertifikats. Wenn Sie ein SSL/TLS-Zertifikat zuordnen möchten AWS Certificate Manager, müssen Sie es in dieselbe AWS Region hochladen wie Ihre Umgebung. Beispielsweise:



- **Example Region**, in die das Zertifikat hochgeladen werden soll

```
aws acm import-certificate --certificate fileb://Certificate.pem --certificate-chain fileb://CertificateChain.pem --private-key fileb://PrivateKey.pem --
region us-west-2
```

## Erstellung der erforderlichen VPC-Serviceendpunkte in einer Amazon VPC mit privatem Routing

Ein vorhandenes Amazon VPC-Netzwerk ohne Internetzugang benötigt zusätzliche VPC-Serviceendpunkte (AWS PrivateLink), um Apache Airflow auf Amazon Managed Workflows for Apache Airflow zu verwenden. Auf dieser Seite werden die VPC-Endpunkte beschrieben, die für die von Amazon MWAA verwendeten AWS Services erforderlich sind, die VPC-Endpunkte, die für Apache Airflow erforderlich sind, und wie die VPC-Endpunkte erstellt und an eine bestehende Amazon VPC mit privatem Routing angehängt werden.

### Inhalt

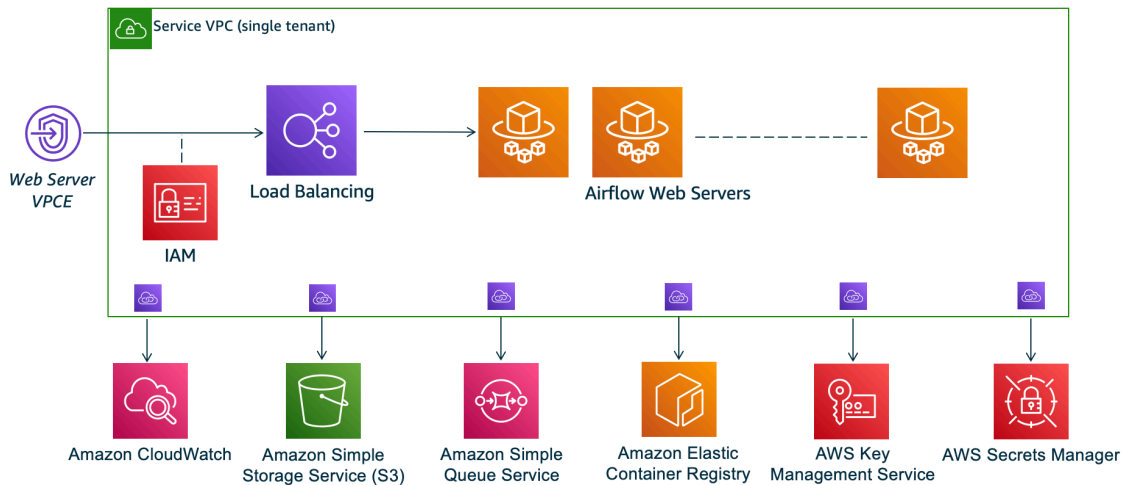
- [Preisgestaltung](#)
- [Privates Netzwerk und privates Routing](#)
- [\(Erforderlich\) VPC-Endpunkte](#)
- [Anhängen der erforderlichen VPC-Endpoints](#)
  - [Für Dienste erforderliche VPC-Endpunkte AWS](#)
  - [Für Apache Airflow sind VPC-Endpunkte erforderlich](#)
- [\(Optional\) Aktivieren Sie private IP-Adressen für Ihren Amazon S3 S3-VPC-Schnittstellenendpunkt](#)
  - [Route 53 verwenden](#)
  - [VPCs mit benutzerdefiniertem DNS](#)

## Preisgestaltung

- [AWS PrivateLink Preise](#)

## Privates Netzwerk und privates Routing

### Private Web Server Option



Der private Netzwerkzugriffsmodus beschränkt den Zugriff auf die Apache Airflow-Benutzeroberfläche auf Benutzer in Ihrer Amazon VPC, denen Zugriff auf die [IAM-Richtlinie für Ihre Umgebung](#) gewährt wurde.

Wenn Sie eine Umgebung mit privatem Webserverzugriff erstellen, müssen Sie alle Ihre Abhängigkeiten in einem Python-Radarchiv (.whl) verpacken und dann auf das .whl in Ihrem `requirements.txt` verweisen. Anweisungen zum Paketieren und Installieren Ihrer Abhängigkeiten mit Wheel finden Sie unter [Abhängigkeiten mit Python Wheel verwalten](#).

Die folgende Abbildung zeigt, wo Sie die Option Privates Netzwerk auf der Amazon MWAA-Konsole finden.

#### Web server access

##### Private network (Recommended)

Additional setup required. Your Airflow UI can only be accessed by secure login behind your VPC. Choose this option if your Airflow UI is only accessed within a corporate network. IAM must be used to handle user authentication.

##### Public network (No additional setup)

Your Airflow UI can be accessed by secure login over the Internet. Choose this option if your Airflow UI is accessed outside of a corporate network. IAM must be used to handle user authentication.

- Privates Routing. Eine [Amazon-VPC ohne Internetzugang schränkt den](#) Netzwerkverkehr innerhalb der VPC ein. Auf dieser Seite wird davon ausgegangen, dass Ihre Amazon VPC keinen Internetzugang hat und VPC-Endpunkte für jeden von Ihrer Umgebung genutzten AWS Service

sowie VPC-Endpunkte für Apache Airflow in derselben AWS Region und Amazon VPC wie Ihre Amazon MWAA-Umgebung benötigt.

## (Erforderlich) VPC-Endpunkte

Der folgende Abschnitt zeigt die erforderlichen VPC-Endpunkte, die für eine Amazon-VPC ohne Internetzugang benötigt werden. Es listet die VPC-Endpunkte für jeden AWS Service auf, der von Amazon MWAA verwendet wird, einschließlich der VPC-Endpunkte, die für Apache Airflow benötigt werden.

```
com.amazonaws.YOUR_REGION.s3
com.amazonaws.YOUR_REGION.monitoring
com.amazonaws.YOUR_REGION.ecr.dkr
com.amazonaws.YOUR_REGION.ecr.api
com.amazonaws.YOUR_REGION.logs
com.amazonaws.YOUR_REGION.sqs
com.amazonaws.YOUR_REGION.kms
com.amazonaws.YOUR_REGION.airflow.api
com.amazonaws.YOUR_REGION.airflow.env
com.amazonaws.YOUR_REGION.airflow.ops
```

## Anhängen der erforderlichen VPC-Endpoints

In diesem Abschnitt werden die Schritte zum Anhängen der erforderlichen VPC-Endpunkte für eine Amazon-VPC mit privatem Routing beschrieben.

### Für Dienste erforderliche VPC-Endpunkte AWS

Der folgende Abschnitt zeigt die Schritte zum Anhängen der VPC-Endpunkte für die von einer Umgebung verwendeten AWS Services an eine bestehende Amazon VPC.

So fügen Sie VPC-Endpunkte an Ihre privaten Subnetze an

1. Öffnen Sie die [Seite Endpoints](#) in der Amazon VPC-Konsole.
2. Verwenden Sie die AWS Regionsauswahl, um Ihre Region auszuwählen.
3. Erstellen Sie den Endpunkt für Amazon S3:
  - a. Klicken Sie auf Endpunkt erstellen.

- b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwörtern suchen: ein **.s3** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
- c. Wir empfehlen, den für den Gateway-Typ aufgelisteten Dienstendpunkt auszuwählen.

Beispiel: **com.amazonaws.us-west-2.s3 amazon Gateway**

- d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass privates DNS aktiviert ist, indem Sie DNS-Name aktivieren auswählen.
  - f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.
4. Erstellen Sie den ersten Endpunkt für Amazon ECR:
- a. Klicken Sie auf Endpunkt erstellen.
  - b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwörtern suchen: ein **.ecr.dkr** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
  - c. Wählen Sie den Dienstendpunkt aus.
  - d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass die Option DNS-Name aktivieren aktiviert ist.
  - f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.
5. Erstellen Sie den zweiten Endpunkt für Amazon ECR:
- a. Klicken Sie auf Endpunkt erstellen.
  - b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwörtern suchen: ein **.ecr.api** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
  - c. Wählen Sie den Dienstendpunkt aus.
  - d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass die Option DNS-Name aktivieren aktiviert ist.

- f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.
6. Erstellen Sie den Endpunkt für CloudWatch Logs:
- a. Klicken Sie auf Endpunkt erstellen.
  - b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwort suchen: ein **.logs** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
  - c. Wählen Sie den Dienstendpunkt aus.
  - d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass die Option DNS-Name aktivieren aktiviert ist.
  - f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.
7. Erstellen Sie den Endpunkt für die CloudWatch Überwachung:
- a. Klicken Sie auf Endpunkt erstellen.
  - b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwort suchen: ein **.monitoring** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
  - c. Wählen Sie den Dienstendpunkt aus.
  - d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass die Option DNS-Name aktivieren aktiviert ist.
  - f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.
8. Erstellen Sie den Endpunkt für Amazon SQS:
- a. Klicken Sie auf Endpunkt erstellen.
  - b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwörtern suchen: ein **.sqs** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
  - c. Wählen Sie den Dienstendpunkt aus.

- d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass die Option DNS-Name aktivieren aktiviert ist.
  - f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.
9. Erstellen Sie den Endpunkt für AWS KMS:
- a. Klicken Sie auf Endpunkt erstellen.
  - b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwort suchen: ein **.kms** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
  - c. Wählen Sie den Dienstendpunkt aus.
  - d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass die Option DNS-Name aktivieren aktiviert ist.
  - f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.

## Für Apache Airflow sind VPC-Endpunkte erforderlich

Der folgende Abschnitt zeigt die Schritte zum Anhängen der VPC-Endpunkte für Apache Airflow an eine bestehende Amazon VPC.

So fügen Sie VPC-Endpunkte an Ihre privaten Subnetze an

1. Öffnen Sie die [Seite Endpoints](#) in der Amazon VPC-Konsole.
2. Verwenden Sie die AWS Regionsauswahl, um Ihre Region auszuwählen.
3. Erstellen Sie den Endpunkt für die Apache Airflow API:
  - a. Klicken Sie auf Endpunkt erstellen.
  - b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwörtern suchen: ein **.airflow.api** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
  - c. Wählen Sie den Dienstendpunkt aus.

- d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass die Option DNS-Name aktivieren aktiviert ist.
  - f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.
4. Erstellen Sie den ersten Endpunkt für die Apache Airflow-Umgebung:
- a. Klicken Sie auf Endpunkt erstellen.
  - b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwörtern suchen: ein **.airflow.env** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
  - c. Wählen Sie den Dienstendpunkt aus.
  - d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass die Option DNS-Name aktivieren aktiviert ist.
  - f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.
5. Erstellen Sie den zweiten Endpunkt für Apache Airflow-Operationen:
- a. Klicken Sie auf Endpunkt erstellen.
  - b. Geben Sie in das Textfeld Nach Attributen filtern oder nach Schlüsselwörtern suchen: ein **.airflow.ops** und drücken Sie dann die Eingabetaste auf Ihrer Tastatur.
  - c. Wählen Sie den Dienstendpunkt aus.
  - d. Wählen Sie die Amazon VPC Ihrer Umgebung in VPC.
  - e. Stellen Sie sicher, dass Ihre beiden privaten Subnetze in verschiedenen Availability Zones ausgewählt sind und dass die Option DNS-Name aktivieren aktiviert ist.
  - f. Wählen Sie die Amazon VPC-Sicherheitsgruppe (n) Ihrer Umgebung aus.
  - g. Wählen Sie unter Richtlinie die Option Vollzugriff aus.
  - h. Wählen Sie Endpunkt erstellen aus.

## (Optional) Aktivieren Sie private IP-Adressen für Ihren Amazon S3 S3-VPC-Schnittstellenendpunkt

Amazon S3 Interface-Endpunkte unterstützen kein privates DNS. Die S3-Endpunktanfragen werden immer noch zu einer öffentlichen IP-Adresse aufgelöst. Um die S3-Adresse in eine private IP-Adresse aufzulösen, müssen Sie [in Route 53 eine private gehostete Zone](#) für den regionalen S3-Endpunkt hinzufügen.

### Route 53 verwenden

In diesem Abschnitt werden die Schritte zum Aktivieren privater IP-Adressen für einen S3-Interface-Endpunkt mithilfe von Route 53 beschrieben.

1. Erstellen Sie eine private gehostete Zone für Ihren Amazon S3 S3-VPC-Schnittstellenendpunkt (z. B. `s3.eu-west-1.amazonaws.com`) und verknüpfen Sie sie mit Ihrer Amazon VPC.
2. Erstellen Sie einen ALIAS A-Eintrag für Ihren Amazon S3 S3-VPC-Schnittstellenendpunkt (z. B. `s3.eu-west-1.amazonaws.com`), der in den DNS-Namen Ihres VPC-Schnittstellen-Endpunkts aufgelöst wird.
3. Erstellen Sie einen ALIAS. Ein Platzhaltereintrag für Ihren Amazon S3 S3-Schnittstellenendpunkt (z. B. `*.s3.eu-west-1.amazonaws.com`), der in den DNS-Namen des VPC-Schnittstellenendpunkts aufgelöst wird.

### VPCs mit benutzerdefiniertem DNS

Wenn Ihre Amazon VPC benutzerdefiniertes DNS-Routing verwendet, müssen Sie die Änderungen in Ihrem DNS-Resolver (nicht Route 53, normalerweise eine EC2-Instance, auf der ein DNS-Server läuft) vornehmen, indem Sie einen CNAME-Eintrag erstellen. Beispielsweise:

```
Name: s3.us-west-2.amazonaws.com
Type: CNAME
Value: *.vpce-0f67d23e37648915c-e2q2e2j3.s3.us-west-2.vpce.amazonaws.com
```



# Verwalten Ihrer eigenen Amazon-VPC-Endpunkte auf Amazon MWAA

Amazon MWAA verwendet Amazon-VPC-Endpunkte, um in verschiedene -AWS-Services zu integrieren, die zum Einrichten einer Apache-Airflow-Umgebung erforderlich sind. Die Verwaltung Ihrer eigenen Endpunkte hat zwei primäre Anwendungsfälle:

1. Das bedeutet, dass Sie Apache Airflow-Umgebungen in einer gemeinsam genutzten Amazon VPC erstellen können, wenn Sie ein verwenden, [AWS Organizations](#) um mehrere AWS Konten zu verwalten und Ressourcen gemeinsam zu nutzen.
2. Sie können restriktivere Zugriffsrichtlinien verwenden, indem Sie Ihre Berechtigungen auf die spezifischen Ressourcen beschränken, die Ihre Endpunkte verwenden.

Wenn Sie Ihre eigenen VPC-Endpunkte verwalten möchten, sind Sie dafür verantwortlich, Ihre eigenen Endpunkte für die Umgebung der Datenbank von RDS für PostgreSQL und für den Umgebungs-Webserver zu erstellen.

Weitere Informationen darüber, wie Amazon MWAA Apache Airflow in der Cloud bereitstellt, finden Sie im [Amazon-MWAA-Architekturdiagramm](#).

## Erstellen einer Umgebung in einer freigegebenen Amazon VPC

Wenn Sie verwenden, [AWS Organizations](#) um mehrere AWS Konten zu verwalten, die Ressourcen gemeinsam nutzen, können Sie vom Kunden verwaltete VPC-Endpunkte mit Amazon MWAA verwenden, um Umgebungsressourcen für ein anderes Konto in Ihrer Organisation gemeinsam zu nutzen.

Wenn Sie den freigegebenen VPC-Zugriff konfigurieren, gibt das -Konto, das die Amazon-VPC-Hauptbenutzer (Besitzer) besitzt, die beiden privaten Subnetze, die von Amazon MWAA benötigt werden, für andere Konten (Teilnehmer) frei, die zur selben Organisation gehören. Teilnehmerkonten, die diese Subnetze gemeinsam nutzen, können Umgebungen in der freigegebenen Amazon VPC anzeigen, erstellen, ändern und löschen.

Angenommen, Sie haben ein -Konto, `Owner`, das als Root Konto in der Organisation fungiert und Eigentümer der Amazon-VPC-Ressourcen ist, und ein Teilnehmerkonto, `Participant`, ein Mitglied derselben Organisation. Wenn eine neue Amazon MWAA in Amazon VPC `Participant` erstellt, die

es mit `teiltOwner`, erstellt Amazon MWAA zuerst die Service-VPC-Ressourcen und wechselt dann bis zu 72 Stunden lang in einen [-PENDING](#)Status.

Nachdem sich der Umgebungsstatus von `CREATING` in geändert hat `PENDING`, `Owner` erstellt ein Prinzipal, der im Namen von `handelt`, die erforderlichen Endpunkte. Dazu listet Amazon MWAA die Datenbank und den Webserver-Endpunkt in der Amazon MWAA-Konsole auf. Sie können auch die [GetEnvironment](#) -API-Aktion aufrufen, um die Service-Endpunkte abzurufen.

### Note

Wenn es sich bei der Amazon VPC, die Sie zum Freigeben von Ressourcen verwenden, um eine private Amazon VPC handelt, müssen Sie dennoch die unter beschriebenen Schritte ausführen [the section called “Verwaltung des Zugriffs auf VPC-Endpunkte”](#). Das Thema behandelt die Einrichtung eines anderen Satzes von Amazon-VPC-Endpunkten im Zusammenhang mit anderen -AWS Services, in die AWS integriert werden kann, wie Amazon ECR, Amazon ECS und Amazon SQS . Diese Services sind für den Betrieb und die Verwaltung Ihrer Apache Airflow-Umgebung in der Cloud von entscheidender Bedeutung.

## Voraussetzungen

Bevor Sie eine Amazon MWAA-Umgebung in einer gemeinsam genutzten VPC erstellen, benötigen Sie die folgenden Ressourcen:

- Ein -AWS Konto, das als Konto verwendet werden `Owner` soll, das Eigentümer der Amazon VPC ist.
- Eine [AWS Organizations](#) Organisationseinheit, die als `Root-MyOrganization` erstellt wurde.
- Ein zweites AWS Konto, `Participant`, unter `MyOrganization` um das Teilnehmerkonto zu bedienen, das die neue Umgebung erstellt.

Darüber hinaus empfehlen wir Ihnen, sich mit den [Verantwortlichkeiten und Berechtigungen für Eigentümer und Teilnehmer](#) vertraut zu machen, wenn Sie Ressourcen in Amazon VPC teilen.

## Erstellen der Amazon VPC

Erstellen Sie zunächst eine neue Amazon VPC, die die Eigentümer- und Teilnehmerkonten gemeinsam nutzen werden:

1. Melden Sie sich mit bei der -Konsole an Owner und öffnen Sie dann die -AWS CloudFormationKonsole. Verwenden Sie die folgende Vorlage, um einen Stack zu erstellen. Dieser Stack stellt eine Reihe von Netzwerkressourcen bereit, darunter eine Amazon VPC, und die Subnetze, die die beiden Konten in diesem Szenario gemeinsam nutzen werden.

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

```
Description: >-
```

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

```
Parameters:
```

```
EnvironmentName:
```

```
Description: An environment name that is prefixed to resource names
```

```
Type: String
```

```
Default: mwaa-
```

```
VpcCIDR:
```

```
Description: Please enter the IP range (CIDR notation) for this VPC
```

```
Type: String
```

```
Default: 10.192.0.0/16
```

```
PublicSubnet1CIDR:
```

```
Description: >-
```

```
Please enter the IP range (CIDR notation) for the public subnet in the first Availability Zone
```

```
Type: String
```

```
Default: 10.192.10.0/24
```

```
PublicSubnet2CIDR:
```

```
Description: >-
```

```
Please enter the IP range (CIDR notation) for the public subnet in the second Availability Zone
```

```
Type: String
```

```
Default: 10.192.11.0/24
```

```
PrivateSubnet1CIDR:
```

```
Description: >-
```

```
Please enter the IP range (CIDR notation) for the private subnet in the first Availability Zone
```

```
Type: String
```

```
Default: 10.192.20.0/24
```

```
PrivateSubnet2CIDR:
```

```
Description: >-
```

```
Please enter the IP range (CIDR notation) for the private subnet in the second Availability Zone
```

```
Type: String
```

```
Default: 10.192.21.0/24
```

```
Resources:
```

```
VPC:
```

```
Type: 'AWS::EC2::VPC'
```

```
Properties:
```

```
CidrBlock: !Ref VpcCIDR
```

```
EnableDnsSupport: true
```

```
EnableDnsHostnames: true
```

```
Tags:
```

```
- Key: Name
```

```
Value: !Ref EnvironmentName
```

```
InternetGateway:
```

```
Type: 'AWS::EC2::InternetGateway'
```

```
Properties:
```

```
Tags:
```

```
- Key: Name
```

```
Value: !Ref EnvironmentName
```

```
InternetGatewayAttachment:
```

```
Type: 'AWS::EC2::VPCGatewayAttachment'
```

```
Properties:
```

```
InternetGatewayId: !Ref InternetGateway
```

```
VpcId: !Ref VPC
```

```
PublicSubnet1:
```

```
Type: 'AWS::EC2::Subnet'
```

```
Properties:
```

```
VpcId: !Ref VPC
```

```
AvailabilityZone: !Select
```

```
- 0
```

```
- !GetAZs ''
```

```
CidrBlock: !Ref PublicSubnet1CIDR
```

```
MapPublicIpOnLaunch: true
```

```
Tags:
```

```
- Key: Name
```

```
Value: !Sub '${EnvironmentName} Public Subnet (AZ1)'
```

```
PublicSubnet2:
```

```
Type: 'AWS::EC2::Subnet'
```

```
Properties:
```

```
VpcId: !Ref VPC
```

```
AvailabilityZone: !Select
```

```
- 1
```

```
- !GetAZs ''
```

```
CidrBlock: !Ref PublicSubnet2CIDR
```

```
MapPublicIpOnLaunch: true
```

```
Tags:
```

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

PrivateSubnet1:

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

PrivateSubnet2:

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

NatGateway1EIP:

```
Type: 'AWS::EC2::EIP'
DependsOn: InternetGatewayAttachment
Properties:
 Domain: vpc
```

NatGateway2EIP:

```
Type: 'AWS::EC2::EIP'
DependsOn: InternetGatewayAttachment
Properties:
 Domain: vpc
```

NatGateway1:

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

NatGateway2:

```
Type: 'AWS::EC2::NatGateway'
Properties:
```

```
 AllocationId: !GetAtt NatGateway2EIP.AllocationId
 SubnetId: !Ref PublicSubnet2
PublicRouteTable:
 Type: 'AWS::EC2::RouteTable'
 Properties:
 VpcId: !Ref VPC
 Tags:
 - Key: Name
 Value: !Sub '${EnvironmentName} Public Routes'
DefaultPublicRoute:
 Type: 'AWS::EC2::Route'
 DependsOn: InternetGatewayAttachment
 Properties:
 RouteTableId: !Ref PublicRouteTable
 DestinationCidrBlock: 0.0.0.0/0
 GatewayId: !Ref InternetGateway
PublicSubnet1RouteTableAssociation:
 Type: 'AWS::EC2::SubnetRouteTableAssociation'
 Properties:
 RouteTableId: !Ref PublicRouteTable
 SubnetId: !Ref PublicSubnet1
PublicSubnet2RouteTableAssociation:
 Type: 'AWS::EC2::SubnetRouteTableAssociation'
 Properties:
 RouteTableId: !Ref PublicRouteTable
 SubnetId: !Ref PublicSubnet2
PrivateRouteTable1:
 Type: 'AWS::EC2::RouteTable'
 Properties:
 VpcId: !Ref VPC
 Tags:
 - Key: Name
 Value: !Sub '${EnvironmentName} Private Routes (AZ1)'
DefaultPrivateRoute1:
 Type: 'AWS::EC2::Route'
 Properties:
 RouteTableId: !Ref PrivateRouteTable1
 DestinationCidrBlock: 0.0.0.0/0
 NatGatewayId: !Ref NatGateway1
PrivateSubnet1RouteTableAssociation:
 Type: 'AWS::EC2::SubnetRouteTableAssociation'
 Properties:
 RouteTableId: !Ref PrivateRouteTable1
 SubnetId: !Ref PrivateSubnet1
```

```
PrivateRouteTable2:
 Type: 'AWS::EC2::RouteTable'
 Properties:
 VpcId: !Ref VPC
 Tags:
 - Key: Name
 Value: !Sub '${EnvironmentName} Private Routes (AZ2)'
DefaultPrivateRoute2:
 Type: 'AWS::EC2::Route'
 Properties:
 RouteTableId: !Ref PrivateRouteTable2
 DestinationCidrBlock: 0.0.0.0/0
 NatGatewayId: !Ref NatGateway2
PrivateSubnet2RouteTableAssociation:
 Type: 'AWS::EC2::SubnetRouteTableAssociation'
 Properties:
 RouteTableId: !Ref PrivateRouteTable2
 SubnetId: !Ref PrivateSubnet2
SecurityGroup:
 Type: 'AWS::EC2::SecurityGroup'
 Properties:
 GroupName: maa-security-group
 GroupDescription: Security group with a self-referencing inbound rule.
 VpcId: !Ref VPC
SecurityGroupIngress:
 Type: 'AWS::EC2::SecurityGroupIngress'
 Properties:
 GroupId: !Ref SecurityGroup
 IpProtocol: '-1'
 SourceSecurityGroupId: !Ref SecurityGroup
Outputs:
 VPC:
 Description: A reference to the created VPC
 Value: !Ref VPC
 PublicSubnets:
 Description: A list of the public subnets
 Value: !Join
 - ','
 - - !Ref PublicSubnet1
 - !Ref PublicSubnet2
 PrivateSubnets:
 Description: A list of the private subnets
 Value: !Join
 - ','
```

```
- - !Ref PrivateSubnet1
- - !Ref PrivateSubnet2
PublicSubnet1:
 Description: A reference to the public subnet in the 1st Availability Zone
 Value: !Ref PublicSubnet1
PublicSubnet2:
 Description: A reference to the public subnet in the 2nd Availability Zone
 Value: !Ref PublicSubnet2
PrivateSubnet1:
 Description: A reference to the private subnet in the 1st Availability Zone
 Value: !Ref PrivateSubnet1
PrivateSubnet2:
 Description: A reference to the private subnet in the 2nd Availability Zone
 Value: !Ref PrivateSubnet2
SecurityGroupIngress:
 Description: Security group with self-referencing inbound rule
 Value: !Ref SecurityGroupIngress
```

2. Nachdem die neuen Amazon VPC-Ressourcen bereitgestellt wurden, navigieren Sie zur AWS Resource Access Manager Konsole und wählen Sie dann Ressourcenfreigabe erstellen aus.
3. Wählen Sie die Subnetze, die Sie im ersten Schritt erstellt haben, aus der Liste der verfügbaren Subnetze aus, die Sie mit teilen könnenParticipant.

## Erstellen der -Umgebung

Führen Sie die folgenden Schritte aus, um eine Amazon MWAA-Umgebung mit vom Kunden verwalteten Amazon VPC-Endpunkten zu erstellen.

1. Melden Sie sich mit an Participantund öffnen Sie die Amazon MWAA-Konsole. Führen Sie Schritt 1: Geben Sie Details an, um einen Amazon S3-Bucket, einen DAG-Ordner und Abhängigkeiten für Ihre neue Umgebung anzugeben. Weitere Informationen finden Sie unter [Erste Schritte](#).
2. Wählen Sie auf der Seite Konfigurieren erweiterter Einstellungen unter Netzwerk die Subnetze aus der freigegebenen Amazon VPC aus.
3. Wählen Sie unter Endpunktverwaltung die Option CUSTOMER aus der Dropdown-Liste aus.
4. Behalten Sie die Standardeinstellung für die verbleibenden Optionen auf der Seite bei und wählen Sie dann Umgebung erstellen auf der Seite Überprüfen und erstellen aus.



Die Umgebung beginnt in einem `-CREATING`-Zustand und wechselt dann zu `PENDING`. Wenn die Umgebung `PENDING` ist, notieren Sie sich den Namen des Datenbank-Endpunktsservice und den Namen des Webserver-Endpunktsservice (wenn Sie einen privaten Webserver einrichten) mithilfe der Konsole.

Wenn Sie eine neue Umgebung mit der Amazon MWAA-Konsole erstellen, erstellt Amazon MWAA eine neue Sicherheitsgruppe mit den erforderlichen Regeln für ein- und ausgehenden Datenverkehr. Notieren Sie sich die Sicherheitsgruppen-ID.

Im nächsten Abschnitt `Owner` verwendet die Service-Endpunkte und die Sicherheitsgruppen-ID, um neue Amazon-VPC-Endpunkte in der freigegebenen Amazon VPC zu erstellen.

## Erstellen der Amazon-VPC-Endpunkte

Führen Sie die folgenden Schritte aus, um die erforderlichen Amazon-VPC-Endpunkte für Ihre Umgebung zu erstellen.

1. Melden Sie sich bei der AWS Management Console mit der offenen <https://console.aws.amazon.com/vpc/> Owner an.
2. Wählen Sie im linken Navigationsbereich Sicherheitsgruppen aus und erstellen Sie dann mithilfe der folgenden Regeln für ein- und ausgehenden Datenverkehr eine neue Sicherheitsgruppe in der freigegebenen Amazon VPC:

|           | Typ                   | Protokoll | Source type (Quellentyp) | Quelle                           |
|-----------|-----------------------|-----------|--------------------------|----------------------------------|
| Eingehend | Gesamter Datenverkehr | Alle      | Alle                     | Ihre Umgebungs sicherheitsgruppe |
| Ausgehend | Gesamter Datenverkehr | Alle      | Alle                     | 0.0.0.0/0                        |

### Warning

Das `Owner` Konto muss eine Sicherheitsgruppe im `Owner` Konto einrichten, um Datenverkehr von der neuen Umgebung zur freigegebenen Amazon VPC zuzulassen.

Sie können dies tun, indem Sie eine neue Sicherheitsgruppe erstellen oder eine vorhandene bearbeiten.

- Wählen Sie Endpunkte und erstellen Sie dann neue Endpunkte für die Umgebungsdatenbank und den Webserver (wenn sich der private Modus befindet) unter Verwendung der Endpunktservicenamen aus den vorherigen Schritten. Wählen Sie die freigegebene Amazon VPC, die Subnetze, die Sie für die Umgebung verwendet haben, und die Sicherheitsgruppe der Umgebung aus.

Bei Erfolg wechselt die Umgebung von PENDING zurück zu CREATING und schließlich zu AVAILABLE. Wenn es ist AVAILABLE, können Sie sich bei der Apache Airflow-Konsole anmelden.

## Fehlerbehebung bei freigegebenen Amazon VPC

Verwenden Sie die folgende Referenz, um Probleme zu beheben, die beim Erstellen von Umgebungen in einer freigegebenen Amazon VPC auftreten.

### Umgebung in **CREATE\_FAILED** nach **PENDING** Status

- Stellen Sie sicher, dass die Subnetze Participant für Owner freigibt [AWS Resource Access Manager](#).
- Stellen Sie sicher, dass die Amazon-VPC-Endpunkte für die Datenbank und den Webserver in denselben Subnetzen erstellt werden, die der Umgebung zugeordnet sind.
- Stellen Sie sicher, dass die Sicherheitsgruppe, die mit Ihren Endpunkten verwendet wird, Datenverkehr von den Sicherheitsgruppen zulässt, die für die Umgebung verwendet werden. Das Owner-Konto erstellt Regeln, die auf die Sicherheitsgruppe in Participant als *verweisen `account-number/security-group-id`*:

| Typ                   | Protokoll | Source type (Quellentyp) | Quelle                               |
|-----------------------|-----------|--------------------------|--------------------------------------|
| Gesamter Datenverkehr | Alle      | Alle                     | <i>123456789012 /sg-0909e8e81919</i> |

Weitere Informationen finden Sie unter [Verantwortlichkeiten und Berechtigungen für Besitzer und Teilnehmer](#)

## Umgebung bleibt im **PENDING** Status hängen

Überprüfen Sie jeden VPC-Endpointstatus, um sicherzustellen, dass er lautet `Available`.

Wenn Sie eine Umgebung mit einem privaten Webserver konfigurieren, müssen Sie auch einen Endpunkt für den Webserver erstellen. Wenn die Umgebung in hängen bleibt `PENDING`, kann dies darauf hindeuten, dass der private Webserver-Endpoint fehlt.

### Empfangener **The Vpc Endpoint Service '*vpce-service-name*' does not exist** Fehler

Wenn der folgende Fehler angezeigt wird, überprüfen Sie, ob das Konto, das die Endpunkte in dem Owner Konto erstellt, das Eigentümer der freigegebenen VPC ist:

```
ClientError: An error occurred (InvalidServiceName) when calling the
CreateVpcEndpoint operation:
```

```
The Vpc Endpoint Service 'vpce-service-name' does not exist
```

# Tutorials für Amazon Managed Workflows für Apache Airflow

Dieses Handbuch enthält step-by-step Tutorials zur Verwendung und Konfiguration einer Umgebung von Amazon Managed Workflows für Apache Airflow.

## Themen

- [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem AWS Client VPN](#)
- [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem Linux Bastion Host](#)
- [Tutorial: Beschränken des Zugriffs eines Amazon MWAA-Benutzers auf eine Teilmenge von DAGs](#)
- [Tutorial: Automatisieren der Verwaltung Ihrer eigenen Umgebungsendpunkte auf Amazon MWAA](#)

## Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem AWS Client VPN

Dieses Tutorial führt Sie durch die Schritte zum Erstellen eines VPN-Tunnels von Ihrem Computer zum Apache Airflow-Webserver für Ihre Amazon Managed Workflows for Apache Airflow-Umgebung. Um über einen VPN-Tunnel eine Verbindung zum Internet herzustellen, müssen Sie zunächst einen AWS Client VPN Endpunkt erstellen. Nach der Einrichtung fungiert ein Client-VPN-Endpunkt als VPN-Server, der eine sichere Verbindung von Ihrem Computer zu den Ressourcen in Ihrer VPC ermöglicht. Anschließend stellen Sie von Ihrem Computer aus mithilfe des [AWS Client VPN für Desktop eine Verbindung zum](#) Client VPN her.

### Abschnitte

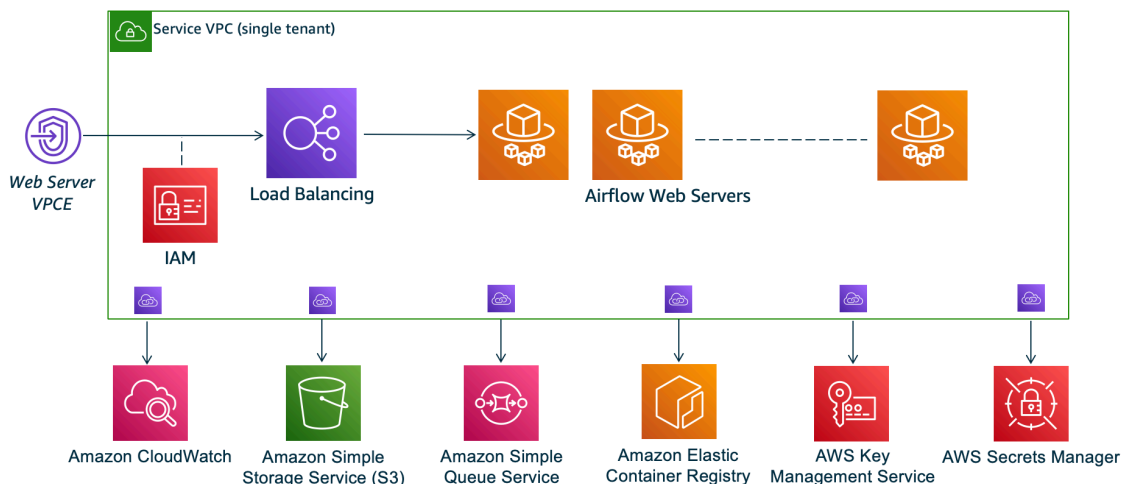
- [Privates Netzwerk](#)
- [Anwendungsfälle](#)
- [Bevor Sie beginnen](#)
- [Zielsetzungen](#)
- [\(Optional\) Erster Schritt: Identifizieren Sie Ihre VPC, CIDR-Regeln und VPC-Sicherheit \(en\)](#)
- [Schritt 2: Erstellen der Server- und Client-Zertifikate](#)
- [Schritt drei: Speichern Sie die AWS CloudFormation Vorlage lokal](#)
- [Schritt vier: Erstellen Sie den AWS CloudFormation Client-VPN-Stack](#)
- [Schritt fünf: Subnetze mit Ihrem Client VPN verknüpfen](#)

- [Schritt 6: Fügen Sie Ihrem Client VPN eine Autorisierungsregel für den Eingang hinzu](#)
- [Schritt 7: Herunterladen der Server VPN-Endpunkt-Konfigurationsdatei](#)
- [Schritt acht: Stellen Sie eine Verbindung zum herAWS Client VPN](#)
- [Als nächstes](#)

## Privates Netzwerk

In diesem Tutorial wird davon ausgegangen, dass Sie den privaten Netzwerkzugriffsmodus für Ihren Apache Airflow-Webserver ausgewählt haben.

### Private Web Server Option



Der private Netzwerkzugriffsmodus beschränkt den Zugriff auf die Apache Airflow-Benutzeroberfläche auf Benutzer innerhalb Ihrer Amazon VPC, denen Zugriff auf die [IAM-Richtlinie für Ihre Umgebung](#) gewährt wurde.

Wenn Sie eine Umgebung mit privatem Webserverzugriff erstellen, müssen Sie alle Ihre Abhängigkeiten in ein Python-Rad-Archiv (.whl) packen und dann .whl in Ihrem auf das `requirements.txt` verweisen. Anweisungen zum Packen und Installieren Ihrer Abhängigkeiten mithilfe von Wheel finden Sie unter [Abhängigkeiten mit Python-Rad verwalten](#).

Die folgende Abbildung zeigt, wo Sie die Option Privates Netzwerk auf der Amazon MWAA-Konsole finden.

## Web server access

**Private network (Recommended)**

Additional setup required. Your Airflow UI can only be accessed by secure login behind your VPC. Choose this option if your Airflow UI is only accessed within a corporate network. IAM must be used to handle user authentication.

**Public network (No additional setup)**

Your Airflow UI can be accessed by secure login over the Internet. Choose this option if your Airflow UI is accessed outside of a corporate network. IAM must be used to handle user authentication.

## Anwendungsfälle

Sie können dieses Tutorial verwenden, bevor oder nachdem Sie eine Amazon MWAA-Umgebung erstellt haben. Sie müssen dieselbe Amazon VPC, dieselbe VPC-Sicherheitsgruppe (n) und dieselben privaten Subnetze wie Ihre Umgebung verwenden. Wenn Sie dieses Tutorial verwenden, nachdem Sie eine Amazon MWAA-Umgebung erstellt haben, können Sie nach Abschluss der Schritte zur Amazon MWAA-Konsole zurückkehren und den Zugriffsmodus Ihres Apache Airflow-Webservers auf Privates Netzwerk ändern.

## Bevor Sie beginnen

1. Suchen Sie nach Benutzerberechtigungen. Stellen Sie sicher, dass Ihr Konto in AWS Identity and Access Management (IAM) über ausreichende Berechtigungen zum Erstellen und Verwalten von VPC-Ressourcen verfügt.
2. Verwenden Sie Ihre Amazon MWAA VPC. In diesem Tutorial wird davon ausgegangen, dass Sie das Client VPN einer vorhandenen VPC zuordnen. Die Amazon VPC muss sich in derselben AWS Region wie eine Amazon MWAA-Umgebung befinden und über zwei private Subnetze verfügen. Wenn Sie noch keine Amazon VPC erstellt haben, verwenden Sie die AWS CloudFormation Vorlage in [Option drei: Erstellen eines Amazon VPC-Netzwerks ohne Internetzugang](#).

## Zielsetzungen

In diesem Tutorial führen Sie folgende Aufgaben durch:

1. Erstellen Sie einen AWS Client VPN Endpunkt mithilfe einer AWS CloudFormation Vorlage für eine bestehende Amazon VPC.

2. Generieren Server- und Client-Zertifikate und Client-Zertifikate und laden Sie dann das Serverzertifikat und der SchlüsselAWS Certificate Manager in dieselbeAWS Region in -Tags hoch.
3. Laden Sie eine Client-VPN-Endpunktkonfigurationsdatei für Ihr Client VPN herunter, ändern Sie sie und verwenden Sie die Datei, um ein VPN-Profil für die Verbindung mit dem Client VPN für Desktop zu erstellen.

## (Optional) Erster Schritt: Identifizieren Sie Ihre VPC, CIDR-Regeln und VPC-Sicherheit (en)

Im folgenden Abschnitt wird beschrieben, wie Sie IDs für Ihre Amazon VPC und VPC-Sicherheitsgruppe finden und wie Sie die CIDR-Regeln identifizieren, die Sie in den nachfolgenden Schritten zum Erstellen Ihres Client VPN benötigen.

### Identifizieren Sie Ihre CIDR-Regeln

Der folgende Abschnitt zeigt, wie Sie die CIDR-Regeln identifizieren, die Sie benötigen, um Ihr Client VPN zu erstellen.

Um den CIDR für Ihr Client VPN zu identifizieren

1. Öffnen [Sie die Seite Ihre Amazon VPCs](#) in der Amazon VPC-Konsole.
2. Verwenden Sie die Regionsauswahl in der Navigationsleiste, um dieselbeAWS Region wie eine Amazon MWAA-Umgebung auszuwählen.
3. Wählen Sie Ihre Amazon VPC.
4. Angenommen, die CIDRs für Ihre privaten Subnetze sind:
  - Privates Subnetz 1:10.192.10.0/24
  - Privates Subnetz 2:10.192.11.0/24

Wenn der CIDR für Ihre Amazon VPC 10.192.0.0 ist/16, dann wäre der Client-IPv4-CIDR, den Sie für Ihr Client VPN angeben würden, 10.192.0.0/22.

5. Speichern Sie diesen CIDR-Wert und den Wert Ihrer VPC-ID für nachfolgende Schritte.

## Identifizieren Sie Ihre VPC und Sicherheitsgruppe (n)

Der folgende Abschnitt zeigt, wie Sie die ID Ihrer Amazon VPC und Sicherheitsgruppe (n) finden, die Sie benötigen, um Ihr Client VPN zu erstellen.

### Note

Möglicherweise verwenden Sie mehr als eine Sicherheitsgruppe. In den nachfolgenden Schritten müssen Sie alle Sicherheitsgruppen Ihrer VPC angeben.

Um die Sicherheitsgruppe (n) zu identifizieren

1. Öffnen Sie die [Seite Sicherheitsgruppen](#) in der Amazon VPC-Konsole.
2. Verwenden Sie die Regionsauswahl in der Navigationsleiste, um die AWS Region auszuwählen.
3. Suchen Sie in der VPC-ID nach der Amazon VPC und identifizieren Sie die Sicherheitsgruppen, die der VPC zugeordnet sind.
4. Speichern Sie die ID Ihrer Sicherheitsgruppe (n) und Ihrer VPC für nachfolgende Schritte.

## Schritt 2: Erstellen der Server- und Client-Zertifikate

Client VPN-Endpunkte unterstützen bei RSA nur Schlüsselgrößen von 1024-Bit und 2048-Bit. Im folgenden Abschnitt wird zum Generieren der OpenVPN und Client-Zertifikate sowie der Server- und Client-Zertifikate sowie der Schlüssel verwendet. Anschließend wird die Zertifikate unter Verwendung der AWS Command Line Interface (AWS CLI) nach ACM hochgeladen.

Um die Client-Zertifikate zu erstellen

1. Folgen Sie diesen schnellen Schritten, um die Zertifikate zu erstellen und über den AWS CLI Link [Client-Authentifizierung und -Autorisierung: Gegenseitige Authentifizierung](#) auf ACM hochzuladen.
2. In diesen Schritten müssen Sie beim Hochladen Ihrer Server- und Client-Zertifikate im AWS CLI Befehl dieselbe AWS Region wie eine Amazon MWAA-Umgebung angeben. Hier sehen Sie einige Beispiele dafür, wie Sie die Region in diesen Befehlen angeben:



a. Example Region für Serverzertifikat

```
aws acm import-certificate --certificate fileb://server.crt --private-key
fileb://server.key --certificate-chain fileb://ca.crt --region us-west-2
```

b. Example Region für das Client-Zertifikat

```
aws acm import-certificate --certificate fileb://client1.domain.tld.crt
--private-key fileb://client1.domain.tld.key --certificate-chain fileb://
ca.crt --region us-west-2
```

c. Speichern Sie nach diesen Schritten den Wert, der in der AWS CLI Antwort für die ARNs des Serverzertifikats und des Client-Zertifikats zurückgegeben wurde. Sie geben diese ARNs in Ihrer AWS CloudFormation Vorlage an, um das Client VPN zu erstellen.

3. In diesen Schritten werden ein Client-Zertifikat und ein privater Schlüssel auf Ihrem Computer gespeichert. Hier sehen Sie ein Beispiel dafür, wo Sie diese Anmeldeinformationen finden:

a. Example unter macOS

Unter macOS werden die Inhalte unter gespeichert/Users/*youruser*/custom\_folder. Wenn Sie alle (`ls -a`)-Inhalte dieses Verzeichnisses auflisten, sollten Sie etwas Ähnliches wie das Folgende sehen:

```
.
..
ca.crt
client1.domain.tld.crt
client1.domain.tld.key
server.crt
server.key
```

b. Speichern Sie nach diesen Schritten den Inhalt oder notieren Sie sich den Speicherort des Client-Zertifikats und den privaten Schlüssel `client1.domain.tld.key`. Sie werden diese Werte zur Konfigurationsdatei für Ihr Client VPN hinzufügen.

## Schritt drei: Speichern Sie dieAWS CloudFormation Vorlage lokal

Der folgende Abschnitt enthält dieAWS CloudFormation Vorlage zum Erstellen des Client VPN. Sie müssen dieselbe Amazon VPC, VPC-Sicherheitsgruppe (n) und dieselben privaten Subnetze wie Ihre Amazon MWAA-Umgebung angeben.

- Kopieren Sie den Inhalt der folgenden Vorlage und speichern Sie es lokal unter `mwaavpnclient.yaml`. Sie können [die Vorlage auch herunterladen](#).

Ersetzen Sie die folgenden Werte:

- **YOUR\_CLIENT\_ROOT\_CERTIFICATE\_ARN**— Der ARN für Ihr `client1.domain.tld`-Zertifikat in `ClientRootCertificateChainArn`.
- **YOUR\_SERVER\_CERTIFICATE\_ARN**— Der ARN für Ihr Serverzertifikat in `ServerCertificateArn`.
- Die Client-IPv4-CIDR-Regel in `ClientCidrBlock`. Eine CIDR-Regel von `10.192.0.0/22` wird bereitgestellt.
- Ihre Amazon VPC-ID in `VpcId`. Eine VPC von `vpc-010101010101` wird bereitgestellt.
- Ihre VPC-Sicherheitsgruppen-ID (n) in `SecurityGroupIds`. Eine Sicherheitsgruppe von `sg-0101010101` wird bereitgestellt.

```
AWSTemplateFormatVersion: 2010-09-09
Description: This template deploys a VPN Client Endpoint.
Resources:
 ClientVpnEndpoint:
 Type: 'AWS::EC2::ClientVpnEndpoint'
 Properties:
 AuthenticationOptions:
 - Type: "certificate-authentication"
 MutualAuthentication:
 ClientRootCertificateChainArn: "YOUR_CLIENT_ROOT_CERTIFICATE_ARN"
 ClientCidrBlock: 10.192.0.0/22
 ClientConnectOptions:
 Enabled: false
 ConnectionLogOptions:
 Enabled: false
 Description: "MWAA Client VPN"
 DnsServers: []
 SecurityGroupIds:
```

```
- sg-0101010101
SelfServicePortal: ''
ServerCertificateArn: "YOUR_SERVER_CERTIFICATE_ARN"
SplitTunnel: true
TagSpecifications:
 - ResourceType: "client-vpn-endpoint"
 Tags:
 - Key: Name
 Value: MAAA-Client-VPN
TransportProtocol: udp
VpcId: vpc-010101010101
VpnPort: 443
```

### Note

Wenn Sie mehr als eine Sicherheitsgruppe für Ihre Umgebung verwenden, können Sie mehrere Sicherheitsgruppen im folgenden Format angeben:

```
SecurityGroupIds:
 - sg-0112233445566778b
 - sg-0223344556677889f
```

## Schritt vier: Erstellen Sie den AWS CloudFormation Client-VPN-Stack

So erstellen Sie das AWS Client VPN

1. Öffnen Sie die [AWS CloudFormation-Konsole](#).
2. Wählen Sie Vorlage ist fertig, Laden Sie eine Vorlagendatei hoch.
3. Wählen Sie Datei auswählen und wählen Sie `Ihremwaa_vpn_client.yaml` Datei aus.
- 4.
5. Wählen Sie Weiter, Weiter.
6. Wählen Sie die Bestätigung aus, und wählen Sie dann Stapel erstellen.

## Schritt fünf: Subnetze mit Ihrem Client VPN verknüpfen

Um private Subnetze mit dem zu verknüpfenAWS Client VPN

1. Öffnen Sie die [Amazon VPC-Konsole](#).
2. Wählen Sie die Seite Client VPN Endpoints aus.
3. Wählen Sie Ihr Client VPN aus und wählen Sie dann auf der Registerkarte Verknüpfungen die Option Zuordnen.
4. Wählen Sie in der Dropdown-Liste Folgendes aus:
  - Ihre Amazon VPC in VPC.
  - Eines Ihrer privaten Subnetze unter Wählen Sie ein Subnetz aus, das Sie zuordnen möchten.
5. Wählen Sie Associate aus.

### Note

Es dauert einige Minuten, bis die VPC und das Subnetz dem Client VPN zugeordnet werden.

## Schritt 6: Fügen Sie Ihrem Client VPN eine Autorisierungsregel für den Eingang hinzu

Sie müssen Ihrem Client VPN eine Autorisierungsregel für den Eingang hinzufügen, indem Sie die CIDR-Regel für Ihre VPC verwenden. Wenn Sie bestimmte Benutzer oder Gruppen aus Ihrer Active Directory-Gruppe oder Ihrem SAML-basierten Identitätsanbieter (IdP) autorisieren möchten, lesen Sie die [Autorisierungsregeln](#) im Client-VPN-Handbuch.

Um den CIDR zu der hinzuzufügenAWS Client VPN

1. Öffnen Sie die [Amazon VPC-Konsole](#).
2. Wählen Sie die Seite Client VPN Endpoints aus.
3. Wählen Sie Ihr Client VPN aus und wählen Sie dann auf der Registerkarte Autorisierung den Eingang autorisieren.
4. Geben Sie Folgendes an:
  - Die CIDR-Regel Ihrer Amazon VPC im Zielnetzwerk muss aktiviert werden. Beispiel:

```
10.192.0.0/16
```

- Wählen Sie unter Zugriff gewähren für alle Benutzer die Option Zugriff gewähren aus.
  - Geben Sie unter Beschreibung einen aussagekräftigen Namen ein.
5. Wählen Sie Autorisierungsregel hinzufügen.

#### Note

Abhängig von den Netzwerkkomponenten für Ihre Amazon VPC benötigen Sie möglicherweise auch diese Autorisierungsregel für den Eingang in Ihre Network Access Control List (NACL).

## Schritt 7: Herunterladen der Server VPN-Endpunkt-Konfigurationsdatei

So laden Sie die Konfigurationsdatei herunter

1. Folgen Sie diesen schnellen Schritten, um die Client-VPN-Konfigurationsdatei unter [Herunterladen der Client-VPN-Endpunktkonfigurationsdatei](#) herunterzuladen.
2. In diesen Schritten werden Sie aufgefordert, dem DNS-Namen Ihres Client-VPN-Endpunkts eine Zeichenfolge voranzustellen. Ein Beispiel:
  - Example DNS-Name des -Endpunkts

Wenn der DNS-Name Ihres Client VPN VPN-Endpunkts wie folgt aussieht:

```
remote cvpn-endpoint-0909091212aeee1.prod.clientvpn.us-west-1.amazonaws.com 443
```

Sie können eine Zeichenfolge hinzufügen, um Ihren Client-VPN-Endpunkt wie folgt zu identifizieren:

```
remote mwaavpn.cvpn-endpoint-0909091212aeee1.prod.clientvpn.us-west-1.amazonaws.com 443
```

3. In diesen Schritten werden Sie dazu den Inhalt des Client-Zertifikats in `<cert></cert>` -Tags hinzu. Anschließend wird den Inhalt des privaten Schlüssels in `<key></key>` -Tags hinzu. Ein Beispiel:

- a. Öffnen Sie eine Befehlszeile und ändern Sie das Verzeichnis zum Speicherort Ihres Client-Zertifikats und Ihres privaten Schlüssels.
- b. Example macOS `client1.domain.tld.crt`

Um den Inhalt der `client1.domain.tld.crt` Datei auf macOS anzuzeigen, können Sie verwenden `cat client1.domain.tld.crt`.

Kopieren Sie den Wert aus dem Terminal und fügen Sie ihn in `downloaded-client-config.ovpn` wie folgt ein:

```
ZZZ1111dddaBBB
-----END CERTIFICATE-----
</ca>
<cert>
-----BEGIN CERTIFICATE-----
YOUR client1.domain.tld.crt
-----END CERTIFICATE-----
</cert>
```

- c. Example macOS `client1.domain.tld.key`

Um den Inhalt von `client1.domain.tld.key` anzuzeigen, können Sie verwenden `cat client1.domain.tld.key`.

Kopieren Sie den Wert aus dem Terminal und fügen Sie ihn in `downloaded-client-config.ovpn` wie folgt ein:

```
ZZZ1111dddaBBB
-----END CERTIFICATE-----
</ca>
<cert>
-----BEGIN CERTIFICATE-----
YOUR client1.domain.tld.crt
-----END CERTIFICATE-----
</cert>
<key>
-----BEGIN CERTIFICATE-----
YOUR client1.domain.tld.key
-----END CERTIFICATE-----
</key>
```

## Schritt acht: Stellen Sie eine Verbindung zum herAWS Client VPN

Der KundeAWS Client VPN wird kostenlos zur Verfügung gestellt. Sie können Ihren Computer direkt mit verbinden,AWS Client VPN um ein durchgehendes VPN-Erlebnis zu erhalten.

Um eine Verbindung zum Client VPN herzustellen

1. Laden Sie das [AWS Client VPNfür den Desktop](#) herunter und installieren Sie es.
2. Öffnen Sie die AWS Client VPN.
3. Wählen Sie im VPN-Client-Menü Datei, Verwaltung Profile.
4. Wählen Sie Profil hinzufügen und wählen Sie dann `downloaded-client-config.ovpn`.
5. Geben Sie im Feld Anzeigename einen beschreibenden Namen ein.
6. Wählen Sie Profil hinzufügen, Fertig.
7. Wählen Sie Connect (Verbinden) aus.

Nachdem Sie sich mit dem Client VPN verbunden haben, müssen Sie die Verbindung zu anderen VPNs trennen, um alle Ressourcen in Ihrer Amazon VPC sehen zu können.

### Note

Möglicherweise müssen Sie den Client beenden und erneut starten, bevor Sie eine Verbindung herstellen können.

## Als nächstes

- Erfahren Sie unter, wie Sie eine Amazon MWAA-Umgebung erstellen [Beginnen Sie mit Amazon Managed Workflows for Apache Airflow](#). Sie müssen eine Umgebung in derselbenAWS Region wie das Client VPN erstellen und dieselbe VPC, dieselben privaten Subnetze und Sicherheitsgruppe wie das Client VPN verwenden.

# Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem Linux Bastion Host

Dieses Tutorial führt Sie durch die Schritte zum Erstellen eines SSH-Tunnels von Ihrem Computer zum Apache Airflow-Webserver für Ihre Amazon Managed Workflows for Apache Airflow-Umgebung. Es wird davon ausgegangen, dass Sie bereits eine Amazon MWAA-Umgebung erstellt haben. Nach der Einrichtung fungiert ein Linux Bastion Host als Jump-Server, der eine sichere Verbindung von Ihrem Computer zu den Ressourcen in Ihrer VPC ermöglicht. Anschließend verwenden Sie ein SOCKS-Proxy-Management-Add-on, um die Proxyeinstellungen in Ihrem Browser für den Zugriff auf Ihre Apache Airflow-Benutzeroberfläche zu steuern.

## Sections

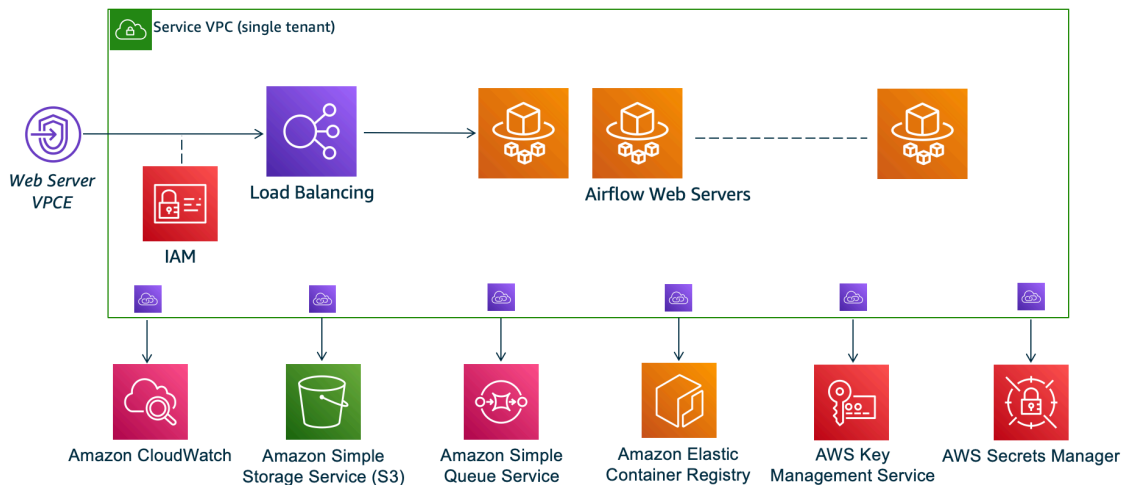
- [Privates Netzwerk](#)
- [Anwendungsfälle](#)
- [Bevor Sie beginnen](#)
- [Ziele](#)
- [Erster Schritt: Erstellen Sie die Bastion-Instanz](#)
- [Schritt zwei: Erstellen Sie den SSH-Tunnel](#)
- [Schritt drei: Konfigurieren Sie die Bastion-Sicherheitsgruppe als Regel für eingehenden Datenverkehr](#)
- [Vierter Schritt: Kopieren Sie die Apache Airflow-URL](#)
- [Fünfter Schritt: Proxyeinstellungen konfigurieren](#)
- [Schritt 6: Öffnen Sie die Apache Airflow-Benutzeroberfläche](#)
- [Als nächstes](#)

## Privates Netzwerk

In diesem Tutorial wird davon ausgegangen, dass Sie den privaten Netzwerkzugriffsmodus für Ihren Apache Airflow-Webserver ausgewählt haben.



## Private Web Server Option



Der private Netzwerkzugriffsmodus beschränkt den Zugriff auf die Apache Airflow-Benutzeroberfläche auf Benutzer in Ihrer Amazon VPC, denen Zugriff auf die [IAM-Richtlinie für Ihre Umgebung](#) gewährt wurde.

Wenn Sie eine Umgebung mit privatem Webserverzugriff erstellen, müssen Sie alle Ihre Abhängigkeiten in einem Python-Radarchiv (.whl) verpacken und dann auf das .whl in Ihrem `requirements.txt` verweisen. Anweisungen zum Paketieren und Installieren Ihrer Abhängigkeiten mit Wheel finden Sie unter [Abhängigkeiten mit Python Wheel verwalten](#).

Die folgende Abbildung zeigt, wo Sie die Option Privates Netzwerk auf der Amazon MWAA-Konsole finden.

### Web server access

#### Private network (Recommended)

Additional setup required. Your Airflow UI can only be accessed by secure login behind your VPC. Choose this option if your Airflow UI is only accessed within a corporate network. IAM must be used to handle user authentication.

#### Public network (No additional setup)

Your Airflow UI can be accessed by secure login over the Internet. Choose this option if your Airflow UI is accessed outside of a corporate network. IAM must be used to handle user authentication.

## Anwendungsfälle

Sie können dieses Tutorial verwenden, nachdem Sie eine Amazon MWAA-Umgebung erstellt haben. Sie müssen dieselbe Amazon VPC, dieselben VPC-Sicherheitsgruppen und dieselben öffentlichen Subnetze wie Ihre Umgebung verwenden.

### Bevor Sie beginnen

1. Suchen Sie nach Benutzerberechtigungen. Stellen Sie sicher, dass Ihr Konto in AWS Identity and Access Management (IAM) über ausreichende Berechtigungen zum Erstellen und Verwalten von VPC-Ressourcen verfügt.
2. Verwenden Sie Ihre Amazon MWAA VPC. In diesem Tutorial wird davon ausgegangen, dass Sie den Bastion-Host einer vorhandenen VPC zuordnen. Die Amazon VPC muss sich in derselben Region wie Ihre Amazon MWAA-Umgebung befinden und über zwei private Subnetze verfügen, wie unter definiert. [Erstellen Sie das VPC-Netzwerk](#)
3. Erstellen Sie einen SSH-Schlüssel. Sie müssen einen Amazon EC2 EC2-SSH-Schlüssel (.pem) in derselben Region wie Ihre Amazon MWAA-Umgebung erstellen, um eine Verbindung zu den virtuellen Servern herzustellen. Wenn Sie keinen SSH-Schlüssel haben, finden Sie weitere Informationen unter [Erstellen oder Importieren eines key pair](#) im Amazon EC2 EC2-Benutzerhandbuch.

### Ziele

In diesem Tutorial führen Sie folgende Aufgaben durch:

1. Erstellen Sie eine Linux Bastion Host-Instanz mithilfe einer [AWS CloudFormation Vorlage für eine bestehende VPC](#).
2. Autorisieren Sie eingehenden Datenverkehr zur Sicherheitsgruppe der Bastion-Instanz mithilfe einer Eingangsregel am Port. 22
3. Autorisieren Sie eingehenden Datenverkehr von der Sicherheitsgruppe einer Amazon MWAA-Umgebung zur Sicherheitsgruppe der Bastion-Instanz.
4. Erstellen Sie einen SSH-Tunnel zur Bastion-Instanz.
5. Installieren und konfigurieren Sie das FoxyProxy Add-on für den Firefox-Browser, um die Apache Airflow-Benutzeroberfläche anzuzeigen.

## Erster Schritt: Erstellen Sie die Bastion-Instanz

Im folgenden Abschnitt werden die Schritte zum Erstellen der Linux-Bastion-Instanz mithilfe einer [AWS CloudFormation Vorlage für eine bestehende VPC](#) auf der AWS CloudFormation Konsole beschrieben.

Um den Linux Bastion Host zu erstellen

1. Öffnen Sie die Seite [Deploy Quick Start](#) auf der AWS CloudFormation Konsole.
2. Verwenden Sie die Regionsauswahl in der Navigationsleiste, um dieselbe AWS Region wie Ihre Amazon MWAA-Umgebung auszuwählen.
3. Wählen Sie Weiter aus.
4. Geben Sie einen Namen in das Textfeld Stackname ein, z. B. `mwa-linux-bastion`
5. Wählen Sie im Bereich Parameter, Netzwerkkonfiguration die folgenden Optionen aus:
  - a. Wählen Sie die VPC-ID Ihrer Amazon MWAA-Umgebung.
  - b. Wählen Sie die Public Subnet 1-ID Ihrer Amazon MWAA-Umgebung.
  - c. Wählen Sie die Public Subnet 2-ID Ihrer Amazon MWAA-Umgebung.
  - d. Geben Sie unter Allowed Bastion External Access CIDR den engstmöglichen Adressbereich (z. B. einen internen CIDR-Bereich) ein.

### Note

Die einfachste Methode, einen Bereich zu identifizieren, besteht darin, denselben CIDR-Bereich wie Ihre öffentlichen Subnetze zu verwenden. Die öffentlichen Subnetze in der AWS CloudFormation Vorlage auf der [Erstellen Sie das VPC-Netzwerk](#) Seite lauten beispielsweise `und. 10.192.10.0/24 10.192.11.0/24`

6. Wählen Sie im Amazon EC2 EC2-Konfigurationsbereich Folgendes aus:
  - a. Wählen Sie Ihren SSH-Schlüssel in der Dropdownliste unter Name des Schlüsselpaars aus.
  - b. Geben Sie einen Namen in das Feld Bastion Host Name ein.
  - c. Wählen Sie „true“ für die TCP-Weiterleitung.

**⚠ Warning**

Die TCP-Weiterleitung muss in diesem Schritt auf `true` gesetzt werden. Andernfalls können Sie im nächsten Schritt keinen SSH-Tunnel erstellen.

7. Wählen Sie Weiter, Weiter aus.
8. Wählen Sie die Bestätigung aus und wählen Sie dann Stack erstellen aus.

Weitere Informationen zur Architektur Ihres Linux Bastion Hosts finden Sie unter [Linux Bastion Hosts on the AWS Cloud: Architektur](#).

## Schritt zwei: Erstellen Sie den SSH-Tunnel

Die folgenden Schritte beschreiben, wie Sie den SSH-Tunnel zu Ihrer Linux-Bastion erstellen. Ein SSH-Tunnel empfängt die Anfrage von Ihrer lokalen IP-Adresse zur Linux-Bastion, weshalb die TCP-Weiterleitung für die Linux-Bastion in den vorherigen Schritten auf TCP-Forwarding für die Linux-Bastion eingestellt wurde. `true`

macOS/Linux

Um einen Tunnel über die Befehlszeile zu erstellen

1. Öffnen Sie die [Instance-Seite](#) auf der Amazon EC2 EC2-Konsole.
2. Wählen einer Instance
3. Kopieren Sie die Adresse in Public IPv4 DNS. z. B. `ec2-4-82-142-1.compute-1.amazonaws.com`.
4. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr SSH-Schlüssel gespeichert ist.
5. Führen Sie den folgenden Befehl aus, um mit SSH eine Verbindung zur Bastion-Instanz herzustellen. Ersetzen Sie den Beispielwert durch Ihren SSH-Schlüsselnamen in. `mykeypair.pem`

```
ssh -i mykeypair.pem -N -D 8157 ec2-user@YOUR_PUBLIC_IPV4_DNS
```

## Windows (PuTTY)

Um einen Tunnel mit PuTTY zu erstellen

1. Öffnen Sie die [Instance-Seite](#) auf der Amazon EC2 EC2-Konsole.
2. Wählen einer Instance
3. Kopieren Sie die Adresse in Public IPv4 DNS. z. B. `ec2-4-82-142-1.compute-1.amazonaws.com`.
4. Öffnen Sie [PuTTY](#) und wählen Sie Sitzung.
5. Geben Sie im Feld Hostname den Hostnamen als `ec2-user@ YOUR_PUBLIC_IPV4_DNS` und den Port als ein. 22
6. Erweitern Sie die Registerkarte SSH und wählen Sie Auth aus. Wählen Sie in der Datei mit dem privaten Schlüssel für die Authentifizierung Ihre lokale „ppk“-Datei aus.
7. Wählen Sie unter SSH die Registerkarte Tunnel und anschließend die Optionen Dynamisch und Automatisch aus.
8. Fügen Sie unter Quellport den 8157 Port (oder einen anderen ungenutzten Port) hinzu und lassen Sie das Feld Zielport leer. Wählen Sie Hinzufügen aus.
9. Wählen Sie die Registerkarte Sitzung und geben Sie einen Sitzungsnamen ein. Zum Beispiel SSH Tunnel.
10. Wählen Sie Speichern, Öffnen.

### Note

Möglicherweise müssen Sie eine Passphrase für Ihren öffentlichen Schlüssel eingeben.

### Note

Wenn Sie eine `Permission denied (publickey)` Fehlermeldung erhalten, empfehlen wir, das Tool [AWSSupport-TroubleshootSSH](#) zu verwenden und Run This Automation (Konsole) auszuwählen, um Fehler bei Ihrer SSH-Einrichtung zu beheben.

## Schritt drei: Konfigurieren Sie die Bastion-Sicherheitsgruppe als Regel für eingehenden Datenverkehr

Der Zugriff auf die Server und der reguläre Internetzugang von den Servern aus sind mit einer speziellen Wartungssicherheitsgruppe möglich, die an diese Server angeschlossen ist. In den folgenden Schritten wird beschrieben, wie die Bastion-Sicherheitsgruppe als eingehende Datenverkehrsquelle zur VPC-Sicherheitsgruppe einer Umgebung konfiguriert wird.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich Netzwerk die Option VPC-Sicherheitsgruppe aus.
4. Wählen Sie Edit inbound rules (Regeln für eingehenden Datenverkehr bearbeiten) aus.
5. Wählen Sie Regel hinzufügen aus.
6. Wählen Sie Ihre VPC-Sicherheitsgruppen-ID in der Dropdownliste Quelle aus.
7. Lassen Sie die übrigen Optionen leer oder setzen Sie sie auf ihre Standardwerte.
8. Wählen Sie Save rules (Regeln speichern) aus.

## Vierter Schritt: Kopieren Sie die Apache Airflow-URL

In den folgenden Schritten wird beschrieben, wie Sie die Amazon MWAA-Konsole öffnen und die URL in die Apache Airflow-Benutzeroberfläche kopieren.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Kopieren Sie die URL in die Airflow-Benutzeroberfläche für nachfolgende Schritte.

## Fünfter Schritt: Proxyeinstellungen konfigurieren

Wenn Sie einen SSH-Tunnel mit dynamischer Port-Weiterleitung verwenden, müssen Sie ein Add-On für die SOCKS-Proxy-Verwaltung einsetzen, um die Proxy-Einstellungen in Ihrem Browser zu steuern. Sie können beispielsweise die `--proxy-server` Funktion von Chromium verwenden, um eine Browsersitzung zu starten, oder die FoxyProxy Erweiterung im Firefox Mozilla-Browser verwenden.

## Option eins: Richten Sie einen SSH-Tunnel mithilfe der lokalen Portweiterleitung ein

Wenn Sie keinen SOCKS-Proxy verwenden möchten, können Sie einen SSH-Tunnel mithilfe der lokalen Portweiterleitung einrichten. Der folgende Beispielbefehl greift auf die Amazon EC2 Resource Manager EC2-Weboberfläche zu, indem er den Datenverkehr auf dem lokalen Port 8157 weiterleitet.

1. Öffnen Sie ein neues Befehlszeilenfenster.
2. Geben Sie den folgenden Befehl ein, um einen SSH-Tunnel zu öffnen.

```
ssh -i mykeypair.pem -N -L 8157:YOUR_VPC_ENDPOINT_ID-
vpce.YOUR_REGION.airflow.amazonaws.com:443
ubuntu@YOUR_PUBLIC_IPV4_DNS.YOUR_REGION.compute.amazonaws.com
```

-L steht für die Verwendung der lokalen Portweiterleitung, mit der Sie einen lokalen Port angeben können, über den Daten an den identifizierten Remote-Port auf dem lokalen Webserver des Knotens weitergeleitet werden.

3. Geben Sie `http://localhost:8157/` Ihren Browser ein.

### Note

Möglicherweise müssen Sie verwenden `https://localhost:8157/`.

## Option zwei: Proxys über die Befehlszeile

In den meisten Webbrowsern können Sie Proxys über eine Befehlszeile oder einen Konfigurationsparameter konfigurieren. Mit Chromium können Sie den Browser beispielsweise mit dem folgenden Befehl starten:

```
chromium --proxy-server="socks5://localhost:8157"
```

Dadurch wird eine Browsersitzung gestartet, die den SSH-Tunnel verwendet, den Sie in den vorherigen Schritten erstellt haben, um ihre Anfragen weiterzuleiten. Sie können Ihre private Amazon MWAA-Umgebungs-URL (mit `https://`) wie folgt öffnen:

```
https://YOUR_VPC_ENDPOINT_ID-vpce.YOUR_REGION.airflow.amazonaws.com/home.
```

## Option drei: Proxys, die für Mozilla Firefox verwendet werden FoxyProxy

Das folgende Beispiel zeigt eine FoxyProxy Standardkonfiguration (Version 7.5.1) für Mozilla Firefox. FoxyProxy stellt eine Reihe von Tools zur Proxyverwaltung bereit. Damit können Sie einen Proxyserver für URLs verwenden, die Mustern entsprechen, die den von der Apache Airflow-Benutzeroberfläche verwendeten Domänen entsprechen.

1. Öffnen Sie in Firefox die Seite mit der [FoxyProxy Standarderweiterung](#).
2. Wählen Sie Zu Firefox hinzufügen.
3. Wählen Sie Hinzufügen aus.
4. Wählen Sie das FoxyProxy Symbol in der Werkzeugleiste Ihres Browsers und dann Optionen.
5. Kopieren Sie den folgenden Code und speichern Sie ihn lokal unter `mwa-proxy.json`. Ersetzen Sie den Beispielwert in ***YOUR\_HOST\_NAME*** durch Ihre Apache Airflow-URL.

```
{
 "e0b7kh1606694837384": {
 "type": 3,
 "color": "#66cc66",
 "title": "airflow",
 "active": true,
 "address": "localhost",
 "port": 8157,
 "proxyDNS": false,
 "username": "",
 "password": "",
 "whitePatterns": [
 {
 "title": "airflow-ui",
 "pattern": "YOUR_HOST_NAME",
 "type": 1,
 "protocols": 1,
 "active": true
 }
],
 "blackPatterns": [],
 "pacURL": "",
 "index": -1
 },
 "k20d21508277536715": {
 "active": true,
 "title": "Default",
```



```
"notes": "These are the settings that are used when no patterns match a URL.",
"color": "#0055E5",
"type": 5,
"whitePatterns": [
 {
 "title": "all URLs",
 "active": true,
 "pattern": "*",
 "type": 1,
 "protocols": 1
 }
],
"blackPatterns": [],
"index": 9007199254740991
},
"logging": {
 "active": true,
 "maxSize": 500
},
"mode": "patterns",
"browserVersion": "82.0.3",
"foxyProxyVersion": "7.5.1",
"foxyProxyEdition": "standard"
}
```

6. Wählen Sie im Bereich „Einstellungen aus FoxyProxy 6.0+ importieren“ die Option „Einstellungen importieren“ und wählen Sie die Datei aus. `mwa-proxy.json`
7. Wählen Sie OK aus.

## Schritt 6: Öffnen Sie die Apache Airflow-Benutzeroberfläche

In den folgenden Schritten wird beschrieben, wie Sie Ihre Apache Airflow-Benutzeroberfläche öffnen.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie „Airflow-Benutzeroberfläche öffnen“.

### Als nächstes

- Erfahren Sie in, wie Sie Airflow CLI-Befehle auf einem SSH-Tunnel zu einem Bastion-Host ausführen. [Befehlsreferenz für Apache Airflow CLI](#)

- Erfahren Sie unter, wie Sie DAG-Code in Ihren Amazon S3 S3-Bucket hochladen [Hinzufügen oder Aktualisieren von DAGs](#).

## Tutorial: Beschränken des Zugriffs eines Amazon MWAA-Benutzers auf eine Teilmenge von DAGs

[Amazon MWAA verwaltet den Zugriff auf Ihre Umgebung, indem es Ihre IAM-Prinzipale einer oder mehreren Standardrollen von Apache Airflow zuordnet](#). Das folgende Tutorial zeigt, wie Sie einzelne Amazon MWAA-Benutzer darauf beschränken können, nur eine bestimmte DAG oder eine Gruppe von DAGs anzuzeigen und mit ihnen zu interagieren.

### Note

Die Schritte in diesem Tutorial können mithilfe des Verbundzugriffs ausgeführt werden, sofern die IAM-Rollen übernommen werden können.

### Themen

- [Voraussetzungen](#)
- [Schritt eins: Gewähren Sie dem Amazon MWAA-Webserver Zugriff auf Ihren IAM-Principal mit der Standardrolle Public Apache Airflow.](#)
- [Schritt zwei: Erstellen Sie eine neue benutzerdefinierte Apache Airflow-Rolle](#)
- [Schritt drei: Weisen Sie Ihrem Amazon MWAA-Benutzer die von Ihnen erstellte Rolle zu](#)
- [Nächste Schritte](#)
- [Zugehörige Ressourcen](#)

## Voraussetzungen

Um die Schritte in diesem Tutorial abzuschließen, benötigen Sie Folgendes:

- Eine [Amazon MWAA-Umgebung mit mehreren DAGs](#)
- Ein IAM-Prinzipal Admin mit [AdministratorAccess](#) Berechtigungen und ein IAM-Benutzer als Principal `MWAAUser1`, für den Sie den DAG-Zugriff einschränken können. Weitere Informationen zu Administratorrollen finden Sie unter [Funktion „Administratorjob“](#) im IAM-Benutzerhandbuch

**Note**

Fügen Sie Ihren IAM-Benutzern keine Berechtigungsrichtlinien direkt hinzu. Wir empfehlen, IAM-Rollen einzurichten, die Benutzer annehmen können, um temporären Zugriff auf Ihre Amazon MWAA-Ressourcen zu erhalten.

- [AWS Command Line Interface Version 2 installiert.](#)

## Schritt eins: Gewähren Sie dem Amazon MWAA-Webserver Zugriff auf Ihren IAM-Principal mit der Standardrolle **Public** Apache Airflow.

Um die Erlaubnis zu erteilen mit dem AWS Management Console

1. Melden Sie sich mit einer Admin Rolle bei Ihrem AWS Konto an und öffnen Sie die [IAM-Konsole](#).
2. Wählen Sie im linken Navigationsbereich Benutzer und dann Ihren Amazon MWAA IAM-Benutzer aus der Benutzertabelle aus.
3. Wählen Sie auf der Seite mit den Benutzerdetails unter Zusammenfassung den Tab Berechtigungen und dann Berechtigungsrichtlinien aus, um die Karte zu erweitern, und klicken Sie dann auf Berechtigungen hinzufügen.
4. Wählen Sie im Abschnitt Berechtigungen gewähren die Option Bestehende Richtlinien direkt anhängen aus und wählen Sie dann Richtlinie erstellen aus, um Ihre eigene benutzerdefinierte Berechtigungsrichtlinie zu erstellen und anzuhängen.
5. Wählen Sie auf der Seite Richtlinie erstellen die Option JSON aus und kopieren Sie dann die folgende JSON-Berechtigungsrichtlinie und fügen Sie sie in den Richtlinieneditor ein. Diese Richtlinie gewährt dem Benutzer mit der Standardrolle Public Apache Airflow Webserverzugriff.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": "airflow:CreateWebLoginToken",
 "Resource": [
 "arn:aws:airflow:YOUR_REGION:YOUR_ACCOUNT_ID:role/YOUR_ENVIRONMENT_NAME/Public"
]
 }
]
}
```

```
]
}
```

## Schritt zwei: Erstellen Sie eine neue benutzerdefinierte Apache Airflow-Rolle

Um eine neue Rolle mit der Apache Airflow-Benutzeroberfläche zu erstellen

1. Öffnen Sie mit Ihrer Administrator-IAM-Rolle die [Amazon MWAA-Konsole](#) und starten Sie die Apache Airflow-Benutzeroberfläche Ihrer Umgebung.
2. Bewegen Sie im Navigationsbereich oben den Mauszeiger auf Sicherheit, um die Dropdownliste zu öffnen, und wählen Sie dann Rollen auflisten, um die Standardrollen von Apache Airflow anzuzeigen.
3. Wählen Sie in der Rollenliste Benutzer und dann oben auf der Seite Aktionen aus, um das Drop-down-Menü zu öffnen. Wählen Sie „Rolle kopieren“ und bestätigen Sie „OK“

### Note

Kopieren Sie die Rollen Ops oder Viewer, um mehr bzw. weniger Zugriff zu gewähren.

4. Suchen Sie in der Tabelle nach der neuen Rolle, die Sie erstellt haben, und wählen Sie Datensatz bearbeiten aus.
5. Gehen Sie auf der Seite „Rolle bearbeiten“ wie folgt vor:
  - Geben Sie unter Name einen neuen Namen für die Rolle in das Textfeld ein. z. B. **Restricted**.
  - Entfernen Sie für die Liste der Berechtigungen die Lese can read on DAGs - und Schreibberechtigungen für die Gruppe von DAGscan edit on DAGs, auf die Sie Zugriff gewähren möchten, und fügen Sie sie anschließend hinzu. Fügen Sie beispielsweise für eine DAG `example_dag.py` **can read on DAG:example\_dag** und **can edit on DAG:example\_dag** hinzu.

Wählen Sie Speichern. Sie sollten jetzt über eine neue Rolle verfügen, die den Zugriff auf eine Teilmenge der in Ihrer Amazon MWAA-Umgebung verfügbaren DAGs einschränkt. Sie können diese Rolle jetzt allen vorhandenen Apache Airflow-Benutzern zuweisen.

## Schritt drei: Weisen Sie Ihrem Amazon MWAA-Benutzer die von Ihnen erstellte Rolle zu

Um die neue Rolle zuzuweisen

1. Führen Sie mit den Zugangsdaten für `MWAAUser` den folgenden CLI-Befehl aus, um die Webserver-URL Ihrer Umgebung abzurufen.

```
$ aws mwaas get-environment --name YOUR_ENVIRONMENT_NAME | jq
' .Environment.WebserverUrl '
```

Bei Erfolg wird die folgende Ausgabe angezeigt:

```
"ab1b2345-678a-90a1-a2aa-34a567a8a901.c13.us-west-2.airflow.amazonaws.com"
```

2. Wenn Sie `MWAAUser` angemeldet sind AWS Management Console, öffnen Sie ein neues Browserfenster und greifen Sie auf die folgende URL zu. `Webserver-URL` Ersetzen Sie es durch Ihre Informationen.

```
https://<Webserver-URL>/home
```

Bei Erfolg wird eine Forbidden Fehlerseite angezeigt, da Ihnen `MWAAUser` noch keine Zugriffsberechtigung für die Apache Airflow-Benutzeroberfläche erteilt wurde.

3. Wenn Sie Admin angemeldet sind AWS Management Console, öffnen Sie erneut die Amazon MWAA-Konsole und starten Sie die Apache Airflow-Benutzeroberfläche Ihrer Umgebung.
4. Erweitern Sie im UI-Dashboard das Drop-down-Menü „Sicherheit“ und wählen Sie dieses Mal „Benutzer auflisten“ aus.
5. Suchen Sie in der Benutzertabelle nach dem neuen Apache Airflow-Benutzer und wählen Sie Datensatz bearbeiten aus. Der Vorname des Benutzers entspricht Ihrem IAM-Benutzernamen im folgenden Muster: `user/mwaa-user`
6. Fügen Sie auf der Seite „Benutzer bearbeiten“ im Abschnitt Rolle die neue benutzerdefinierte Rolle hinzu, die Sie erstellt haben, und wählen Sie dann Speichern aus.

### Note

Das Feld Nachname ist erforderlich, aber ein Leerzeichen erfüllt die Anforderung.

Der Public IAM-Principal erteilt die `MWAAUser` Berechtigung zum Zugriff auf die Apache Airflow-Benutzeroberfläche, während die neue Rolle die zusätzlichen Berechtigungen bereitstellt, die zum Anzeigen ihrer DAGs erforderlich sind.

#### Important

Jede der fünf Standardrollen (z. B. `Admin`), die nicht von IAM autorisiert wurden und die über die Apache Airflow-Benutzeroberfläche hinzugefügt wurden, wird bei der nächsten Benutzeranmeldung entfernt.

## Nächste Schritte

- Weitere Informationen zur Verwaltung des Zugriffs auf Ihre Amazon MWAA-Umgebung und Beispiele für JSON-IAM-Richtlinien, die Sie für Benutzer Ihrer Umgebung verwenden können, finden Sie unter [the section called “Zugreifen auf eine Amazon MWAA-Umgebung”](#)

## Zugehörige Ressourcen

- [Zugriffskontrolle](#) (Apache Airflow-Dokumentation) — Weitere Informationen zu den Standardrollen von Apache Airflow finden Sie auf der Apache Airflow-Dokumentationswebsite.

## Tutorial: Automatisieren der Verwaltung Ihrer eigenen Umgebungsendpunkte auf Amazon MWAA

Wenn Sie verwenden, [AWS Organizations](#) um mehrere AWS Konten zu verwalten, die Ressourcen gemeinsam nutzen, können Sie mit Amazon MWAA Ihre eigenen Amazon-VPC-Endpunkte erstellen und verwalten. Das bedeutet, dass Sie strengere Sicherheitsrichtlinien verwenden können, die nur den Zugriff auf die für Ihre Umgebung erforderlichen Ressourcen ermöglichen.

Wenn Sie eine Umgebung in einer gemeinsam genutzten Amazon VPC erstellen, gibt das Konto, das die Haupt-Arbeit-VPC besitzt (Besitzer), die beiden privaten Subnetze frei, die von Amazon MWAA für andere Konten (Teilnehmer) benötigt werden, die zur selben Organisation gehören.

Teilnehmerkonten, die diese Subnetze gemeinsam nutzen, können dann Umgebungen in der freigegebenen VPC anzeigen, erstellen, ändern und löschen.

Wenn Sie eine Umgebung in einer freigegebenen oder anderweitig durch Richtlinien eingeschränkten VPC erstellen, erstellt Amazon MWAA zuerst die Service-VPC-Ressourcen und gibt dann bis zu 72 Stunden einen [-PENDING](#)Status ein.

Wenn sich der Umgebungsstatus von CREATING zu ändertPENDING, sendet Amazon MWAA eine Amazon- EventBridge Benachrichtigung über die Statusänderung. Auf diese Weise kann das Besitzerkonto die erforderlichen Endpunkte im Namen von Teilnehmern basierend auf Endpunktserviceinformationen aus der Amazon MWAA-Konsole oder API oder programmgesteuert erstellen. Im Folgenden erstellen wir neue Amazon VPC-Endpunkte mithilfe einer Lambda-Funktion und einer EventBridge Regel, die Benachrichtigungen über Amazon MWAA-Statusänderungen abhört.

Hier erstellen wir die neuen Endpunkte in derselben Amazon VPC wie die Umgebung. Um eine freigegebene Amazon VPC einzurichten, erstellen Sie die EventBridge Regel und die Lambda-Funktion würde im Besitzerkonto und die Amazon MWAA-Umgebung im Teilnehmerkonto.

Themen

- [Voraussetzungen](#)
- [Erstellen der Amazon VPC](#)
- [So erstellen Sie die Lambda-Funktion:](#)
- [Erstellen der EventBridge Regel](#)
- [Erstellen der Amazon MWAA-Umgebung](#)

## Voraussetzungen

Um die Schritte in diesem Tutorial auszuführen, benötigen Sie Folgendes:

- ...

## Erstellen der Amazon VPC

Verwenden Sie die folgende AWS CloudFormation Vorlage und den folgenden AWS CLI Befehl, um eine neue Amazon VPC zu erstellen. Die Vorlage richtet die Amazon-VPC-Ressourcen ein und ändert die Endpunktrichtlinie, um den Zugriff auf eine bestimmte Warteschlange einzuschränken.

1. Laden Sie die AWS CloudFormation [Vorlage](#) herunter und entpacken Sie dann die `.yaml` Datei.
2. Navigieren Sie in einem neuen Befehlszeilenfenster zu dem Ordner, in dem Sie die Vorlage gespeichert haben, und verwenden Sie dann `aws cloudformation create-stack` um den Stack zu erstellen. Das `--template-body` Flag gibt den Pfad zur Vorlage an.

```
$ aws cloudformation create-stack --stack-name stack-name --template-body file://
cfn-vpc-private-network.yaml
```

Im nächsten Abschnitt erstellen Sie die Lambda-Funktion.

## So erstellen Sie die Lambda-Funktion:

Verwenden Sie den folgenden Python-Code und die folgende IAM-JSON-Richtlinie, um eine neue Lambda-Funktion und -Ausführungsrolle zu erstellen. Diese Funktion erstellt Amazon-VPC-Endpunkte für einen privaten Apache-Airflow-Webserver und eine Amazon SQS-Warteschlange. Amazon MWAA verwendet Amazon SQS, um Aufgaben mit Bolery unter mehreren Workern in die Warteschlange zu stellen, wenn Ihre Umgebung skaliert wird.

1. Laden Sie den Python-[Funktionscode](#) herunter.
2. Laden Sie die IAM-[Berechtigungsrichtlinie](#) herunter und entpacken Sie dann die Datei.
3. Öffnen Sie eine Eingabeaufforderung und navigieren Sie dann zu dem Ordner, in dem Sie die JSON-Berechtigungsrichtlinie gespeichert haben. Verwenden Sie den IAM-[create-role](#) Befehl, um die neue Rolle zu erstellen.

```
$ aws iam create-role --role-name function-role \
--assume-role-policy-document file://lambda-mwaa-vpce-policy.json
```

Notieren Sie sich den Rollen-ARN aus der AWS CLI Antwort. Im nächsten Schritt geben wir diese neue Rolle mithilfe ihres ARN als Ausführungsrolle der Funktion an.

4. Navigieren Sie zu dem Ordner, in dem Sie den Funktionscode gespeichert haben, und verwenden Sie dann den [create-function](#) Befehl, um eine neue Funktion zu erstellen.

```
$ aws lambda create-function --function-name mwaa-vpce-lambda \
--zip-file file://mwaa-lambda-shared-vpc.zip --runtime python3.8 --role
arn:aws:iam::123456789012:role/function-role --handler lambda_handler
```



Notieren Sie sich den Funktions-ARN aus der AWS CLI Antwort. Im nächsten Schritt geben wir den ARN an, um die Funktion als Ziel für eine neue EventBridge Regel zu konfigurieren.

Im nächsten Abschnitt erstellen Sie die EventBridge Regel, die diese Funktion aufruft, wenn die Umgebung in einen -PENDINGZustand wechselt.

## Erstellen der EventBridge Regel

Gehen Sie wie folgt vor, um eine neue Regel zu erstellen, die auf Amazon MWAA-Benachrichtigungen wartet und auf Ihre neue Lambda-Funktion abzielt.

1. Verwenden Sie den EventBridge `put-rule` Befehl , um eine neue EventBridge Regel zu erstellen.

```
$ aws events put-rule --name "mwa-lambda-rule" \
--event-pattern "{\"source\":[\"aws.airflow\"],\"detail-type\":[\"MWA Environment Status Change\"]}"
```

Das Ereignismuster lauscht auf Benachrichtigungen, die Amazon MWAA sendet, wenn sich ein Umgebungsstatus ändert.

```
{
 "source": ["aws.airflow"],
 "detail-type": ["MWA Environment Status Change"]
}
```

2. Verwenden Sie den `put-targets` Befehl , um die Lambda-Funktion als Ziel für die neue Regel hinzuzufügen.

```
$ aws events put-targets --rule "mwa-lambda-rule" \
--targets "Id"="1", "Arn"="arn:aws::lambda:region:123456789012:function:mwa-vpce-lambda"
```

Sie können eine neue Amazon MWAA-Umgebung mit vom Kunden verwalteten Amazon VPC-Endpunkten erstellen.

## Erstellen der Amazon MWAA-Umgebung

Verwenden Sie die Amazon MWAA-Konsole, um eine neue Umgebung mit vom Kunden verwalteten Amazon VPC-Endpunkten zu erstellen.

1. Öffnen Sie die [Amazon MWAA](#)-Konsole und wählen Sie Umgebung erstellen aus.
2. Geben Sie unter Name einen eindeutigen Namen ein.
3. Wählen Sie für Airflow-Version die neueste Version aus.
4. Wählen Sie einen Amazon S3-Bucket und einen DAGs-Ordner aus, z. B. dags/ zur Verwendung mit der Umgebung, und wählen Sie dann Weiter aus.
5. Gehen Sie auf der Seite Konfigurieren erweiterter Einstellungen wie folgt vor:
  - a. Wählen Sie für Virtual Private Cloud die Amazon VPC aus, die Sie im [vorherigen Schritt](#) erstellt haben.
  - b. Wählen Sie für Webserverzugriff die Option Öffentliches Netzwerk (Internet zugänglich) aus.
  - c. Wählen Sie für Sicherheitsgruppen die Sicherheitsgruppe aus, die Sie mit erstellt habenAWS CloudFormation. Da die Sicherheitsgruppen für die AWS PrivateLink Endpunkte aus dem vorherigen Schritt selbstreferenzieren, müssen Sie dieselbe Sicherheitsgruppe für Ihre Umgebung auswählen.
  - d. Wählen Sie für Endpunktverwaltung die Option Kundenverwaltete Endpunkte aus.
6. Behalten Sie die verbleibenden Standardeinstellungen bei und wählen Sie dann Weiter aus.
7. Überprüfen Sie Ihre Auswahl und wählen Sie dann Umgebung erstellen aus.

### Tip

Weitere Informationen zum Einrichten einer neuen Umgebung finden Sie unter [Erste Schritte mit Amazon MWAA](#).

Wenn die Umgebung istPENDING, sendet Amazon MWAA eine Benachrichtigung, die dem Ereignismuster entspricht, das Sie für Ihre Regel festgelegt haben. Die Regel ruft Ihre Lambda-Funktion auf. Die Funktion analysiert das Benachrichtigungsereignis und ruft die erforderlichen Endpunktinformationen für den Webserver und die Amazon SQS-Warteschlange ab. Anschließend werden die Endpunkte in Ihrer Amazon VPC erstellt.

Wenn die Endpunkte verfügbar sind, setzt Amazon MWAA die Erstellung Ihrer Umgebung fort. Wenn Sie bereit sind, ändert sich der Umgebungsstatus in `AVAILABLE` und Sie können über die Amazon MWAA-Konsole auf den Apache Airflow-Webserver zugreifen.

# Codebeispiele für Amazon Managed Workflows für Apache Airflow

Dieses Handbuch enthält Codebeispiele, einschließlich DAGs und benutzerdefinierter Plugins, die Sie in einer Amazon Managed Workflows for Apache Airflow-Umgebung verwenden können. Weitere Beispiele für die Verwendung von Apache Airflow mit AWS Diensten finden Sie im [dags](#) Verzeichnis im Apache Airflow-Repository. GitHub

## Beispiele

- [Verwenden einer DAG zum Importieren von Variablen in die CLI](#)
- [Erstellen einer SSH-Verbindung mit dem SSHOperator](#)
- [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow Snowflake-Verbindung](#)
- [Verwenden einer DAG zum Schreiben benutzerdefinierter Metriken CloudWatch](#)
- [Aurora-PostgreSQL-Datenbankbereinigung in einer Amazon-MWAA-Umgebung](#)
- [Exportieren von Umgebungsmetadaten in CSV-Dateien auf Amazon S3](#)
- [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Variable](#)
- [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Verbindung](#)
- [Ein benutzerdefiniertes Plugin mit Oracle erstellen](#)
- [Erstellen eines benutzerdefinierten Plugins, das Laufzeitumgebungsvariablen generiert](#)
- [Ändern der Zeitzone einer DAG auf Amazon MWAA](#)
- [Erfrischung eines CodeArtifact Zeichen](#)
- [Erstellen eines benutzerdefinierten Plugins mit Apache Hive und Hadoop](#)
- [Ein benutzerdefiniertes Plugin für Apache Airflow erstellen PythonVirtualenvOperator](#)
- [Aufrufen von DAGs mit einer Lambda-Funktion](#)
- [DAGs in verschiedenen Amazon MWAA-Umgebungen aufrufen](#)
- [Amazon MWAA mit Amazon RDS für Microsoft SQL Server verwenden](#)
- [Amazon MWAA mit Amazon EMR verwenden](#)
- [Amazon MWAA mit Amazon EKS verwenden](#)
- [Herstellen einer Verbindung zu Amazon ECS mithilfe des ECSOperator](#)
- [Verwenden von dbt mit Amazon MWAA](#)

- [AWS Blogs und Tutorials](#)

## Verwenden einer DAG zum Importieren von Variablen in die CLI

Der folgende Beispielcode importiert Variablen mithilfe der CLI in Amazon Managed Workflows for Apache Airflow.

Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Abhängigkeiten](#)
- [Codebeispiel](#)
- [Als nächstes](#)

### Version

- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

### Voraussetzungen

- Für die Verwendung des Codebeispiels auf dieser Seite sind keine zusätzlichen Berechtigungen erforderlich.

### Berechtigungen

Dein AWS-Konto benötigt Zugriff auf `AmazonMWAACliAccess`-Politik. Weitere Informationen hierzu finden Sie unter [Apache Airflow CLI-Richtlinie: AirflowCli AmazonMWAACli Access](#).

### Abhängigkeiten

- Um dieses Codebeispiel mit Apache Airflow v2 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v2-Basisinstallation](#) auf deiner Umgebung.

## Codebeispiel

Der folgende Beispielcode benötigt drei Eingaben: Ihr Amazon MWAA-Umgebungsname (inmwaa\_env), derAWSRegion Ihrer Umgebung (inaws\_region) und die lokale Datei, die die Variablen enthält, die Sie importieren möchten (invar\_file).

```
import boto3
import json
import requests
import base64
import getopt
import sys

argv = sys.argv[1:]
mwaa_env=''
aws_region=''
var_file=''

try:
 opts, args = getopt.getopt(argv, 'e:v:r:', ['environment', 'variable-
file','region'])
 #if len(opts) == 0 and len(opts) > 3:
 if len(opts) != 3:
 print ('Usage: -e MWAA environment -v variable file location and filename -r
aws region')
 else:
 for opt, arg in opts:
 if opt in ("-e"):
 mwaa_env=arg
 elif opt in ("-r"):
 aws_region=arg
 elif opt in ("-v"):
 var_file=arg

 boto3.setup_default_session(region_name="{}".format(aws_region))
 mwaa_env_name = "{}".format(mwaa_env)

 client = boto3.client('mwaa')
 mwaa_cli_token = client.create_cli_token(
 Name=mwaa_env_name
)

 with open ("{}".format(var_file), "r") as myfile:
```

```
fileconf = myfile.read().replace('\n', '')

json_dictionary = json.loads(fileconf)
for key in json_dictionary:
 print(key, " ", json_dictionary[key])
 val = (key + " " + json_dictionary[key])
 mwaa_auth_token = 'Bearer ' + mwaa_cli_token['CliToken']
 mwaa_webserver_hostname = 'https://{0}/aws_mwaa/
cli'.format(mwaa_cli_token['WebServerHostname'])
 raw_data = "variables set {0}".format(val)
 mwaa_response = requests.post(
 mwaa_webserver_hostname,
 headers={
 'Authorization': mwaa_auth_token,
 'Content-Type': 'text/plain'
 },
 data=raw_data
)
 mwaa_std_err_message = base64.b64decode(mwaa_response.json()
['stderr']).decode('utf8')
 mwaa_std_out_message = base64.b64decode(mwaa_response.json()
['stdout']).decode('utf8')
 print(mwaa_response.status_code)
 print(mwaa_std_err_message)
 print(mwaa_std_out_message)

except:
 print('Use this script with the following options: -e MAAA environment -v variable
file location and filename -r aws region')
 print("Unexpected error:", sys.exc_info()[0])
 sys.exit(2)
```

## Als nächstes

- Erfahren Sie, wie Sie den DAG-Code in diesem Beispiel auf `diedagsOrdner` in Ihrem Amazon S3-Bucket in [Hinzufügen oder Aktualisieren von DAGs](#).

## Erstellen einer SSH-Verbindung mit dem **SSHOperator**

Das folgende Beispiel beschreibt, wie Sie die DAG (SSHOperatorIn a Directed Acyclic Graph) verwenden können, um von Ihrer Amazon Managed Workflows for Apache Airflow-Umgebung aus

eine Verbindung zu einer Amazon EC2-Remoteinstanz herzustellen. Sie können einen ähnlichen Ansatz verwenden, um eine Verbindung zu einer beliebigen Remote-Instance mit SSH-Zugriff herzustellen.

Im folgenden Beispiel laden Sie einen geheimen SSH-Schlüssel (.pem) in das dags Verzeichnis Ihrer Umgebung auf Amazon S3 hoch. Anschließend installieren Sie die erforderlichen Abhängigkeiten mithilfe der Benutzeroberfläche `requirements.txt` und erstellen eine neue Apache Airflow-Verbindung. Schließlich schreiben Sie eine DAG, die eine SSH-Verbindung zur Remote-Instanz herstellt.

## Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Voraussetzungen](#)
- [Kopieren Sie Ihren geheimen Schlüssel zu Amazon S3](#)
- [Erstellen Sie eine neue Apache Airflow-Verbindung](#)
- [Codebeispiel](#)

## Version

- Sie können das Codebeispiel auf dieser Seite mit Apache Airflow v2 und höher in [Python 3.10](#) verwenden.

## Voraussetzungen

Um den Beispielcode auf dieser Seite zu verwenden, benötigen Sie Folgendes:

- Eine [Amazon MWAA-Umgebung](#).
- Ein geheimer SSH-Schlüssel. Das Codebeispiel geht davon aus, dass Sie über eine Amazon EC2 EC2-Instance verfügen und sich .pem in derselben Region wie Ihre Amazon MWAA-Umgebung befinden. Wenn Sie keinen Schlüssel haben, finden Sie weitere Informationen unter [Erstellen oder Importieren eines key pair](#) im Amazon EC2 EC2-Benutzerhandbuch.



## Berechtigungen

- Für die Verwendung des Codebeispiels auf dieser Seite sind keine zusätzlichen Berechtigungen erforderlich.

## Voraussetzungen

Fügen Sie den folgenden Parameter hinzu `requirements.txt`, um das `apache-airflow-providers-ssh` Paket auf dem Webserver zu installieren. Sobald Ihre Umgebung aktualisiert wurde und Amazon MWAA die Abhängigkeit erfolgreich installiert hat, wird in der Benutzeroberfläche ein neuer SSH-Verbindungstyp angezeigt.

```
-c https://raw.githubusercontent.com/apache/airflow/constraints-Airflow-version/constraints-Python-version.txt
apache-airflow-providers-ssh
```

### Note

`-c` definiert die URL der Einschränkungen in `requirements.txt`. Dadurch wird sichergestellt, dass Amazon MWAA die richtige Paketversion für Ihre Umgebung installiert.

## Kopieren Sie Ihren geheimen Schlüssel zu Amazon S3

Verwenden Sie den folgenden AWS Command Line Interface Befehl, um Ihren `.pem` Schlüssel in das `dags` Verzeichnis Ihrer Umgebung in Amazon S3 zu kopieren.

```
$ aws s3 cp your-secret-key.pem s3://your-bucket/dags/
```


Amazon MWAA kopiert den Inhalt einschließlich des `.pem` Schlüssels in `dags` das lokale `/usr/local/airflow/dags/` Verzeichnis. Auf diese Weise kann Apache Airflow auf den Schlüssel zugreifen.

## Erstellen Sie eine neue Apache Airflow-Verbindung

Um eine neue SSH-Verbindung mit der Apache Airflow-Benutzeroberfläche zu erstellen

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.

2. Wählen Sie aus der Liste der Umgebungen Open Airflow UI für Ihre Umgebung aus.
3. Wählen Sie auf der Apache Airflow-UI-Seite in der oberen Navigationsleiste Admin aus, um die Dropdownliste zu erweitern, und wählen Sie dann Verbindungen aus.
4. Wählen Sie auf der Seite Verbindungen auflisten die Option + oder die Schaltfläche Neuen Datensatz hinzufügen, um eine neue Verbindung hinzuzufügen.
5. Fügen Sie auf der Seite „Verbindung hinzufügen“ die folgenden Informationen hinzu:
  - a. Geben Sie als Verbindungs-ID ein **ssh\_new**.
  - b. Wählen Sie als Verbindungstyp SSH aus der Dropdownliste aus.

 Note

Wenn der SSH-Verbindungstyp nicht in der Liste verfügbar ist, hat Amazon MWAA das erforderliche Paket nicht installiert. `apache-airflow-providers-ssh`  
Aktualisieren Sie Ihre `requirements.txt` Datei, sodass sie dieses Paket enthält, und versuchen Sie es erneut.

- c. Geben Sie für Host die IP-Adresse der Amazon EC2 EC2-Instance ein, zu der Sie eine Verbindung herstellen möchten. z. B. **12.345.67.89**.
- d. Geben Sie unter Nutzernamen ein, **ec2-user** ob Sie eine Verbindung zu einer Amazon EC2 EC2-Instance herstellen. Ihr Benutzername kann je nach Art der Remote-Instance, zu der Apache Airflow eine Verbindung herstellen soll, unterschiedlich sein.
- e. Geben Sie für Extra das folgende Schlüssel-Wert-Paar im JSON-Format ein:

```
{ "key_file": "/usr/local/airflow/dags/your-secret-key.pem" }
```

Dieses Schlüssel-Wert-Paar weist Apache Airflow an, im lokalen Verzeichnis nach dem geheimen Schlüssel zu suchen. `/dags`

## Codebeispiel

Die folgende DAG verwendet die `SSHOperator`, um eine Verbindung zu Ihrer Amazon EC2 EC2-Zielinstanz herzustellen, und führt dann den `hostname` Linux-Befehl aus, um den Namen der Instanz auszudrucken. Sie können die DAG so ändern, dass sie jeden Befehl oder jedes Skript auf der Remote-Instance ausführt.

1. Öffnen Sie ein Terminal und navigieren Sie zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist. Beispielsweise:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `ssh.py`.

```
from airflow.decorators import dag
from datetime import datetime
from airflow.providers.ssh.operators.ssh import SSHOperator

@dag(
 dag_id="ssh_operator_example",
 schedule_interval=None,
 start_date=datetime(2022, 1, 1),
 catchup=False,
)
def ssh_dag():
 task_1=SSHOperator(
 task_id="ssh_task",
 ssh_conn_id='ssh_new',
 command='hostname',
)

my_ssh_dag = ssh_dag()
```

3. Führen Sie den folgenden AWS CLI Befehl aus, um die DAG in den Bucket Ihrer Umgebung zu kopieren, und lösen Sie die DAG dann mithilfe der Apache Airflow-Benutzeroberfläche aus.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. Bei Erfolg erhalten Sie in den Aufgabenprotokollen für die `ssh_operator_example` DAG eine Ausgabe, die `ssh_task` der folgenden ähnelt:

```
[2022-01-01, 12:00:00 UTC] {{base.py:79}} INFO - Using connection to: id: ssh_new.
Host: 12.345.67.89, Port: None,
Schema: , Login: ec2-user, Password: None, extra: {'key_file': '/usr/local/airflow/
dags/your-secret-key.pem'}
[2022-01-01, 12:00:00 UTC] {{ssh.py:264}} WARNING - Remote Identification Change is
not verified. This won't protect against Man-In-The-Middle attacks
[2022-01-01, 12:00:00 UTC] {{ssh.py:270}} WARNING - No Host Key Verification. This
won't protect against Man-In-The-Middle attacks
```

```
[2022-01-01, 12:00:00 UTC] {{transport.py:1819}} INFO - Connected (version 2.0,
client OpenSSH_7.4)
[2022-01-01, 12:00:00 UTC] {{transport.py:1819}} INFO - Authentication (publickey)
successful!
[2022-01-01, 12:00:00 UTC] {{ssh.py:139}} INFO - Running command: hostname
[2022-01-01, 12:00:00 UTC]{{ssh.py:171}} INFO - ip-123-45-67-89.us-
west-2.compute.internal
[2022-01-01, 12:00:00 UTC] {{taskinstance.py:1280}} INFO - Marking task as SUCCESS.
dag_id=ssh_operator_example, task_id=ssh_task, execution_date=20220712T200914,
start_date=20220712T200915, end_date=20220712T200916
```

## Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow Snowflake-Verbindung

Die folgenden Beispielaufträge AWS Secrets Manager um einen geheimen Schlüssel für eine Apache Airflow Snowflake-Verbindung auf Amazon Managed Workflows for Apache Airflow zu erhalten. Es wird davon ausgegangen, dass Sie die Schritte in abgeschlossen haben [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).

### Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Voraussetzungen](#)
- [Codebeispiel](#)
- [Als nächstes](#)

### Version

- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

### Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Das Secrets Manager-Backend als Apache Airflow-Konfigurationsoption, wie gezeigt in [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).
- Eine Apache Airflow-Verbindungszeichenfolge in Secrets Manager, wie gezeigt in [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).

## Berechtigungen

- Secrets Manager-Berechtigungen wie gezeigt in [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).

## Voraussetzungen

Um den Beispielcode auf dieser Seite zu verwenden, fügen Sie die folgenden Abhängigkeiten zu Ihrem `requirements.txt`. Weitere Informationen hierzu finden Sie unter [Python-Abhängigkeiten installieren](#).

```
apache-airflow-providers-snowflake==1.3.0
```

## Codebeispiel

In den folgenden Schritten wird beschrieben, wie Sie den DAG-Code erstellen, der Secrets Manager aufruft, um das Geheimnis abzurufen.

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist. Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `snowflake_connection.py`.

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

Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
```

the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so.

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

"""

```
from airflow import DAG
from airflow.providers.snowflake.operators.snowflake import SnowflakeOperator
from airflow.utils.dates import days_ago

snowflake_query = [
 """use warehouse "MY_WAREHOUSE";""",
 """select * from "SNOWFLAKE_SAMPLE_DATA"."WEATHER"."WEATHER_14_TOTAL" limit
100;""",
]

with DAG(dag_id='snowflake_test', schedule_interval=None, catchup=False,
start_date=days_ago(1)) as dag:
 snowflake_select = SnowflakeOperator(
 task_id="snowflake_select",
 sql=snowflake_query,
 snowflake_conn_id="snowflake_conn",
)
```

## Als nächstes

- Erfahren Sie, wie Sie den DAG-Code in diesem Beispiel auf `diedags` Ordner in Ihrem Amazon S3-Bucket in [Hinzufügen oder Aktualisieren von DAGs](#).

## Verwenden einer DAG zum Schreiben benutzerdefinierter Metriken CloudWatch

Sie können das folgende Codebeispiel verwenden, um einen gerichteten azyklischen Graphen (DAG) zu schreiben, der eine `PythonOperator` Metriken auf Betriebssystemebene für eine Amazon

MWAA-Umgebung abzurufen. Die DAG veröffentlicht die Daten dann als benutzerdefinierte Metriken auf Amazon.CloudWatch.

Benutzerdefinierte Metriken auf Betriebssystemebene bieten Ihnen zusätzliche Einblicke darüber, wie Ihre Umgebungsmitarbeiter Ressourcen wie virtuellen Speicher und CPU nutzen. Sie können diese Informationen verwenden, um die auszuwählen [Umgebungsklass](#) das passt am besten zu Ihrer Arbeitsbelastung.

## Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Abhängigkeiten](#)
- [Codebeispiel](#)

## Version

- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

## Voraussetzungen

Um das Codebeispiel auf dieser Seite zu verwenden, benötigen Sie Folgendes:

- Ein [Amazon MWAA-Umgebung](#).

## Berechtigungen

- Für die Verwendung des Codebeispiels auf dieser Seite sind keine zusätzlichen Berechtigungen erforderlich.

## Abhängigkeiten

- Für die Verwendung des Codebeispiels auf dieser Seite sind keine zusätzlichen Abhängigkeiten erforderlich.

## Codebeispiel

1. Navigieren Sie in der Befehlszeile zu dem Ordner, in dem Ihr DAG-Code gespeichert ist.  
Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `dags/custom-metrics.py`. Ersetzen `MWAA-ENV-NAME` mit Ihrem Umgebungsnamen.

```
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from airflow.utils.dates import days_ago
from datetime import datetime
import os, json, boto3, psutil, socket

def publish_metric(client, name, value, cat, unit='None'):
 environment_name = os.getenv("MWAA_ENV_NAME")
 value_number = float(value)
 hostname = socket.gethostname()
 ip_address = socket.gethostbyname(hostname)
 print('writing value', value_number, 'to metric', name)
 response = client.put_metric_data(
 Namespace='MWAA-Custom',
 MetricData=[
 {
 'MetricName': name,
 'Dimensions': [
 {
 'Name': 'Environment',
 'Value': environment_name
 },
 {
 'Name': 'Category',
 'Value': cat
 },
 {
 'Name': 'Host',
 'Value': ip_address
 },
],
 'Timestamp': datetime.now(),
```



```
 'Value': value_number,
 'Unit': unit
 },
]
)
print(response)
return response

def python_fn(**kwargs):
 client = boto3.client('cloudwatch')

 cpu_stats = psutil.cpu_stats()
 print('cpu_stats', cpu_stats)

 virtual = psutil.virtual_memory()
 cpu_times_percent = psutil.cpu_times_percent(interval=0)

 publish_metric(client=client, name='virtual_memory_total',
cat='virtual_memory', value=virtual.total, unit='Bytes')
 publish_metric(client=client, name='virtual_memory_available',
cat='virtual_memory', value=virtual.available, unit='Bytes')
 publish_metric(client=client, name='virtual_memory_used', cat='virtual_memory',
value=virtual.used, unit='Bytes')
 publish_metric(client=client, name='virtual_memory_free', cat='virtual_memory',
value=virtual.free, unit='Bytes')
 publish_metric(client=client, name='virtual_memory_active',
cat='virtual_memory', value=virtual.active, unit='Bytes')
 publish_metric(client=client, name='virtual_memory_inactive',
cat='virtual_memory', value=virtual.inactive, unit='Bytes')
 publish_metric(client=client, name='virtual_memory_percent',
cat='virtual_memory', value=virtual.percent, unit='Percent')

 publish_metric(client=client, name='cpu_times_percent_user',
cat='cpu_times_percent', value=cpu_times_percent.user, unit='Percent')
 publish_metric(client=client, name='cpu_times_percent_system',
cat='cpu_times_percent', value=cpu_times_percent.system, unit='Percent')
 publish_metric(client=client, name='cpu_times_percent_idle',
cat='cpu_times_percent', value=cpu_times_percent.idle, unit='Percent')

 return "OK"

with DAG(dag_id=os.path.basename(__file__).replace(".py", ""),
schedule_interval='*/5 * * * *', catchup=False, start_date=days_ago(1)) as dag:
```

```
t = PythonOperator(task_id="memory_test", python_callable=python_fn,
provide_context=True)
```

- Führen Sie Folgendes aus AWS CLIBefehl, um die DAG in den Bucket Ihrer Umgebung zu kopieren und dann die DAG mithilfe der Apache Airflow-Benutzeroberfläche auszulösen.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

- Wenn die DAG erfolgreich ausgeführt wird, sollten Sie in Ihren Apache Airflow-Protokollen etwas Ähnliches wie das Folgende sehen:

```
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO -
cpu_stats scpustats(ctx_switches=3253992384, interrupts=1964237163,
soft_interrupts=492328209, syscalls=0)
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - writing value
16024199168.0 to metric virtual_memory_total
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - {'ResponseMetadata':
{'RequestId': 'fad289ac-aa51-46a9-8b18-24e4e4063f4d', 'HTTPStatusCode': 200,
'HTTPHeaders': {'x-amzn-requestid': 'fad289ac-aa51-46a9-8b18-24e4e4063f4d',
'content-type': 'text/xml', 'content-length': '212', 'date': 'Tue, 16 Aug 2022
17:54:45 GMT'}, 'RetryAttempts': 0}}
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - writing value
14356287488.0 to metric virtual_memory_available
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - {'ResponseMetadata':
{'RequestId': '6ef60085-07ab-4865-8abf-dc94f90cab46', 'HTTPStatusCode': 200,
'HTTPHeaders': {'x-amzn-requestid': '6ef60085-07ab-4865-8abf-dc94f90cab46',
'content-type': 'text/xml', 'content-length': '212', 'date': 'Tue, 16 Aug 2022
17:54:45 GMT'}, 'RetryAttempts': 0}}
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - writing value
1342296064.0 to metric virtual_memory_used
[2022-08-16, 10:54:46 UTC] {{logging_mixin.py:109}} INFO - {'ResponseMetadata':
{'RequestId': 'd5331438-5d3c-4df2-bc42-52dcf8d60a00', 'HTTPStatusCode': 200,
'HTTPHeaders': {'x-amzn-requestid': 'd5331438-5d3c-4df2-bc42-52dcf8d60a00',
'content-type': 'text/xml', 'content-length': '212', 'date': 'Tue, 16 Aug 2022
17:54:45 GMT'}, 'RetryAttempts': 0}}
...
[2022-08-16, 10:54:46 UTC] {{python.py:152}} INFO - Done. Returned value was: OK
[2022-08-16, 10:54:46 UTC] {{taskinstance.py:1280}} INFO - Marking task as SUCCESS.
dag_id=dag-custom-metrics, task_id=memory_test, execution_date=20220816T175444,
start_date=20220816T175445, end_date=20220816T175446
[2022-08-16, 10:54:46 UTC] {{local_task_job.py:154}} INFO - Task exited with return
code 0
```

# Aurora-PostgreSQL-Datenbankbereinigung in einer Amazon-MWAA-Umgebung

Amazon Managed Workflows for Apache Airflow verwendet eine Aurora-PostgreSQL-Datenbank als Apache-Airflow-Metadatendatenbank, in der DAG ausgeführt wird und Aufgaben-Instances gespeichert werden. Der folgende Beispielcode löscht regelmäßig Einträge aus der dedizierten Aurora-PostgreSQL-Datenbank für Ihre Amazon-MWAA-Umgebung.

Themen

- [Version](#)
- [Voraussetzungen](#)
- [Abhängigkeiten](#)
- [Codebeispiel](#)

## Version

- Sie können das Codebeispiel auf dieser Seite mit Apache Airflow v2 und höher in [Python 3.10](#) verwenden.

## Voraussetzungen

Um den Beispielcode auf dieser Seite zu verwenden, benötigen Sie Folgendes:

- Eine [Amazon MWAA-Umgebung](#) .

## Abhängigkeiten

- Um dieses Codebeispiel mit Apache Airflow v2 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v2-Basisinstallation](#) in Ihrer Umgebung.

## Codebeispiel

Die folgende DAG bereinigt die Metadatendatenbank für die in angegebenen Tabellen `TABLES_TO_CLEAN`. Das Beispiel löscht Daten aus den angegebenen Tabellen der letzten

sieben Tage. Um anzupassen, wie weit die Einträge gelöscht werden, legen Sie `MAX_AGE_IN_DAYS` auf einen anderen Wert fest.

## Apache Airflow v2

```
from airflow import settings
from airflow.utils.dates import days_ago
from airflow.models import DagTag, DagModel, DagRun, ImportError, Log, SlaMiss,
 RenderedTaskInstanceFields, TaskInstance, TaskReschedule, XCom
from airflow.decorators import dag, task
from airflow.utils.dates import days_ago
from time import sleep

from airflow.version import version
major_version, minor_version = int(version.split('.')[0]), int(version.split('.')[1])
if major_version >= 2 and minor_version >= 6:
 from airflow.jobs.job import Job
else:
 # The BaseJob class was renamed as of Apache Airflow v2.6
 from airflow.jobs.base_job import BaseJob as Job

Delete entries for the past seven days. Adjust MAX_AGE_IN_DAYS to set how far back
this DAG cleans the database.
MAX_AGE_IN_DAYS = 7
MIN_AGE_IN_DAYS = 0
DECREMENT = -7

This is a list of (table, time) tuples.
table = the table to clean in the metadata database
time = the column in the table associated to the timestamp of an entry
or None if not applicable.
TABLES_TO_CLEAN = [[Job, Job.latest_heartbeat],
 [TaskInstance, TaskInstance.execution_date],
 [TaskReschedule, TaskReschedule.execution_date],
 [DagTag, None],
 [DagModel, DagModel.last_parsed_time],
 [DagRun, DagRun.execution_date],
 [ImportError, ImportError.timestamp],
 [Log, Log.dttm],
 [SlaMiss, SlaMiss.execution_date],
 [RenderedTaskInstanceFields, RenderedTaskInstanceFields.execution_date],
 [XCom, XCom.execution_date],
```

```
]

@task()
def cleanup_db_fn(x):
 session = settings.Session()

 if x[1]:
 for oldest_days_ago in range(MAX_AGE_IN_DAYS, MIN_AGE_IN_DAYS, DECREMENT):
 earliest_days_ago = max(oldest_days_ago + DECREMENT, MIN_AGE_IN_DAYS)
 print(f"deleting {str(x[0])} entries between {earliest_days_ago} and
{oldest_days_ago} days old...")
 earliest_date = days_ago(earliest_days_ago)
 oldest_date = days_ago(oldest_days_ago)
 query = session.query(x[0]).filter(x[1] >= oldest_date).filter(x[1] <=
earliest_date)
 query.delete(synchronize_session= False)
 session.commit()
 sleep(5)
 else:
 # No time column specified for the table. Delete all entries
 print("deleting", str(x[0]), "...")
 query = session.query(x[0])
 query.delete(synchronize_session= False)
 session.commit()

 session.close()

@dag(
 dag_id="cleanup_db",
 schedule_interval="@weekly",
 start_date=days_ago(7),
 catchup=False,
 is_paused_upon_creation=False
)

def clean_db_dag_fn():
 t_last=None
 for x in TABLES_TO_CLEAN:
 t=cleanup_db_fn(x)
 if t_last:
 t_last >> t
 t_last = t
```

```
clean_db_dag = clean_db_dag_fn()
```

## Exportieren von Umgebungsmetadaten in CSV-Dateien auf Amazon S3

Das folgende Codebeispiel zeigt, wie Sie einen gerichteten azyklischen Graphen (DAG) erstellen können, der die Datenbank nach einer Reihe von DAG-Ausführungsinformationen abfragt und die Daten in .csv Dateien schreibt, die auf Amazon S3 gespeichert sind.

Möglicherweise möchten Sie Informationen aus der Aurora PostgreSQL-Datenbank Ihrer Umgebung exportieren, um die Daten lokal zu untersuchen, sie im Objektspeicher zu archivieren oder sie mit Tools wie dem [Amazon S3 S3-to-Amazon-Redshift-Operator und der Datenbankbereinigung zu kombinieren](#), um Amazon MWAA-Metadaten aus der Umgebung zu entfernen, sie aber für future Analysen aufzubewahren.

[Sie können die Datenbank nach allen Objekten abfragen, die in Apache Airflow-Modellen aufgeführt sind.](#) In diesem Codebeispiel werden drei Modelle, `undDagRun`, `verwendetTaskFail`, die Informationen bereitstellen `TaskInstance`, die für DAG-Läufe relevant sind.

### Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Voraussetzungen](#)
- [Codebeispiel](#)

### Version

- Sie können das Codebeispiel auf dieser Seite mit Apache Airflow v2 und höher in [Python 3.10](#) verwenden.

### Voraussetzungen

Um den Beispielcode auf dieser Seite zu verwenden, benötigen Sie Folgendes:

- Eine [Amazon MWAA-Umgebung](#).
- Ein [neuer Amazon S3 S3-Bucket](#), in den Sie Ihre Metadateninformationen exportieren möchten.

## Berechtigungen

Amazon MWAA benötigt für die Amazon S3-Aktion die Erlaubnis, die abgefragten Metadateninformationen in Amazon S3 `s3:PutObject` zu schreiben. Fügen Sie der Ausführungsrolle Ihrer Umgebung die folgende Richtlinienerklärung hinzu.

```
{
 "Effect": "Allow",
 "Action": "s3:PutObject*",
 "Resource": "arn:aws:s3:::your-new-export-bucket"
}
```

Diese Richtlinie beschränkt den Schreibzugriff nur auf *your-new-export-bucket*.

## Voraussetzungen

- Um dieses Codebeispiel mit Apache Airflow v2 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v2-Basisinstallation](#) in Ihrer Umgebung.

## Codebeispiel

In den folgenden Schritten wird beschrieben, wie Sie eine DAG erstellen können, die Aurora PostgreSQL abfragt und das Ergebnis in Ihren neuen Amazon S3 S3-Bucket schreibt.

1. Navigieren Sie in Ihrem Terminal zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist.  
Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `metadata_to_csv.py`. Sie können den zugewiesenen Wert ändern, `MAX_AGE_IN_DAYS` um das Alter der ältesten Datensätze zu steuern, die Ihre DAG aus der Metadatenbank abfragt.

```
from airflow.decorators import dag, task
from airflow import settings
import os
import boto3
from airflow.utils.dates import days_ago
from airflow.models import DagRun, TaskFail, TaskInstance
import csv, re
from io import StringIO

DAG_ID = os.path.basename(__file__).replace(".py", "")

MAX_AGE_IN_DAYS = 30
S3_BUCKET = '<your-export-bucket>'
S3_KEY = 'files/export/{0}.csv'

You can add other objects to export from the metadatabase,
OBJECTS_TO_EXPORT = [
 [DagRun, DagRun.execution_date],
 [TaskFail, TaskFail.execution_date],
 [TaskInstance, TaskInstance.execution_date],
]

@task()
def export_db_task(**kwargs):
 session = settings.Session()
 print("session: ", str(session))

 oldest_date = days_ago(MAX_AGE_IN_DAYS)
 print("oldest_date: ", oldest_date)

 s3 = boto3.client('s3')

 for x in OBJECTS_TO_EXPORT:
 query = session.query(x[0]).filter(x[1] >= days_ago(MAX_AGE_IN_DAYS))
 print("type", type(query))
 allrows=query.all()
 name=re.sub("[<>]", "", str(x[0]))
 print(name,": ", str(allrows))

 if len(allrows) > 0:
 outfileStr=""
 f = StringIO(outfileStr)
 w = csv.DictWriter(f, vars(allrows[0]).keys())
```



```

 w.writeheader()
 for y in allrows:
 w.writerow(vars(y))
 outkey = S3_KEY.format(name[6:])
 s3.put_object(Bucket=S3_BUCKET, Key=outkey, Body=f.getvalue())

@dag(
 dag_id=DAG_ID,
 schedule_interval=None,
 start_date=days_ago(1),
)
def export_db():
 t = export_db_task()

metadb_to_s3_test = export_db()

```

3. Führen Sie den folgenden AWS CLI Befehl aus, um die DAG in den Bucket Ihrer Umgebung zu kopieren, und lösen Sie die DAG dann mithilfe der Apache Airflow-Benutzeroberfläche aus.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. Wenn der Vorgang erfolgreich ist, geben Sie in den Aufgabenprotokollen für die Aufgabe eine Ausgabe ähnlich der `export_db` folgenden aus:

```

[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - type <class
 'sqlalchemy.orm.query.Query'>
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - class
airflow.models.dagrun.DagRun : [your-tasks]
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - type <class
 'sqlalchemy.orm.query.Query'>
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - class
airflow.models.taskfail.TaskFail : [your-tasks]
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - type <class
 'sqlalchemy.orm.query.Query'>
[2022-01-01, 12:00:00 PDT] {{logging_mixin.py:109}} INFO - class
airflow.models.taskinstance.TaskInstance : [your-tasks]
[2022-01-01, 12:00:00 PDT] {{python.py:152}} INFO - Done. Returned value was: OK
[2022-01-01, 12:00:00 PDT] {{taskinstance.py:1280}} INFO - Marking task as
 SUCCESS. dag_id=metadb_to_s3, task_id=export_db, execution_date=20220101T000000,
 start_date=20220101T000000, end_date=20220101T000000
[2022-01-01, 12:00:00 PDT] {{local_task_job.py:154}} INFO - Task exited with return
 code 0

```

```
[2022-01-01, 12:00:00 PDT] {{local_task_job.py:264}} INFO - 0 downstream tasks
scheduled from follow-on schedule check
```

Sie können jetzt auf die exportierten .csv Dateien in Ihrem neuen Amazon S3 S3-Bucket in zugreifen und sie herunterladen/`files/export/`.

## Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Variable

Die folgenden Beispielaufrufe AWS Secrets Manager um einen geheimen Schlüssel für eine Apache Airflow-Variable in Amazon Managed Workflows for Apache Airflow abzurufen. Es wird davon ausgegangen, dass Sie die Schritte in abgeschlossen haben [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).

Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Voraussetzungen](#)
- [Codebeispiel](#)
- [Als nächstes](#)

### Version

- Der Beispielcode auf dieser Seite kann verwendet werden mit Apache Airflow v1 in [Python 3.7](#).
- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

### Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Das Secrets Manager-Backend als Apache Airflow-Konfigurationsoption, wie gezeigt in [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).
- Eine Apache Airflow-Variablenzeichenfolge in Secrets Manager, wie gezeigt in [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).

## Berechtigungen

- Secrets Manager-Berechtigungen wie gezeigt in [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).

## Voraussetzungen

- Um dieses Codebeispiel mit Apache Airflow v1 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v1-Basisinstallation](#) auf deine Umgebung.
- Um dieses Codebeispiel mit Apache Airflow v2 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v2-Basisinstallation](#) auf deine Umgebung.

## Codebeispiel

In den folgenden Schritten wird beschrieben, wie Sie den DAG-Code erstellen, der Secrets Manager aufruft, um das Geheimnis abzurufen.

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist.  
Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `secrets-manager-var.py`.

```
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
```

```
from airflow.models import Variable
from airflow.utils.dates import days_ago
from datetime import timedelta
import os
DAG_ID = os.path.basename(__file__).replace(".py", "")
DEFAULT_ARGS = {
 'owner': 'airflow',
 'depends_on_past': False,
 'email': ['airflow@example.com'],
 'email_on_failure': False,
 'email_on_retry': False,
}
def get_variable_fn(**kwargs):
 my_variable_name = Variable.get("test-variable", default_var="undefined")
 print("my_variable_name: ", my_variable_name)
 return my_variable_name
with DAG(
 dag_id=DAG_ID,
 default_args=DEFAULT_ARGS,
 dagrun_timeout=timedelta(hours=2),
 start_date=days_ago(1),
 schedule_interval='@once',
 tags=['variable']
) as dag:
 get_variable = PythonOperator(
 task_id="get_variable",
 python_callable=get_variable_fn,
 provide_context=True
)
```

## Als nächstes

- Erfahren Sie, wie Sie den DAG-Code in diesem Beispiel auf `diedags` Ordner in Ihrem Amazon S3-Bucket in [Hinzufügen oder Aktualisieren von DAGs](#).

## Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Verbindung

Die folgenden Beispielaufrufe AWS Secrets Manager um einen geheimen Schlüssel für eine Apache Airflow-Verbindung auf Amazon Managed Workflows for Apache Airflow zu erhalten. Es wird davon

ausgegangen, dass Sie die Schritte in abgeschlossen haben [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).

## Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Voraussetzungen](#)
- [Codebeispiel](#)
- [Als nächstes](#)

## Version

- Der Beispielcode auf dieser Seite kann verwendet werden mit Apache Airflow v1 in [Python 3.7](#).
- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

## Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Das Secrets Manager-Backend als Apache Airflow-Konfigurationsoption, wie gezeigt in [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).
- Eine Apache Airflow-Verbindungszeichenfolge in Secrets Manager, wie gezeigt in [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).

## Berechtigungen

- Secrets Manager-Berechtigungen wie gezeigt in [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#).

## Voraussetzungen

- Um dieses Codebeispiel mit Apache Airflow v1 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v1-Basisinstallation](#) auf deine Umgebung.
- Um dieses Codebeispiel mit Apache Airflow v2 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v2-Basisinstallation](#) auf deine Umgebung.

## Codebeispiel

In den folgenden Schritten wird beschrieben, wie Sie den DAG-Code erstellen, der Secrets Manager aufruft, um das Geheimnis abzurufen.

### Apache Airflow v2

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist. Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `untersecrets-manager.py`.

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

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

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

```
"""
from airflow import DAG, settings, secrets
from airflow.operators.python import PythonOperator
from airflow.utils.dates import days_ago
from airflow.providers.amazon.aws.hooks.base_aws import AwsBaseHook

from datetime import timedelta
import os

The steps to create this secret key can be found at: https://
docs.aws.amazon.com/mwaa/latest/userguide/connections-secrets-manager.html
sm_secretId_name = 'airflow/connections/myconn'

default_args = {
 'owner': 'airflow',
 'start_date': days_ago(1),
 'depends_on_past': False
}

Gets the secret myconn from Secrets Manager
def read_from_aws_sm_fn(**kwargs):
 ### set up Secrets Manager
 hook = AwsBaseHook(client_type='secretsmanager')
 client = hook.get_client_type('secretsmanager')
 response = client.get_secret_value(SecretId=sm_secretId_name)
 myConnSecretString = response["SecretString"]

 return myConnSecretString

'os.path.basename(__file__).replace(".py", "")' uses the file name secrets-
manager.py for a DAG ID of secrets-manager
with DAG(
 dag_id=os.path.basename(__file__).replace(".py", ""),
 default_args=default_args,
 dagrun_timeout=timedelta(hours=2),
 start_date=days_ago(1),
 schedule_interval=None
) as dag:
 write_all_to_aws_sm = PythonOperator(
 task_id="read_from_aws_sm",
 python_callable=read_from_aws_sm_fn,
 provide_context=True
```

```
)
```

## Apache Airflow v1

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist.  
Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal `untersecrets-manager.py`.

```
from airflow import DAG, settings, secrets
from airflow.operators.python_operator import PythonOperator
from airflow.utils.dates import days_ago
from airflow.contrib.hooks.aws_hook import AwsHook

from datetime import timedelta
import os

The steps to create this secret key can be found at: https://
docs.aws.amazon.com/mwaa/latest/userguide/connections-secrets-manager.html
sm_secretId_name = 'airflow/connections/myconn'

default_args = {
 'owner': 'airflow',
 'start_date': days_ago(1),
 'depends_on_past': False
}

Gets the secret myconn from Secrets Manager
def read_from_aws_sm_fn(**kwargs):
 ### set up Secrets Manager
 hook = AwsHook()
 client = hook.get_client_type('secretsmanager')
 response = client.get_secret_value(SecretId=sm_secretId_name)
 myConnSecretString = response["SecretString"]

 return myConnSecretString
```



```
'os.path.basename(__file__).replace(".py", "")' uses the file name secrets-
manager.py for a DAG ID of secrets-manager
with DAG(
 dag_id=os.path.basename(__file__).replace(".py", ""),
 default_args=default_args,
 dagrun_timeout=timedelta(hours=2),
 start_date=days_ago(1),
 schedule_interval=None
) as dag:
 write_all_to_aws_sm = PythonOperator(
 task_id="read_from_aws_sm",
 python_callable=read_from_aws_sm_fn,
 provide_context=True
)
```

## Als nächstes

- Erfahren Sie, wie Sie den DAG-Code in diesem Beispiel auf `diedags` Ordner in Ihrem Amazon S3-Bucket in [Hinzufügen oder Aktualisieren von DAGs](#).

## Ein benutzerdefiniertes Plugin mit Oracle erstellen

Das folgende Beispiel führt Sie durch die Schritte zum Erstellen eines benutzerdefinierten Plug-ins mit Oracle für Amazon MWAA. Es kann mit anderen benutzerdefinierten Plug-ins und Binärdateien in Ihrer Datei `plugins.zip` kombiniert werden.

### Inhalt

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Voraussetzungen](#)
- [Codebeispiel](#)
- [Erstellen Sie das benutzerdefinierte Plugin](#)
  - [Abhängigkeiten herunterladen](#)
  - [Benutzerdefiniertes Plugin](#)
  - [Plugins.zip](#)

- [Airflow-Konfigurationsoptionen](#)
- [Als nächstes](#)

## Version

- Der Beispielcode auf dieser Seite kann verwendet werden mit Apache Airflow v1 in [Python 3.7](#).
- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

## Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Ein [Amazon MWAA-Umgebung](#).
- Arbeiterprotokollierung auf jeder Protokollebene aktiviert, CRITICAL oder höher, für Ihre Umgebung. Weitere Informationen zu Amazon MWAA-Protokolltypen und zur Verwaltung Ihrer Protokollgruppen finden Sie unter [the section called “Airflow-Protokolle anzeigen”](#)

## Berechtigungen

- Für die Verwendung des Codebeispiels auf dieser Seite sind keine zusätzlichen Berechtigungen erforderlich.

## Voraussetzungen

Um den Beispielcode auf dieser Seite zu verwenden, fügen Sie die folgenden Abhängigkeiten zu Ihrem `requirements.txt`. Weitere Informationen hierzu finden Sie unter [Python-Abhängigkeiten installieren](#).

### Apache Airflow v2

```
-c https://raw.githubusercontent.com/apache/airflow/constraints-2.0.2/
constraints-3.7.txt
cx_Oracle
apache-airflow-providers-oracle
```

## Apache Airflow v1

```
cx_Oracle==8.1.0
apache-airflow[oracle]==1.10.12
```

## Codebeispiel

In den folgenden Schritten wird beschrieben, wie Sie den DAG-Code erstellen, mit dem das benutzerdefinierte Plugin getestet wird.

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist. Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `oracle.py`.

```
from airflow import DAG
from airflow.operators.python_operator import PythonOperator
from airflow.utils.dates import days_ago
import os
import cx_Oracle

DAG_ID = os.path.basename(__file__).replace(".py", "")

def testHook(**kwargs):
 cx_Oracle.init_oracle_client()
 version = cx_Oracle.clientversion()
 print("cx_Oracle.clientversion",version)
 return version

with DAG(dag_id=DAG_ID, schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
 hook_test = PythonOperator(
 task_id="hook_test",
 python_callable=testHook,
 provide_context=True
)
```

## Erstellen Sie das benutzerdefinierte Plugin

In diesem Abschnitt wird beschrieben, wie Sie die Abhängigkeiten herunterladen, das benutzerdefinierte Plugin und die Datei `plugins.zip` erstellen.

### Abhängigkeiten herunterladen

Amazon MWAA extrahiert den Inhalt von `plugins.zip` in `/usr/local/airflow/plugins` auf jedem Amazon MWAA-Scheduler und Worker-Container. Dies wird verwendet, um Binärdateien zu Ihrer Umgebung hinzuzufügen. Die folgenden Schritte beschreiben, wie Sie die für das benutzerdefinierte Plugin benötigten Dateien zusammenstellen.

Rufen Sie das Amazon Linux-Container-Image ab

1. Rufen Sie in Ihrer Befehlszeile das Amazon Linux-Container-Image ab und führen Sie den Container lokal aus. Beispiele:

```
docker pull amazonlinux
docker run -it amazonlinux:latest /bin/bash
```

Ihre Befehlszeile sollte eine Bash-Befehlszeile aufrufen. Beispiele:

```
bash-4.2#
```

2. Installieren Sie die Linux-native asynchrone I/O-Funktion (`libaio`).

```
yum -y install libaio
```

3. Lassen Sie dieses Fenster für weitere Schritte offen. Wir werden die folgenden Dateien lokal kopieren: `lib64/libaio.so.1`, `lib64/libaio.so.1.0.0`, `lib64/libaio.so.1.0.1`.

### Kundenordner herunterladen

1. Installieren Sie das Unzip-Paket lokal. Beispiele:

```
sudo yum install unzip
```

2. Erstellen Sie ein `oracle_plugin`-Verzeichnis. Beispiele:

```
mkdir oracle_plugin
```

```
cd oracle_plugin
```

3. Verwenden Sie den folgenden curl-Befehl, um das herunterzuladen [instantclient-basic-linux.x64-18.5.0.0dbru.zip](#) von [Oracle Instant Client-Downloads für Linux x86-64 \(64-Bit\)](#).

```
curl https://download.oracle.com/otn_software/linux/instantclient/185000/instantclient-basic-linux.x64-18.5.0.0dbru.zip > client.zip
```

4. Entpacken Sie die Datei `client.zip`. Beispiele:

```
unzip *.zip
```

## Extrahieren Sie Dateien aus Docker

1. Zeigen Sie in einer neuen Befehlszeile Ihre Docker-Container-ID an und notieren Sie sie. Beispiele:

```
docker container ls
```

Ihre Befehlszeile sollte alle Container und ihre IDs zurückgeben. Beispiele:

```
debc16fd6970
```

2. In dein `oracle_plugin` Verzeichnis, entpacke das `lib64/libaio.so.1`, `lib64/libaio.so.1.0.0`, `lib64/libaio.so.1.0.1` Dateien in die lokale `instantclient_18_5` Ordner. Beispiele:

```
docker cp debc16fd6970:/lib64/libaio.so.1 instantclient_18_5/
docker cp debc16fd6970:/lib64/libaio.so.1.0.0 instantclient_18_5/
docker cp debc16fd6970:/lib64/libaio.so.1.0.1 instantclient_18_5/
```

## Benutzerdefiniertes Plugin

Apache Airflow führt beim Start den Inhalt von Python-Dateien im Plugins-Ordner aus. Dies wird verwendet, um Umgebungsvariablen festzulegen und zu ändern. In den folgenden Schritten wird der Beispielcode für das benutzerdefinierte Plugin beschrieben.

- Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `env_var_plugin_oracle.py`.

```
from airflow.plugins_manager import AirflowPlugin
import os

os.environ["LD_LIBRARY_PATH"]='/usr/local/airflow/plugins/instantclient_18_5'
os.environ["DPI_DEBUG_LEVEL"]="64"

class EnvVarPlugin(AirflowPlugin):
 name = 'env_var_plugin'
```

## Plugins.zip

Die folgenden Schritte zeigen, wie Sie das erstellen `plugins.zip`. Der Inhalt dieses Beispiels kann mit Ihren anderen Plugins und Binärdateien zu einer einzigen kombiniert werden `plugins.zip` Datei.

Zippen Sie den Inhalt des Plugin-Verzeichnisses

1. Navigieren Sie in der Befehlszeile zum `oracle_plugin` Verzeichnis. Beispiele:

```
cd oracle_plugin
```

2. Zippen Sie die `instantclient_18_5` Verzeichnis in `plugins.zip`. Beispiele:

```
zip -r ../plugins.zip ./
```

3. In Ihrer Befehlszeile sollten Sie Folgendes sehen:

```
oracle_plugin$ ls
client.zip instantclient_18_5
```

4. Entferne die `client.zip` Datei. Beispiele:

```
rm client.zip
```

Zippen Sie die Datei `env_var_plugin_oracle.py`

1. Füge das hinzu `env_var_plugin_oracle.py` Datei im Stammverzeichnis von `plugins.zip`.  
Beispiele:

```
zip plugins.zip env_var_plugin_oracle.py
```

2. Ihre `plugins.zip` sollte jetzt Folgendes enthalten:

```
env_var_plugin_oracle.py
instantclient_18_5/
```

## Airflow-Konfigurationsoptionen

Wenn Sie Apache Airflow v2 verwenden, fügen Sie hinzu `core.lazy_load_plugins : False` als Apache Airflow-Konfigurationsoption. Weitere Informationen finden Sie unter [Verwenden von Konfigurationsoptionen zum Laden von Plugins in 2](#).

### Als nächstes

- Erfahren Sie, wie Sie das hochladen `requirements.txt` Datei in diesem Beispiel in Ihren Amazon S3-Bucket in [Python-Abhängigkeiten installieren](#).
- Erfahren Sie, wie Sie den DAG-Code in diesem Beispiel auf `diedags` Ordner in Ihrem Amazon S3-Bucket in [Hinzufügen oder Aktualisieren von DAGs](#).
- Erfahre mehr darüber, wie du die hochlad `plugins.zip` Datei in diesem Beispiel in Ihren Amazon S3-Bucket in [Installation benutzerdefinierter Plugins](#).

## Erstellen eines benutzerdefinierten Plugins, das Laufzeitumgebungsvariablen generiert

Das folgende Beispiel führt Sie durch die Schritte zur Erstellung eines benutzerdefinierten Plug-ins, das zur Laufzeit Umgebungsvariablen in einer Amazon Managed Workflows for Apache Airflow-Umgebung generiert.

### Themen

- [Version](#)

- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Voraussetzungen](#)
- [Benutzerdefiniertes Plugin](#)
- [Plugins.zip](#)
- [Airflow-Konfigurationsoptionen für den Luftstrom](#)
- [Als nächstes](#)

## Version

- Der Beispielcode auf dieser Seite kann mit Apache Airflow v1 in [Python 3.7](#) verwendet werden.

## Voraussetzungen

Um den Beispiel-Code auf dieser Seite zu verwenden.

- Eine [Amazon MWAA-Umgebung](#).

## Berechtigungen

- Für die Verwendung des Codebeispiels auf dieser Seite sind keine zusätzlichen Berechtigungen erforderlich.

## Voraussetzungen

- Um dieses Codebeispiel mit Apache Airflow v1 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v1-Basisinstallation](#) in Ihrer Umgebung.

## Benutzerdefiniertes Plugin

Apache Airflow führt beim Start den Inhalt der Python-Dateien im Plugins-Ordner aus. Dies wird verwendet, um Umgebungsvariablen festzulegen und zu ändern. Im folgenden wird den Beispiel-Code für das benutzerdefinierte Plug-In in den Standard die.



1. Navigieren Sie in Ihrer Eingabeaufforderung zu dem Verzeichnis. Beispiel:

```
cd plugins
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie es lokal als `alsenv_var_plugin.py` im obigen Ordner.

```
from airflow.plugins_manager import AirflowPlugin
import os

os.environ["PATH"] = os.getenv("PATH") + ":/usr/local/airflow/.local/lib/python3.7/
site-packages"
os.environ["JAVA_HOME"]="/usr/lib/jvm/java-1.8.0-
openjdk-1.8.0.272.b10-1.amzn2.0.1.x86_64"

class EnvVarPlugin(AirflowPlugin):
 name = 'env_var_plugin'
```

## Plugins.zip

Im folgenden wird die folgenden folgenden Schritte den `plugins.zip`. Der Inhalt dieses Beispiels kann mit anderen Plugins und Binärdateien in einer einzigen `plugins.zip` Datei kombiniert werden.

1. Navigieren Sie in der Eingabeaufforderung zu dem `hive_plugin` Verzeichnis aus dem vorherigen Schritt. Beispiel:

```
cd plugins
```

2. Komprimieren Sie den Inhalt Ihres `plugins` Ordners.

```
zip -r ../plugins.zip ./
```

## Airflow-Konfigurationsoptionen für den Luftstrom

Wenn Sie Apache Airflow v2 verwenden, fügen Sie `escore.lazy_load_plugins : False` als Apache Airflow-Konfigurationsoption hinzu. Weitere Informationen finden Sie unter [Verwenden von Konfigurationsoptionen zum Laden von Plugins in 2.](#)

## Als nächstes

- Erfahren Sie, wie Sie die `requirements.txt` Datei in diesem Beispiel in Ihren Amazon S3 S3-Bucket hochladen [Python-Abhängigkeiten installieren](#).
- Erfahren Sie, wie Sie den DAG-Code in diesem Beispiel in den `dags` Ordner in Ihrem Amazon S3 S3-Bucket hochladen [Hinzufügen oder Aktualisieren von DAGs](#).
- Erfahren Sie mehr darüber, wie Sie die `plugins.zip` Datei in diesem Beispiel in Ihren Amazon S3 S3-Bucket hochladen können [Installation benutzerdefinierter Plugins](#).

## Ändern der Zeitzone einer DAG auf Amazon MWAA

Apache Airflow plant Ihren gerichteten azyklischen Graphen (DAG) standardmäßig in UTC+0. Die folgenden Schritte zeigen, wie Sie die Zeitzone ändern können, in der Amazon MWAA Ihre DAGs ausführt. [Pendel](#). Optional zeigt dieses Thema, wie Sie ein benutzerdefiniertes Plugin erstellen können, um die Zeitzone für die Apache Airflow-Logs Ihrer Umgebung zu ändern.

### Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Erstellen Sie ein Plugin, um die Zeitzone in Airflow-Protokollen zu ändern](#)
- [plugins.zip-Ziel erstellen](#)
- [Codebeispiel](#)
- [Als nächstes](#)

## Version

- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

## Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Ein [Amazon MWAA-Umgebung](#).

## Berechtigungen

- Für die Verwendung des Codebeispiels auf dieser Seite sind keine zusätzlichen Berechtigungen erforderlich.

## Erstellen Sie ein Plugin, um die Zeitzone in Airflow-Protokollen zu ändern

Apache Airflow führt die Python-Dateien in der `plugins` Verzeichnis beim Start. Mit dem folgenden Plugin können Sie die Zeitzone des Executors überschreiben, wodurch die Zeitzone geändert wird, in der Apache Airflow Logs schreibt.

1. Erstellen Sie ein Verzeichnis mit dem Namen `plugins` für Ihr benutzerdefiniertes Plugin und navigieren Sie zum Verzeichnis. Beispiele:

```
$ mkdir plugins
$ cd plugins
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `dag-timezone-plugin.py` in der `plugins` Ordner.

```
import time
import os

os.environ['TZ'] = 'America/Los_Angeles'
time.tzset()
```

3. In der `plugins` Verzeichnis, erstelle eine leere Python-Datei mit dem Namen `__init__.py`. Dein `plugins` Das Verzeichnis sollte dem folgenden ähneln:

```
plugins/
|-- __init__.py
|-- dag-timezone-plugin.py
```

## plugins.zip Ziel erstellen

Die folgenden Schritte zeigen, wie Sie erstellen `plugins.zip`. Der Inhalt dieses Beispiels kann mit anderen Plugins und Binärdateien zu einem einzigen kombiniert werden `plugins.zip` Datei.

1. Navigieren Sie in der Befehlszeile zum `plugins` Verzeichnis aus dem vorherigen Schritt.  
Beispiele:

```
cd plugins
```

2. Zippen Sie den Inhalt in Ihrem `plugins` Verzeichnis.

```
zip -r ../plugins.zip ./
```

3. hochladen `plugins.zip` zu Ihrem S3-Bucket

```
$ aws s3 cp plugins.zip s3://your-mwaa-bucket/
```

## Codebeispiel

Um die Standardzeitzone (UTC+0) zu ändern, in der die DAG läuft, verwenden wir eine Bibliothek namens [Pendel](#), eine Python-Bibliothek für die Arbeit mit zeitzonensensitiver Datetime.

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihre DAGs gespeichert sind.  
Beispiele:

```
$ cd dags
```

2. Kopieren Sie den Inhalt des folgenden Beispiels und speichern Sie ihn unter `tz-aware-dag.py`.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from datetime import datetime, timedelta
Import the Pendulum library.
import pendulum

Instantiate Pendulum and set your timezone.
local_tz = pendulum.timezone("America/Los_Angeles")

with DAG(
```

```

dag_id = "tz_test",
schedule_interval="0 12 * * **",
catchup=False,
start_date=datetime(2022, 1, 1, tzinfo=local_tz)
) as dag:
 bash_operator_task = BashOperator(
 task_id="tz_aware_task",
 dag=dag,
 bash_command="date"
)

```

3. Führen Sie Folgendes aus AWS CLI Befehl, um die DAG in den Bucket Ihrer Umgebung zu kopieren und dann die DAG mithilfe der Apache Airflow-Benutzeroberfläche auszulösen.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. Bei Erfolg geben Sie in den Aufgabenprotokollen für die folgende Ausgabe Folgendes aus `tz_aware_task` in der `tz_test` DAG:

```

[2022-08-01, 12:00:00 PDT] {{subprocess.py:74}} INFO - Running command: ['bash', '-c', 'date']
[2022-08-01, 12:00:00 PDT] {{subprocess.py:85}} INFO - Output:
[2022-08-01, 12:00:00 PDT] {{subprocess.py:89}} INFO - Mon Aug 1 12:00:00 PDT 2022
[2022-08-01, 12:00:00 PDT] {{subprocess.py:93}} INFO - Command exited with return code 0
[2022-08-01, 12:00:00 PDT] {{taskinstance.py:1280}} INFO - Marking task as SUCCESS. dag_id=tz_test, task_id=tz_aware_task, execution_date=20220801T190033, start_date=20220801T190035, end_date=20220801T190035
[2022-08-01, 12:00:00 PDT] {{local_task_job.py:154}} INFO - Task exited with return code 0
[2022-08-01, 12:00:00 PDT] {{local_task_job.py:264}} INFO - 0 downstream tasks scheduled from follow-on schedule check

```

## Als nächstes

- Erfahre mehr darüber, wie du die hochladstplugins.zip-Datei in diesem Beispiel in Ihren Amazon S3-Bucket in [Installation benutzerdefinierter Plugins](#).

# Erfrischung eines CodeArtifact Zeichen

Wenn du verwendest CodeArtifact Um Python-Abhängigkeiten zu installieren, benötigt Amazon MWAA ein aktives Token. Um Amazon MWAA den Zugriff auf eine zu ermöglichen CodeArtifact Zur Laufzeit können Sie ein Repository verwenden [Startskript](#) und stelle das ein [PIP\\_EXTRA\\_INDEX\\_URL](#) mit dem Token.

Im folgenden Thema wird beschrieben, wie Sie ein Startskript erstellen können, das [get\\_authorization\\_token](#) CodeArtifact API-Vorgang zum Abrufen eines neuen Tokens bei jedem Start oder jeder Aktualisierung Ihrer Umgebung.

## Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Codebeispiel](#)
- [Als nächstes](#)

## Version

- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

## Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Ein [Amazon MWAA-Umgebung](#).
- Ein [CodeArtifact Endlager](#) wo Sie Abhängigkeiten für Ihre Umgebung speichern.

## Berechtigungen

Um das zu aktualisieren CodeArtifact token und das Ergebnis in Amazon S3 schreiben Amazon MWAA muss in der Ausführungsrolle über die folgenden Berechtigungen verfügen.

- `DecodeArtifact:GetAuthorizationToken`Aktion ermöglicht Amazon MWAAs Abrufen eines neuen Tokens von CodeArtifact. Die folgende Richtlinie gewährt die Erlaubnis für jeden CodeArtifact Domain, die Sie erstellen. Sie können den Zugriff auf Ihre Domänen weiter einschränken, indem Sie den Ressourcenwert in der Anweisung ändern und nur die Domänen angeben, auf die Ihre Umgebung zugreifen soll.

```
{
 "Effect": "Allow",
 "Action": "codeartifact:GetAuthorizationToken",
 "Resource": "arn:aws:codeartifact:us-west-2:*:domain/*"
}
```

- `Dsts:GetServiceBearerToken`Aktion ist erforderlich, um das aufzurufen CodeArtifact [GetAuthorizationToken](#)API-Vorgang. Dieser Vorgang gibt ein Token zurück, das verwendet werden muss, wenn Sie einen Paketmanager wie pip mit CodeArtifact. Um einen Paketmanager mit einem zu verwenden CodeArtifact Repository, die Rolle der Ausführungsrolle in Ihrer Umgebung muss dies zulassen `sts:GetServiceBearerToken` wie in der folgenden Grundsatzklärung dargestellt.

```
{
 "Sid": "AllowServiceBearerToken",
 "Effect": "Allow",
 "Action": "sts:GetServiceBearerToken",
 "Resource": "*"
}
```

## Codebeispiel

In den folgenden Schritten wird beschrieben, wie Sie ein Startskript erstellen können, das CodeArtifact Token.

1. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `code_artifact_startup_script.sh`.

```
#!/bin/sh

Startup script for MWAAs, see https://docs.aws.amazon.com/mwaa/latest/userguide/using-startup-script.html
```

```
set -eu

setup code artifact endpoint and token
https://pip.pypa.io/en/stable/cli/pip_install/#cmdoption-0
https://docs.aws.amazon.com/mwaa/latest/userguide/samples-code-artifact.html
DOMAIN="amazon"
DOMAIN_OWNER="112233445566"
REGION="us-west-2"
REPO_NAME="MyRepo"
echo "Getting token for CodeArtifact with args: --domain $DOMAIN --region $REGION
--domain-owner $DOMAIN_OWNER"
TOKEN=$(aws codeartifact get-authorization-token --domain $DOMAIN --region $REGION
--domain-owner $DOMAIN_OWNER | jq -r '.authorizationToken')
echo "Setting Pip env var for '--index-url' to point to CodeArtifact"
export PIP_EXTRA_INDEX_URL="https://aws:$TOKEN@$DOMAIN-
$DOMAIN_OWNER.d.codeartifact.$REGION.amazonaws.com/pypi/$REPO_NAME/simple/"
echo "CodeArtifact startup setup complete"
```

2. Navigieren Sie zu dem Ordner, in dem Sie das Skript gespeichert haben. Benutzecpin einem neuen Eingabeaufforderungsfenster, um das Skript in Ihren Bucket hochzuladen. Ersetzen *dein-s3-Eimer* mit deinen Informationen.

```
$ aws s3 cp code_artifact_startup_script.sh s3://your-s3-bucket/
code_artifact_startup_script.sh
```

Bei Erfolg gibt Amazon S3 den URL-Pfad zum Objekt aus:

```
upload: ./code_artifact_startup_script.sh to s3://your-s3-bucket/
code_artifact_startup_script.sh
```

Nachdem Sie das Skript hochgeladen haben, wird Ihre Umgebung aktualisiert und das Skript beim Start ausgeführt.

## Als nächstes

- Erfahren Sie, wie Sie Startskripts verwenden, um Ihre Umgebung anzupassen in [the section called "Verwenden eines Startskripts"](#).
- Erfahren Sie in diesem Beispiel, wie Sie den DAG-Code auf das hochladendagsOrdner in Ihrem Amazon S3-Bucket in [Hinzufügen oder Aktualisieren von DAGs](#).



- Erfahren Sie mehr darüber, wie Sie das hochladenplugins.zipDatei in diesem Beispiel in Ihren Amazon S3-Bucket in [Installation benutzerdefinierter Plugins](#).

## Erstellen eines benutzerdefinierten Plugins mit Apache Hive und Hadoop

Amazon MWAA extrahiert den Inhalt einerplugins.zipzu/usr/local/airflow/plugins. Dies kann verwendet werden, um Binärdateien zu Ihren Containern hinzuzufügen. Darüber hinaus führt Apache Airflow den Inhalt von Python-Dateien in derpluginsOrdner unterInbetriebnahme—ermöglicht es Ihnen, Umgebungsvariablen festzulegen und zu ändern. Das folgende Beispiel führt Sie durch die Schritte zur Erstellung eines benutzerdefinierten Plug-ins mit Apache Hive und Hadoop in einer Amazon Managed Workflows for Apache Airflow-Umgebung. Es kann mit anderen benutzerdefinierten Plugins und Binärdateien kombiniert werden.

### Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Voraussetzungen](#)
- [Abhängigkeiten herunterladen](#)
- [Benutzerdefiniertes Plugin](#)
- [Plugins.zip](#)
- [Codebeispiel](#)
- [Airflow-Konfigurationsoptionen](#)
- [Als nächstes](#)

### Version

- Der Beispielcode auf dieser Seite kann verwendet werden mitApache Airflow v1in[Python 3.7](#).
- Sie können das Codebeispiel auf dieser Seite verwenden mitApache Airflow v2 und höherin[Python 3.10](#).

## Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Ein [Amazon MWAA-Umgebung](#).

## Berechtigungen

- Für die Verwendung des Codebeispiels auf dieser Seite sind keine zusätzlichen Berechtigungen erforderlich.

## Voraussetzungen

Um den Beispielcode auf dieser Seite zu verwenden, fügen Sie die folgenden Abhängigkeiten zu Ihrem `requirements.txt`. Weitere Informationen hierzu finden Sie unter [Python-Abhängigkeiten installieren](#).

### Apache Airflow v2

```
-c https://raw.githubusercontent.com/apache/airflow/constraints-2.0.2/
constraints-3.7.txt
apache-airflow-providers-amazon[apache.hive]
```

### Apache Airflow v1

```
apache-airflow[hive]==1.10.12
```

## Abhängigkeiten herunterladen

Amazon MWAA extrahiert den Inhalt von `plugins.zip` in `/usr/local/airflow/plugins` auf jedem Amazon MWAA-Scheduler und Worker-Container. Dies wird verwendet, um Binärdateien zu Ihrer Umgebung hinzuzufügen. Die folgenden Schritte beschreiben, wie Sie die für das benutzerdefinierte Plugin benötigten Dateien zusammenstellen.

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Sie Ihr Plugin erstellen möchten.  
Beispiele:

```
cd plugins
```

2. Herunterladen [Hadoop](#) von einem [Spiegel](#), zum Beispiel:

```
wget https://downloads.apache.org/hadoop/common/hadoop-3.3.0/hadoop-3.3.0.tar.gz
```

3. Herunterladen [Bienenstock](#) von einem [Spiegel](#), zum Beispiel:

```
wget https://downloads.apache.org/hive/hive-3.1.2/apache-hive-3.1.2-bin.tar.gz
```

4. Erstellen Sie ein Verzeichnis. Beispiele:

```
mkdir hive_plugin
```

5. Extrahieren Sie Hadoop.

```
tar -xvzf hadoop-3.3.0.tar.gz -C hive_plugin
```

6. Extrahieren Sie Hive.

```
tar -xvzf apache-hive-3.1.2-bin.tar.gz -C hive_plugin
```

## Benutzerdefiniertes Plugin

Apache Airflow führt beim Start den Inhalt von Python-Dateien im Plugins-Ordner aus. Dies wird verwendet, um Umgebungsvariablen festzulegen und zu ändern. In den folgenden Schritten wird der Beispielcode für das benutzerdefinierte Plugin beschrieben.

1. Navigieren Sie in der Befehlszeile zum `hive_plugin` Verzeichnis. Beispiele:

```
cd hive_plugin
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `hive_plugin.py` im `hive_plugin` Verzeichnis.

```
from airflow.plugins_manager import AirflowPlugin
import os
os.environ["JAVA_HOME"]="/usr/lib/jvm/jre"
os.environ["HADOOP_HOME"]='/usr/local/airflow/plugins/hadoop-3.3.0'
```

```
os.environ["HADOOP_CONF_DIR"]='/usr/local/airflow/plugins/hadoop-3.3.0/etc/hadoop'
os.environ["HIVE_HOME"]='/usr/local/airflow/plugins/apache-hive-3.1.2-bin'
os.environ["PATH"] = os.getenv("PATH") + ":/usr/local/airflow/plugins/
hadoop-3.3.0:/usr/local/airflow/plugins/apache-hive-3.1.2-bin/bin:/usr/local/
airflow/plugins/apache-hive-3.1.2-bin/lib"
os.environ["CLASSPATH"] = os.getenv("CLASSPATH") + ":/usr/local/airflow/plugins/
apache-hive-3.1.2-bin/lib"
class EnvVarPlugin(AirflowPlugin):
 name = 'hive_plugin'
```

3. Kopieren Sie den Inhalt des folgenden Textes und speichern Sie ihn lokal unter `.airflowignore` in der `hive_plugin` Verzeichnis.

```
hadoop-3.3.0
apache-hive-3.1.2-bin
```

## Plugins.zip

Die folgenden Schritte zeigen, wie Sie erstellen `plugins.zip`. Der Inhalt dieses Beispiels kann mit anderen Plugins und Binärdateien zu einem einzigen kombiniert werden `plugins.zip` Datei.

1. Navigieren Sie in der Befehlszeile zum `hive_plugin` Verzeichnis aus dem vorherigen Schritt.  
Beispiele:

```
cd hive_plugin
```

2. Zippen Sie den Inhalt in Ihrem `plugins` Ordner.

```
zip -r ../hive_plugin.zip ./
```

## Codebeispiel

In den folgenden Schritten wird beschrieben, wie Sie den DAG-Code erstellen, mit dem das benutzerdefinierte Plugin getestet wird.

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist.  
Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `hive.py`.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

with DAG(dag_id="hive_test_dag", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
 hive_test = BashOperator(
 task_id="hive_test",
 bash_command='hive --help'
)
```

## Airflow-Konfigurationsoptionen

Wenn Sie Apache Airflow v2 verwenden, fügen Sie hinzu `core.lazy_load_plugins : False` als Apache Airflow-Konfigurationsoption. Weitere Informationen finden Sie unter [Verwenden von Konfigurationsoptionen zum Laden von Plugins in 2](#).

### Als nächstes

- Erfahren Sie, wie Sie die hochladen `requirements.txt` Datei in diesem Beispiel in Ihren Amazon S3-Bucket in [Python-Abhängigkeiten installieren](#).
- Erfahren Sie, wie Sie den DAG-Code in diesem Beispiel auf `dags` Ordner in Ihrem Amazon S3-Bucket in [Hinzufügen oder Aktualisieren von DAGs](#).
- Erfahre mehr darüber, wie du die hochladst `plugins.zip` Datei in diesem Beispiel in Ihren Amazon S3-Bucket in [Installation benutzerdefinierter Plugins](#).

## Ein benutzerdefiniertes Plugin für Apache Airflow erstellen

### PythonVirtualenvOperator

Das folgende Beispiel zeigt, wie Sie den Apache Airflow `patchen.PythonVirtualenvOperator` mit einem benutzerdefinierten Plugin auf Amazon Managed Workflows für Apache Airflow.

## Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Voraussetzungen](#)
- [Benutzerdefinierter Plugin-Beispielcode](#)
- [Plugins.zip](#)
- [Codebeispiel](#)
- [Airflow-Konfigurationsoptionen](#)
- [Als nächstes](#)

## Version

- Der Beispielcode auf dieser Seite kann verwendet werden mit Apache Airflow v1 in [Python 3.7](#).
- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

## Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Ein [Amazon MWAA-Umgebung](#).

## Berechtigungen

- Für die Verwendung des Codebeispiels auf dieser Seite sind keine zusätzlichen Berechtigungen erforderlich.

## Voraussetzungen

Um den Beispielcode auf dieser Seite zu verwenden, fügen Sie die folgenden Abhängigkeiten zu Ihrem `requirements.txt`. Weitere Informationen hierzu finden Sie unter [Python-Abhängigkeiten installieren](#).

```
virtualenv
```

## Benutzerdefinierter Plugin-Beispielcode

Apache Airflow führt beim Start den Inhalt von Python-Dateien im Plugins-Ordner aus. Dieses Plugin patcht das eingebaute `PythonVirtualenvOperator` während dieses Startvorgangs, um es mit Amazon MWAA kompatibel zu machen. Die folgenden Schritte zeigen den Beispielcode für das benutzerdefinierte Plugin.

### Apache Airflow v2

1. Navigieren Sie in der Befehlszeile zum `plugins` Verzeichnis oben. Beispiele:

```
cd plugins
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `virtual_python_plugin.py`.

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

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

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

from airflow.plugins_manager import AirflowPlugin
import airflow.utils.python_virtualenv
from typing import List

def _generate_virtualenv_cmd(tmp_dir: str, python_bin: str,
 system_site_packages: bool) -> List[str]:
```

```
cmd = ['python3', '/usr/local/airflow/.local/lib/python3.7/site-packages/
virtualenv', tmp_dir]
if system_site_packages:
 cmd.append('--system-site-packages')
if python_bin is not None:
 cmd.append(f'--python={python_bin}')
return cmd

airflow.utils.python_virtualenv._generate_virtualenv_cmd=_generate_virtualenv_cmd

class VirtualPythonPlugin(AirflowPlugin):
 name = 'virtual_python_plugin'
```

## Apache Airflow v1

1. Navigieren Sie in der Befehlszeile zum `plugins` Verzeichnis oben. Beispiele:

```
cd plugins
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `virtual_python_plugin.py`.

```
from airflow.plugins_manager import AirflowPlugin
from airflow.operators.python_operator import PythonVirtualenvOperator

def _generate_virtualenv_cmd(self, tmp_dir):
 cmd = ['python3', '/usr/local/airflow/.local/lib/python3.7/site-packages/
virtualenv', tmp_dir]
 if self.system_site_packages:
 cmd.append('--system-site-packages')
 if self.python_version is not None:
 cmd.append('--python=python{}'.format(self.python_version))
 return cmd
PythonVirtualenvOperator._generate_virtualenv_cmd=_generate_virtualenv_cmd

class EnvVarPlugin(AirflowPlugin):
 name = 'virtual_python_plugin'
```



## Plugins.zip

Die folgenden Schritte zeigen, wie Sie das erstellen `plugins.zip`.

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, das `virtual_python_plugin.py` enthält. Beispiele:

```
cd plugins
```

2. Zippen Sie den Inhalt in Ihren `plugins` Ordner.

```
zip plugins.zip virtual_python_plugin.py
```

## Codebeispiel

In den folgenden Schritten wird beschrieben, wie Sie den DAG-Code für das benutzerdefinierte Plugin erstellen.

### Apache Airflow v2

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist. Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `virtualenv_test.py`.

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

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

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
```

```
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""

from airflow import DAG
from airflow.operators.python import PythonVirtualenvOperator
from airflow.utils.dates import days_ago
import os

os.environ["PATH"] = os.getenv("PATH") + ":/usr/local/airflow/.local/bin"

def virtualenv_fn():
 import boto3
 print("boto3 version ",boto3.__version__)

with DAG(dag_id="virtualenv_test", schedule_interval=None, catchup=False,
start_date=days_ago(1)) as dag:
 virtualenv_task = PythonVirtualenvOperator(
 task_id="virtualenv_task",
 python_callable=virtualenv_fn,
 requirements=["boto3>=1.17.43"],
 system_site_packages=False,
 dag=dag,
)
```

## Apache Airflow v1

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist.  
Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal  
unter `virtualenv_test.py`.

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

Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
```

```
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.
```

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

```
from airflow import DAG
from airflow.operators.python_operator import PythonVirtualenvOperator
from airflow.utils.dates import days_ago
import os

os.environ["PATH"] = os.getenv("PATH") + ":/usr/local/airflow/.local/bin"

def virtualenv_fn():
 import boto3
 print("boto3 version ",boto3.__version__)

with DAG(dag_id="virtualenv_test", schedule_interval=None, catchup=False,
start_date=days_ago(1)) as dag:
 virtualenv_task = PythonVirtualenvOperator(
 task_id="virtualenv_task",
 python_callable=virtualenv_fn,
 requirements=["boto3>=1.17.43"],
 system_site_packages=False,
 dag=dag,
)
```

## Airflow-Konfigurationsoptionen

Wenn Sie Apache Airflow v2 verwenden, fügen Sie hinzu `core.lazy_load_plugins : False` als Apache Airflow-Konfigurationsoption. Weitere Informationen finden Sie unter [Verwenden von Konfigurationsoptionen zum Laden von Plugins in 2](#).

## Als nächstes

- Erfahren Sie, wie Sie das hochladen `requirements.txt` Datei in diesem Beispiel in Ihren Amazon S3-Bucket in [Python-Abhängigkeiten installieren](#).

- Erfahren Sie, wie Sie den DAG-Code in diesem Beispiel auf `diedagsOrdner` in Ihrem Amazon S3-Bucket in [Hinzufügen oder Aktualisieren von DAGs](#).
- Erfahre mehr darüber, wie du die hochladstplugins.zip-Datei in diesem Beispiel in Ihren Amazon S3-Bucket in [Installation benutzerdefinierter Plugins](#).

## Aufrufen von DAGs mit einer Lambda-Funktion

Das folgende Codebeispiel verwendet eine [AWS Lambda](#)-Funktion, um ein Apache Airflow-CLI-Token abzurufen und einen gerichteten azyklischen Graphen (DAG) in einer Amazon MWAA-Umgebung aufzurufen.

### Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Abhängigkeiten](#)
- [Codebeispiel](#)

## Version

- Sie können das Codebeispiel auf dieser Seite mit Apache Airflow v2 und höher in [Python 3.10](#) verwenden.

## Voraussetzungen

Um dieses Codebeispiel zu verwenden, müssen Sie:

- Verwenden Sie den [öffentlichen Netzwerkzugriffsmodus](#) für Ihre [Amazon MWAA-Umgebung](#).
- Verwenden Sie eine [Lambda-Funktion](#), die die neueste Python-Laufzeit verwendet.

### Note

Wenn sich die Lambda-Funktion und Ihre Amazon MWAA-Umgebung in derselben VPC befinden, können Sie diesen Code in einem privaten Netzwerk verwenden.

Für diese Konfiguration benötigt die Ausführungsrolle der Lambda-Funktion die Erlaubnis, den CreateNetworkInterface API-Vorgang Amazon Elastic Compute Cloud (Amazon EC2) aufzurufen. Sie können diese Berechtigung mithilfe der [AWSLambdaVPAccessExecutionRole](#) AWS verwalteten Richtlinie erteilen.

## Berechtigungen

Um das Codebeispiel auf dieser Seite verwenden zu können, benötigt die Ausführungsrolle Ihrer Amazon MWAA-Umgebung Zugriff, um die `airflow:CreateCliToken` Aktion auszuführen. Sie können diese Berechtigung mithilfe der `AmazonMWAAAirflowCliAccess` AWS verwalteten Richtlinie erteilen:

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

Weitere Informationen finden Sie unter [Apache Airflow CLI-Richtlinie: AirflowCli AmazonMWAA Access](#).

## Abhängigkeiten

- Um dieses Codebeispiel mit Apache Airflow v2 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v2-Basisinstallation](#) in Ihrer Umgebung.

## Codebeispiel

1. Öffnen Sie die AWS Lambda Konsole unter <https://console.aws.amazon.com/lambda/>.
2. Wählen Sie Ihre Lambda-Funktion aus der Funktionsliste aus.

3. Kopieren Sie auf der Funktionsseite den folgenden Code und ersetzen Sie ihn durch die Namen Ihrer Ressourcen:

- YOUR\_ENVIRONMENT\_NAME— Der Name Ihrer Amazon MWAA-Umgebung.
- YOUR\_DAG\_NAME— Der Name der DAG, die Sie aufrufen möchten.

```
import boto3
import http.client
import base64
import ast
mwaa_env_name = 'YOUR_ENVIRONMENT_NAME'
dag_name = 'YOUR_DAG_NAME'
mwaa_cli_command = 'dags trigger'

client = boto3.client('mwaa')

def lambda_handler(event, context):
 # get web token
 mwaa_cli_token = client.create_cli_token(
 Name=mwaa_env_name
)

 conn = http.client.HTTPSConnection(mwaa_cli_token['WebServerHostname'])
 payload = mwaa_cli_command + " " + dag_name
 headers = {
 'Authorization': 'Bearer ' + mwaa_cli_token['CliToken'],
 'Content-Type': 'text/plain'
 }
 conn.request("POST", "/aws_mwaa/cli", payload, headers)
 res = conn.getresponse()
 data = res.read()
 dict_str = data.decode("UTF-8")
 mydata = ast.literal_eval(dict_str)
 return base64.b64decode(mydata['stdout'])
```

4. Wählen Sie Bereitstellen.
5. Wählen Sie Test, um Ihre Funktion mit der Lambda-Konsole aufzurufen.
6. Um zu überprüfen, ob Ihr Lambda Ihre DAG erfolgreich aufgerufen hat, verwenden Sie die Amazon MWAA-Konsole, um zur Apache Airflow-Benutzeroberfläche Ihrer Umgebung zu navigieren, und gehen Sie dann wie folgt vor:

- a. Suchen Sie auf der DAG-Seite Ihre neue Ziel-DAG in der Liste der DAGs.
- b. Überprüfen Sie unter Letzte Ausführung den Zeitstempel für die letzte DAG-Ausführung. Dieser Zeitstempel sollte genau mit dem letzten Zeitstempel für Ihre andere `invoke_dag` Umgebung übereinstimmen.
- c. Überprüfen Sie unter Letzte Aufgaben, ob die letzte Ausführung erfolgreich war.

## DAGs in verschiedenen Amazon MWAA-Umgebungen aufrufen

Im folgenden Codebeispiel wird ein Apache Airflow-CLI-Token erstellt. Der Code verwendet dann einen gerichteten azyklischen Graphen (DAG) in einer Amazon MWAA-Umgebung, um eine DAG in einer anderen Amazon MWAA-Umgebung aufzurufen.

### Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Abhängigkeiten](#)
- [Codebeispiel](#)

### Version

- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

### Voraussetzungen

Um das Codebeispiel auf dieser Seite zu verwenden, benötigen Sie Folgendes:

- Zwei [Amazon MWAA-Umgebungen](#) mit öffentlichem Netzwerkzugriff auf den Webserver, einschließlich Ihrer aktuellen Umgebung.
- Eine Beispiel-DAG, die in den Amazon Simple Storage Service (Amazon S3) -Bucket Ihrer Zielumgebung hochgeladen wurde.

## Berechtigungen

Um das Codebeispiel auf dieser Seite verwenden zu können, muss die Ausführungsrolle Ihrer Umgebung über die Berechtigung verfügen, ein Apache Airflow-CLI-Token zu erstellen. Sie können das anhängen `AWS verwaltete Richtlinie AmazonMWA AirflowCliAccess` um diese Erlaubnis zu erteilen.

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

Weitere Informationen finden Sie unter [Apache Airflow CLI-Richtlinie: AirflowCli AmazonMWA Access](#).

## Abhängigkeiten

- Um dieses Codebeispiel mit Apache Airflow v2 zu verwenden, sind keine zusätzlichen Abhängigkeiten erforderlich. Der Code verwendet die [Apache Airflow v2-Basisinstallation](#) auf deine Umgebung.

## Codebeispiel

Im folgenden Codebeispiel wird davon ausgegangen, dass Sie eine DAG in Ihrer aktuellen Umgebung verwenden, um eine DAG in einer anderen Umgebung aufzurufen.

1. Navigieren Sie in Ihrem Terminal zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist. Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `invoke_dag.py`. Ersetzen Sie die folgenden Werte durch Ihre Informationen.



- `your-new-environment-name`— Der Name der anderen Umgebung, in der Sie die DAG aufrufen möchten.
- `your-target-dag-id`— Die ID der DAG in der anderen Umgebung, die Sie aufrufen möchten.

```
from airflow.decorators import dag, task
import boto3
from datetime import datetime, timedelta
import os, requests

DAG_ID = os.path.basename(__file__).replace(".py", "")

@task()
def invoke_dag_task(**kwargs):
 client = boto3.client('mwa')
 token = client.create_cli_token(Name='your-new-environment-name')
 url = f"https://{token['WebServerHostname']}/aws_mwa/cli"
 body = 'dags trigger your-target-dag-id'
 headers = {
 'Authorization': 'Bearer ' + token['CliToken'],
 'Content-Type': 'text/plain'
 }
 requests.post(url, data=body, headers=headers)

@dag(
 dag_id=DAG_ID,
 schedule_interval=None,
 start_date=datetime(2022, 1, 1),
 dagrun_timeout=timedelta(minutes=60),
 catchup=False
)
def invoke_dag():
 t = invoke_dag_task()

invoke_dag_test = invoke_dag()
```

3. Führen Sie Folgendes ausAWS CLIBefehl, um die DAG in den Bucket Ihrer Umgebung zu kopieren und dann die DAG mithilfe der Apache Airflow-Benutzeroberfläche auszulösen.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. Wenn die DAG erfolgreich ausgeführt wird, sehen Sie in den Aufgabenprotokollen für eine Ausgabe ähnlich der folgenden `invoke_dag_task`.

```
[2022-01-01, 12:00:00 PDT] {{python.py:152}} INFO - Done. Returned value was: None
[2022-01-01, 12:00:00 PDT] {{taskinstance.py:1280}} INFO - Marking task as SUCCESS.
dag_id=invoke_dag, task_id=invoke_dag_task, execution_date=20220101T120000,
start_date=20220101T120000, end_date=20220101T120000
[2022-01-01, 12:00:00 PDT] {{local_task_job.py:154}} INFO - Task exited with return
code 0
[2022-01-01, 12:00:00 PDT] {{local_task_job.py:264}} INFO - 0 downstream tasks
scheduled from follow-on schedule check
```

Um zu überprüfen, ob Ihre DAG erfolgreich aufgerufen wurde, navigieren Sie zur Apache Airflow-Benutzeroberfläche für Ihre neue Umgebung und gehen Sie dann wie folgt vor:

- a. Auf der DAGs-Seite, suchen Sie Ihre neue Ziel-DAG in der Liste der DAGs.
- b. Unter **Letzter Lauf**, überprüfen Sie den Zeitstempel für den letzten DAG-Lauf. Dieser Zeitstempel sollte genau dem neuesten Zeitstempel für `invoke_dag` in deiner anderen Umgebung.
- c. Unter **Aktuelle Aufgaben**, überprüfen Sie, ob der letzte Lauf erfolgreich war.

## Amazon MWAA mit Amazon RDS für Microsoft SQL Server verwenden

Sie können Amazon Managed Workflows für Apache Airflow verwenden, um eine Verbindung zu einem herzustellen [RDS für SQL Server](#). Der folgende Beispielcode verwendet DAGs in einer Amazon Managed Workflows for Apache Airflow-Umgebung, um eine Verbindung zu einem Amazon RDS für Microsoft SQL Server herzustellen und Abfragen auf diesem auszuführen.

Themen

- [Version](#)
- [Voraussetzungen](#)
- [Abhängigkeiten](#)
- [Apache Airflow v2-Verbindung](#)
- [Codebeispiel](#)
- [Als nächstes](#)

## Version

- Der Beispielcode auf dieser Seite kann verwendet werden mit Apache Airflow v1 in [Python 3.7](#).
- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

## Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Ein [Amazon MWAA-Umgebung](#).
- Amazon MWAA und RDS for SQL Server werden in derselben Amazon VPC ausgeführt.
- Die VPC-Sicherheitsgruppen von Amazon MWAA und dem Server sind mit den folgenden Verbindungen konfiguriert:
  - Eine Regel für eingehenden Verkehr für den Port 1433 für Amazon RDS in der Sicherheitsgruppe von Amazon MWAA geöffnet
  - Oder eine ausgehende Regel für den Hafen von 1433 von Amazon MWAA nach RDS öffnen
- Apache Airflow Connection for RDS for SQL Server spiegelt den Hostnamen, Port, Benutzernamen und Passwort aus der Amazon RDS SQL Server-Datenbank wider, die im vorherigen Prozess erstellt wurden.

## Abhängigkeiten

Um den Beispielcode in diesem Abschnitt zu verwenden, fügen Sie die folgende Abhängigkeit zu Ihrem `requirements.txt`. Weitere Informationen hierzu finden Sie unter [Python-Abhängigkeiten installieren](#).

### Apache Airflow v2

```
apache-airflow-providers-microsoft-mssql==1.0.1
apache-airflow-providers-odbc==1.0.1
pymssql==2.2.1
```

## Apache Airflow v1

```
apache-airflow[mssql]==1.10.12
```

## Apache Airflow v2-Verbindung

Wenn Sie eine Verbindung in Apache Airflow v2 verwenden, stellen Sie sicher, dass das Airflow-Verbindungsobjekt die folgenden Schlüssel-Wert-Paare enthält:

1. Konn-ID:mssql\_default
2. Verbindungstyp:Amazon-Webdienste
3. Gastgeber: YOUR\_DB\_HOST
4. Schema:
5. Einloggen:Administrator
6. Passwort:
7. Hafen:1433
8. Zuschlag:

## Codebeispiel

1. Navigieren Sie in der Befehlszeile zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist.  
Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `mysql-server.py`.

```
"""
```

```
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
```

```
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
```

```
"""
```

```
import pymssql
import logging
import sys
from airflow import DAG
from datetime import datetime
from airflow.operators.mssql_operator import MsSqlOperator
from airflow.operators.python_operator import PythonOperator

default_args = {
 'owner': 'aws',
 'depends_on_past': False,
 'start_date': datetime(2019, 2, 20),
 'provide_context': True
}

dag = DAG(
 'mssql_conn_example', default_args=default_args, schedule_interval=None)

drop_db = MsSqlOperator(
 task_id="drop_db",
 sql="DROP DATABASE IF EXISTS testdb;",
 mssql_conn_id="mssql_default",
 autocommit=True,
 dag=dag
)

create_db = MsSqlOperator(
 task_id="create_db",
 sql="create database testdb;",
 mssql_conn_id="mssql_default",
 autocommit=True,
 dag=dag
)

create_table = MsSqlOperator(
 task_id="create_table",
 sql="CREATE TABLE testdb.dbo.pet (name VARCHAR(20), owner VARCHAR(20));",
 mssql_conn_id="mssql_default",
 autocommit=True,
```

```
 dag=dag
)

insert_into_table = MsSqlOperator(
 task_id="insert_into_table",
 sql="INSERT INTO testdb.dbo.pet VALUES ('Olaf', 'Disney');",
 mssql_conn_id="mssql_default",
 autocommit=True,
 dag=dag
)

def select_pet(**kwargs):
 try:
 conn = pymssql.connect(
 server='sampledb.<xxxxxxx>.<region>.rds.amazonaws.com',
 user='admin',
 password='<yoursupersecretpassword>',
 database='testdb'
)

 # Create a cursor from the connection
 cursor = conn.cursor()
 cursor.execute("SELECT * from testdb.dbo.pet")
 row = cursor.fetchone()

 if row:
 print(row)
 except:
 logging.error("Error when creating pymssql database connection: %s",
 sys.exc_info()[0])

select_query = PythonOperator(
 task_id='select_query',
 python_callable=select_pet,
 dag=dag,
)

drop_db >> create_db >> create_table >> insert_into_table >> select_query
```

## Als nächstes

- Erfahren Sie, wie Sie die hochgeladene `requirements.txt`-Datei in diesem Beispiel in Ihren Amazon S3-Bucket in [Python-Abhängigkeiten installieren](#).
- Erfahren Sie, wie Sie den DAG-Code in diesem Beispiel auf `diedags` Ordner in Ihrem Amazon S3-Bucket in [Hinzufügen oder Aktualisieren von DAGs](#).
- Entdecken Sie Beispielskripte und andere [Beispiele für pymssql-Module](#).
- Erfahren Sie mehr über das Ausführen von SQL-Code in einer bestimmten Microsoft SQL-Datenbank mithilfe des [mssql\\_operator](#) in der [Apache Airflow-Referenzhandbuch](#).

## Amazon MWAA mit Amazon EMR verwenden

Das folgende Codebeispiel zeigt, wie Sie eine Integration mithilfe von Amazon EMR und Amazon Managed Workflows für Apache Airflow aktivieren.

### Themen

- [Version](#)
- [Codebeispiel](#)

## Version

- Der Beispielcode auf dieser Seite kann mit Apache Airflow v1 in [Python 3.7](#) verwendet werden.

## Codebeispiel

```
"""
```

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

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

```
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
```

```
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""
from airflow import DAG

from airflow.contrib.operators.emr_add_steps_operator import EmrAddStepsOperator
from airflow.contrib.operators.emr_create_job_flow_operator import
EmrCreateJobFlowOperator
from airflow.contrib.sensors.emr_step_sensor import EmrStepSensor

from airflow.utils.dates import days_ago
from datetime import timedelta
import os

DAG_ID = os.path.basename(__file__).replace(".py", "")

DEFAULT_ARGS = {
 'owner': 'airflow',
 'depends_on_past': False,
 'email': ['airflow@example.com'],
 'email_on_failure': False,
 'email_on_retry': False,
}

SPARK_STEPS = [
 {
 'Name': 'calculate_pi',
 'ActionOnFailure': 'CONTINUE',
 'HadoopJarStep': {
 'Jar': 'command-runner.jar',
 'Args': ['/usr/lib/spark/bin/run-example', 'SparkPi', '10'],
 },
 },
]

JOB_FLOW_OVERRIDES = {
 'Name': 'my-demo-cluster',
 'ReleaseLabel': 'emr-5.30.1',
 'Applications': [
 {
 'Name': 'Spark'
 },
],
}
```



```

 'Instances': {
 'InstanceGroups': [
 {
 'Name': "Master nodes",
 'Market': 'ON_DEMAND',
 'InstanceRole': 'MASTER',
 'InstanceType': 'm5.xlarge',
 'InstanceCount': 1,
 },
 {
 'Name': "Slave nodes",
 'Market': 'ON_DEMAND',
 'InstanceRole': 'CORE',
 'InstanceType': 'm5.xlarge',
 'InstanceCount': 2,
 }
],
 'KeepJobFlowAliveWhenNoSteps': False,
 'TerminationProtected': False,
 'Ec2KeyName': 'mykeypair',
 },
 'VisibleToAllUsers': True,
 'JobFlowRole': 'EMR_EC2_DefaultRole',
 'ServiceRole': 'EMR_DefaultRole'
}

with DAG(
 dag_id=DAG_ID,
 default_args=DEFAULT_ARGS,
 dagrun_timeout=timedelta(hours=2),
 start_date=days_ago(1),
 schedule_interval='@once',
 tags=['emr'],
) as dag:

 cluster_creator = EmrCreateJobFlowOperator(
 task_id='create_job_flow',
 job_flow_overrides=JOB_FLOW_OVERRIDES
)

 step_adder = EmrAddStepsOperator(
 task_id='add_steps',
 job_flow_id="{{ task_instance.xcom_pull(task_ids='create_job_flow',
key='return_value') }}"

```

```
 aws_conn_id='aws_default',
 steps=SPARK_STEPS,
)

 step_checker = EmrStepSensor(
 task_id='watch_step',
 job_flow_id="{{ task_instance.xcom_pull('create_job_flow',
key='return_value') }}",
 step_id="{{ task_instance.xcom_pull(task_ids='add_steps',
key='return_value')[0] }}",
 aws_conn_id='aws_default',
)

 cluster_creator >> step_adder >> step_checker
```

## Amazon MWAA mit Amazon EKS verwenden

Das folgende Beispiel zeigt, wie Sie Amazon Managed Workflows für Apache Airflow mit Amazon EKS verwenden.

### Themen

- [Version](#)
- [Voraussetzungen](#)
- [Erstellen Sie einen öffentlichen Schlüssel für Amazon EC2](#)
- [Erstellen Sie den Cluster](#)
- [Erstelle einemwaaNamensraum](#)
- [Erstellen Sie eine Rolle fürmwaaNamensraum](#)
- [Eine IAM-Rolle für den Amazon EKS-Cluster erstellen und anhängen](#)
- [Erstellen Sie die Datei requirements.txt](#)
- [Erstellen Sie eine Identitätszuordnung für Amazon EKS](#)
- [Erstellen der kubeconfig](#)
- [Erstellen Sie eine DAG](#)
- [Fügen Sie die DAG hinzu und kube\\_config.yaml zum Amazon S3-Bucket](#)
- [Das Beispiel aktivieren und auslösen](#)

## Version

- Der Beispielcode auf dieser Seite kann verwendet werden mit Apache Airflow v1 in [Python 3.7](#).
- Sie können das Codebeispiel auf dieser Seite verwenden mit Apache Airflow v2 und höher in [Python 3.10](#).

## Voraussetzungen

Um das Beispiel in diesem Thema zu verwenden, benötigen Sie Folgendes:

- Ein [Amazon MWAA-Umgebung](#).
- eksctl. Weitere Informationen finden Sie unter [Installieren Sie eksctl](#).
- kubectl. Weitere Informationen finden Sie unter [kubectl installieren und einrichten](#). In einigen Fällen wird dies mit eksctl installiert.
- Ein EC2-Schlüsselpaar in der Region, in der Sie Ihre Amazon MWAA-Umgebung erstellen. Weitere Informationen finden Sie unter [Ein Schlüsselpaar erstellen oder importieren](#).

### Note

Wenn Sie eine verwendene `eksctl`-Befehl, Sie können einen einschließen `--profile` um ein anderes Profil als das Standardprofil anzugeben.

## Erstellen Sie einen öffentlichen Schlüssel für Amazon EC2

Verwenden Sie den folgenden Befehl, um einen öffentlichen Schlüssel aus Ihrem privaten Schlüsselpaar zu erstellen.

```
ssh-keygen -y -f myprivatekey.pem > mypublickey.pub
```

Weitere Informationen finden Sie unter [Den öffentlichen Schlüssel für Ihr Schlüsselpaar abrufen](#).

## Erstellen Sie den Cluster

Verwenden Sie den folgenden Befehl, um den Cluster zu erstellen. Wenn Sie einen benutzerdefinierten Namen für den Cluster wünschen oder ihn in einer anderen Region erstellen möchten, ersetzen Sie die Werte für Name und Region. Sie müssen den Cluster in derselben Region erstellen, in der Sie die Amazon MWAA-Umgebung erstellen. Ersetzen Sie die Werte für die Subnetze, sodass sie den Subnetzen in Ihrem Amazon VPC-Netzwerk entsprechen, die Sie für Amazon MWAA verwenden. Ersetzen Sie den Wert für `ssh-public-key` um dem von Ihnen verwendeten Schlüssel zu entsprechen. Sie können einen vorhandenen Schlüssel von Amazon EC2 verwenden, der sich in derselben Region befindet, oder einen neuen Schlüssel in derselben Region erstellen, in der Sie Ihre Amazon MWAA-Umgebung erstellen.

```
eksctl create cluster \
--name mwaa-eks \
--region us-west-2 \
--version 1.18 \
--nodegroup-name linux-nodes \
--nodes 3 \
--nodes-min 1 \
--nodes-max 4 \
--with-oidc \
--ssh-access \
--ssh-public-key MyPublicKey \
--managed \
--vpc-public-subnets "subnet-11111111111111111111, subnet-22222222222222222222" \
--vpc-private-subnets "subnet-33333333333333333333, subnet-44444444444444444444"
```

Es dauert einige Zeit, bis die Erstellung des Clusters abgeschlossen ist. Sobald der Vorgang abgeschlossen ist, können Sie mithilfe des folgenden Befehls überprüfen, ob der Cluster erfolgreich erstellt wurde und der IAM-OIDC-Anbieter konfiguriert ist:

```
eksctl utils associate-iam-oidc-provider \
--region us-west-2 \
--cluster mwaa-eks \
--approve
```

## Erstelle einen `mwaa` Namensraum

Nachdem Sie bestätigt haben, dass der Cluster erfolgreich erstellt wurde, verwenden Sie den folgenden Befehl, um einen Namespace für die Pods zu erstellen.

```
kubectl create namespace mwaa
```

## Erstellen Sie eine Rolle für `mwaa` Namensraum

Nachdem Sie den Namespace erstellt haben, erstellen Sie eine Rolle und eine Rollenbindung für einen Amazon MWAA-Benutzer auf EKS, der Pods in einem MWAA-Namespace ausführen kann. Wenn Sie einen anderen Namen für den Namespace verwendet haben, ersetzen Sie `mwaa` durch `-n mwaa` mit dem Namen, den du benutzt hast.

```
cat << EOF | kubectl apply -f - -n mwaa
kind: Role
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 name: mwaa-role
rules:
 - apiGroups:
 - ""
 - "apps"
 - "batch"
 - "extensions"
 resources:
 - "jobs"
 - "pods"
 - "pods/attach"
 - "pods/exec"
 - "pods/log"
 - "pods/portforward"
 - "secrets"
 - "services"
 verbs:
 - "create"
 - "delete"
 - "describe"
 - "get"
 - "list"
 - "patch"
 - "update"

kind: RoleBinding
apiVersion: rbac.authorization.k8s.io/v1
metadata:
 name: mwaa-role-binding
```

```
subjects:
- kind: User
 name: mwaa-service
roleRef:
 kind: Role
 name: mwaa-role
 apiGroup: rbac.authorization.k8s.io
EOF
```

Bestätigen Sie, dass die neue Rolle auf den Amazon EKS-Cluster zugreifen kann, indem Sie den folgenden Befehl ausführen. Achten Sie darauf, den richtigen Namen zu verwenden, falls Sie ihn nicht verwendet haben *mwaa*:

```
kubectl get pods -n mwaa --as mwaa-service
```

Sie sollten eine Meldung erhalten, die besagt:

```
No resources found in mwaa namespace.
```

## Eine IAM-Rolle für den Amazon EKS-Cluster erstellen und anhängen

Sie müssen eine IAM-Rolle erstellen und sie dann an den Amazon EKS (k8s) -Cluster binden, damit sie für die Authentifizierung über IAM verwendet werden kann. Die Rolle wird nur zur Anmeldung am Cluster verwendet und besitzt keine Berechtigungen für die Konsolen- oder API-Aufrufe.

Erstellen Sie eine neue Rolle für die Amazon MWAA-Umgebung, indem Sie die Schritte unter [Amazon MWAA-Ausführungsrolle](#). Anstatt jedoch die in diesem Thema beschriebenen Richtlinien zu erstellen und anzuhängen, fügen Sie die folgende Richtlinie hinzu:

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Effect": "Allow",
 "Action": "airflow:PublishMetrics",
 "Resource": "arn:aws:airflow:${MWAA_REGION}:${ACCOUNT_NUMBER}:environment/
${MWAA_ENV_NAME}"
 },
 {
 "Effect": "Deny",
 "Action": "s3:ListAllMyBuckets",
```

```

 "Resource": [
 "arn:aws:s3:::{MWAAS3_BUCKET}",
 "arn:aws:s3:::{MWAAS3_BUCKET}/*"
]
 },
 {
 "Effect": "Allow",
 "Action": [
 "s3:GetObject*",
 "s3:GetBucket*",
 "s3:List*"
],
 "Resource": [
 "arn:aws:s3:::{MWAAS3_BUCKET}",
 "arn:aws:s3:::{MWAAS3_BUCKET}/*"
]
 },
 {
 "Effect": "Allow",
 "Action": [
 "logs:CreateLogStream",
 "logs:CreateLogGroup",
 "logs:PutLogEvents",
 "logs:GetLogEvents",
 "logs:GetLogRecord",
 "logs:GetLogGroupFields",
 "logs:GetQueryResults",
 "logs:DescribeLogGroups"
],
 "Resource": [
 "arn:aws:logs:${MWAAS3_REGION}:${ACCOUNT_NUMBER}:log-group:airflow-
 ${MWAAS3_ENV_NAME}-*"
]
 },
 {
 "Effect": "Allow",
 "Action": "cloudwatch:PutMetricData",
 "Resource": "*"
 },
 {
 "Effect": "Allow",
 "Action": [
 "sqs:ChangeMessageVisibility",
 "sqs:DeleteMessage",

```

```

 "sqs:GetQueueAttributes",
 "sqs:GetQueueUrl",
 "sqs:ReceiveMessage",
 "sqs:SendMessage"
],
 "Resource": "arn:aws:sqs:${MWSAA_REGION}:*:airflow-celery-*"
},
{
 "Effect": "Allow",
 "Action": [
 "kms:Decrypt",
 "kms:DescribeKey",
 "kms:GenerateDataKey*",
 "kms:Encrypt"
],
 "NotResource": "arn:aws:kms:*:${ACCOUNT_NUMBER}:key/*",
 "Condition": {
 "StringLike": {
 "kms:ViaService": [
 "sqs.${MWSAA_REGION}.amazonaws.com"
]
 }
 }
},
{
 "Effect": "Allow",
 "Action": [
 "eks:DescribeCluster"
],
 "Resource": "arn:aws:eks:${MWSAA_REGION}:${ACCOUNT_NUMBER}:cluster/
${EKS_CLUSTER_NAME}"
}
]
}

```

Nachdem Sie die Rolle erstellt haben, bearbeiten Sie Ihre Amazon MWSAA-Umgebung, um die von Ihnen erstellte Rolle als Ausführungsrolle für die Umgebung zu verwenden. Um die Rolle zu ändern, bearbeiten Sie die zu verwendende Umgebung. Sie wählen die Ausführungsrolle unter Berechtigungen.

**Bekannte Probleme:**



- Es gibt ein bekanntes Problem mit Rollen-ARNs, deren Unterpfade sich nicht bei Amazon EKS authentifizieren können. Die Problemumgehung hierfür besteht darin, die Servicerolle manuell zu erstellen, anstatt die von Amazon MWAA selbst erstellte Rolle zu verwenden. Weitere Informationen finden Sie unter [Rollen mit Pfaden funktionieren nicht, wenn der Pfad in ihrem ARN in der aws-auth-Configmap enthalten ist](#)
- Wenn die Amazon MWAA-Serviceliste in IAM nicht verfügbar ist, müssen Sie eine alternative Servicerichtlinie wie Amazon EC2 auswählen und dann die Vertrauensrichtlinie der Rolle so aktualisieren, dass sie den folgenden Anforderungen entspricht:

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

Weitere Informationen finden Sie unter [So verwenden Sie Vertrauensrichtlinien mit IAM-Rollen](#).

## Erstellen Sie die Datei requirements.txt

Um den Beispielcode in diesem Abschnitt zu verwenden, stellen Sie sicher, dass Sie eine der folgenden Datenbankoptionen zu Ihrem hinzugefügt haben `requirements.txt`. Weitere Informationen hierzu finden Sie unter [Python-Abhängigkeiten installieren](#).

### Apache Airflow v2

```
kubernetes
apache-airflow[cncf.kubernetes]==3.0.0
```

## Apache Airflow v1

```
awscli
kubernetes==12.0.1
```

## Erstellen Sie eine Identitätszuordnung für Amazon EKS

Verwenden Sie den ARN für die Rolle, die Sie im folgenden Befehl erstellt haben, um eine Identitätszuordnung für Amazon EKS zu erstellen. Die Region ändern *deine Region* in die Region, in der Sie die Umgebung geschaffen haben. Ersetzen Sie den ARN für die Rolle und ersetzen Sie ihn schließlich *mwaa-execution-role* mit der Ausführungsrolle Ihrer Umgebung.

```
eksctl create iamidentitymapping \
--region your-region \
--cluster mwaa-eks \
--arn arn:aws:iam::111222333444:role/mwaa-execution-role \
--username mwaa-service
```

## Erstellen der **kubeconfig**

Verwenden Sie den folgenden Befehl, um die **kubeconfig**:

```
aws eks update-kubeconfig \
--region us-west-2 \
--kubeconfig ./kube_config.yaml \
--name mwaa-eks \
--alias aws
```

Wenn du beim Laufen ein bestimmtes Profil verwendet hast `update-kubeconfig` du musst das entfernen `env`: Der Datei `kube_config.yaml` wurde ein Abschnitt hinzugefügt, sodass er korrekt mit Amazon MWAA funktioniert. Löschen Sie dazu Folgendes aus der Datei und speichern Sie sie dann:

```
env:
- name: AWS_PROFILE
 value: profile_name
```

## Erstellen Sie eine DAG

Verwenden Sie das folgende Codebeispiel, um eine Python-Datei zu erstellen, z. B. `mwaas_pod_example.py` für die DAG.

### Apache Airflow v2

```
"""
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""

from airflow import DAG
from datetime import datetime
from airflow.providers.cncf.kubernetes.operators.kubernetes_pod import
 KubernetesPodOperator

default_args = {
 'owner': 'aws',
 'depends_on_past': False,
 'start_date': datetime(2019, 2, 20),
 'provide_context': True
}

dag = DAG(
 'kubernetes_pod_example', default_args=default_args, schedule_interval=None)

#use a kube_config stored in s3 dags folder for now
kube_config_path = '/usr/local/airflow/dags/kube_config.yaml'

podRun = KubernetesPodOperator(
 namespace="mwaas",
 image="ubuntu:18.04",
 cmds=["bash"],
```

```

arguments=["-c", "ls"],
labels={"foo": "bar"},
name="mwaa-pod-test",
task_id="pod-task",
get_logs=True,
dag=dag,
is_delete_operator_pod=False,
config_file=kube_config_path,
in_cluster=False,
cluster_context='aws'
)

```

## Apache Airflow v1

```

"""
Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.
Permission is hereby granted, free of charge, to any person obtaining a copy of
this software and associated documentation files (the "Software"), to deal in
the Software without restriction, including without limitation the rights to
use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of
the Software, and to permit persons to whom the Software is furnished to do so.
THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS
FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER
IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
"""

from airflow import DAG
from datetime import datetime
from airflow.contrib.operators.kubernetes_pod_operator import KubernetesPodOperator

default_args = {
 'owner': 'aws',
 'depends_on_past': False,
 'start_date': datetime(2019, 2, 20),
 'provide_context': True
}

dag = DAG(
 'kubernetes_pod_example', default_args=default_args, schedule_interval=None)

#use a kube_config stored in s3 dags folder for now

```

```
kube_config_path = '/usr/local/airflow/dags/kube_config.yaml'

podRun = KubernetesPodOperator(
 namespace="mwa",
 image="ubuntu:18.04",
 cmds=["bash"],
 arguments=["-c", "ls"],
 labels={"foo": "bar"},
 name="mwa-pod-test",
 task_id="pod-task",
 get_logs=True,
 dag=dag,
 is_delete_operator_pod=False,
 config_file=kube_config_path,
 in_cluster=False,
 cluster_context='aws'
)
```

## Fügen Sie die DAG hinzu und `kube_config.yaml` zum Amazon S3-Bucket

Geben Sie die von Ihnen erstellte DAG ein und `kube_config.yaml` Datei in den Amazon S3-Bucket für die Amazon MWA-Umgebung. Sie können Dateien entweder mit der Amazon S3-Konsole oder der AWS Command Line Interface.

## Das Beispiel aktivieren und auslösen

Aktivieren Sie das Beispiel in Apache Airflow und lösen Sie es dann aus.

Verwenden Sie nach erfolgreicher Ausführung und Fertigstellung den folgenden Befehl, um den Pod zu überprüfen:

```
kubectl get pods -n mwa
```

Die Ausgabe sollte folgendermaßen oder ähnlich aussehen:

```
NAME READY STATUS RESTARTS AGE
mwa-pod-test-aa11bb22cc3344445555666677778888 0/1 Completed 0 2m23s
```

Anschließend können Sie die Ausgabe des Pods mit dem folgenden Befehl überprüfen. Ersetzen Sie den Namenswert durch den Wert, der vom vorherigen Befehl zurückgegeben wurde:

```
kubectl logs -n mwaa mwaa-pod-test-aa11bb22cc3344445555666677778888
```

## Herstellen einer Verbindung zu Amazon ECS mithilfe des **ECSOperator**

Das Thema beschreibt, wie Sie das verwenden können `ECSOperator` von Amazon MWAA aus eine Verbindung zu einem Amazon Elastic Container Service (Amazon ECS) -Container herzustellen. In den folgenden Schritten fügen Sie der Ausführungsrolle Ihrer Umgebung die erforderlichen Berechtigungen hinzu. Verwenden Sie eine `AWS CloudFormation` Vorlage, um einen Amazon ECS Fargate-Cluster zu erstellen und schließlich eine DAG zu erstellen und hochzuladen, die eine Verbindung zu Ihrem neuen Cluster herstellt.

### Themen

- [Version](#)
- [Voraussetzungen](#)
- [Berechtigungen](#)
- [Erstellen Sie einen Amazon ECS-Cluster](#)
- [Codebeispiel](#)

### Version

- Sie können das Codebeispiel auf dieser Seite verwenden mit `Apache Airflow v2` und höher in [Python 3.10](#).

### Voraussetzungen

Um den Beispielcode auf dieser Seite verwenden zu können, benötigen Sie Folgendes:

- Ein [Amazon MWAA-Umgebung](#).

### Berechtigungen

- Die Ausführungsrolle für Ihre Umgebung erfordert die Erlaubnis, Aufgaben in Amazon ECS auszuführen. Sie können entweder das anhängen [Amazon ECS\\_FullAccess](#) AWS-verwaltete

Richtlinie für Ihre Ausführungsrolle oder erstellen Sie die folgende Richtlinie und fügen Sie sie Ihrer Ausführungsrolle hinzu.

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "VisualEditor0",
 "Effect": "Allow",
 "Action": [
 "ecs:RunTask",
 "ecs:DescribeTasks"
],
 "Resource": "*"
 },
 {
 "Action": "iam:PassRole",
 "Effect": "Allow",
 "Resource": [
 "*"
],
 "Condition": {
 "StringLike": {
 "iam:PassedToService": "ecs-tasks.amazonaws.com"
 }
 }
 }
]
}
```

- Zusätzlich zum Hinzufügen der erforderlichen Berechtigungen für die Ausführung von Aufgaben in Amazon ECS müssen Sie auch die CloudWatch-protokollierte Richtlinienerklärung in Ihrer Amazon MWAA-Ausführungsrolle, um den Zugriff auf die Amazon ECS-Aufgabenprotokollgruppe zu ermöglichen, wie im Folgenden dargestellt. Die Amazon ECS-Protokollgruppe wird erstellt von AWS CloudFormation-Vorlage in [the section called "Erstellen Sie einen Amazon ECS-Cluster"](#).

```
{
 "Effect": "Allow",
 "Action": [
 "logs:CreateLogStream",
 "logs:CreateLogGroup",
 "logs:PutLogEvents",
```

```

 "logs:GetLogEvents",
 "logs:GetLogRecord",
 "logs:GetLogGroupFields",
 "logs:GetQueryResults"
],
 "Resource": [
 "arn:aws:logs:region:account-id:log-group:airflow-environment-name-*",
 "arn:aws:logs:*:*:log-group:ecs-mwaa-group:"
]
}

```

Weitere Informationen zur Amazon MWAA-Ausführungsrolle und zum Anhängen einer Richtlinie finden Sie unter [Ausführungsrolle](#).

## Erstellen Sie einen Amazon ECS-Cluster

Unter Verwendung des Folgenden AWS CloudFormation Vorlage, Sie erstellen einen Amazon ECS Fargate-Cluster zur Verwendung mit Ihrem Amazon MWAA-Workflow. Weitere Informationen finden Sie unter [Eine Aufgabendefinition erstellen](#) in der Amazon Elastic Container Service — Entwicklerleitfaden.

1. Erstellen Sie eine JSON-Datei mit dem folgenden Code und speichern Sie sie unter `ecs-mwaa-cfn.json`.

```

{
 "AWSTemplateFormatVersion": "2010-09-09",
 "Description": "This template deploys an ECS Fargate cluster with an Amazon Linux image as a test for MWAA.",
 "Parameters": {
 "VpcId": {
 "Type": "AWS::EC2::VPC::Id",
 "Description": "Select a VPC that allows instances access to ECR, as used with MWAA."
 },
 "SubnetIds": {
 "Type": "List<AWS::EC2::Subnet::Id>",
 "Description": "Select at two private subnets in your selected VPC, as used with MWAA."
 },
 "SecurityGroups": {

```



```

 "Type": "List<AWS::EC2::SecurityGroup::Id>",
 "Description": "Select at least one security group in your selected
VPC, as used with MAAA."
 }
},
"Resources": {
 "Cluster": {
 "Type": "AWS::ECS::Cluster",
 "Properties": {
 "ClusterName": {
 "Fn::Sub": "${AWS::StackName}-cluster"
 }
 }
 },
 "LogGroup": {
 "Type": "AWS::Logs::LogGroup",
 "Properties": {
 "LogGroupName": {
 "Ref": "AWS::StackName"
 },
 "RetentionInDays": 30
 }
 },
 "ExecutionRole": {
 "Type": "AWS::IAM::Role",
 "Properties": {
 "AssumeRolePolicyDocument": {
 "Statement": [
 {
 "Effect": "Allow",
 "Principal": {
 "Service": "ecs-tasks.amazonaws.com"
 },
 "Action": "sts:AssumeRole"
 }
]
 },
 "ManagedPolicyArns": [
 "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
]
 }
 },
 "TaskDefinition": {

```

```
 "Type": "AWS::ECS::TaskDefinition",
 "Properties": {
 "Family": {
 "Fn::Sub": "${AWS::StackName}-task"
 },
 "Cpu": 2048,
 "Memory": 4096,
 "NetworkMode": "awsvpc",
 "ExecutionRoleArn": {
 "Ref": "ExecutionRole"
 },
 "ContainerDefinitions": [
 {
 "Name": {
 "Fn::Sub": "${AWS::StackName}-container"
 },
 "Image": "137112412989.dkr.ecr.us-east-1.amazonaws.com/
amazonlinux:latest",
 "PortMappings": [
 {
 "Protocol": "tcp",
 "ContainerPort": 8080,
 "HostPort": 8080
 }
],
 "LogConfiguration": {
 "LogDriver": "awslogs",
 "Options": {
 "awslogs-region": {
 "Ref": "AWS::Region"
 },
 "awslogs-group": {
 "Ref": "LogGroup"
 },
 "awslogs-stream-prefix": "ecs"
 }
 }
 }
],
 "RequiresCompatibilities": [
 "FARGATE"
]
 }
 },
```

```

 "Service": {
 "Type": "AWS::ECS::Service",
 "Properties": {
 "ServiceName": {
 "Fn::Sub": "${AWS::StackName}-service"
 },
 "Cluster": {
 "Ref": "Cluster"
 },
 "TaskDefinition": {
 "Ref": "TaskDefinition"
 },
 "DesiredCount": 1,
 "LaunchType": "FARGATE",
 "PlatformVersion": "1.3.0",
 "NetworkConfiguration": {
 "AwsvpcConfiguration": {
 "AssignPublicIp": "ENABLED",
 "Subnets": {
 "Ref": "SubnetIds"
 },
 "SecurityGroups": {
 "Ref": "SecurityGroups"
 }
 }
 }
 }
 }
 }
}

```

2. Verwenden Sie in der Befehlszeile Folgendes AWS CLI-Befehl, um einen neuen Stapel zu erstellen. Sie müssen die Werte ersetzen `SecurityGroups` und `SubnetIds` mit Werten für die Sicherheitsgruppen und Subnetze Ihrer Amazon MWAA-Umgebung.

```

$ aws cloudformation create-stack \
 --stack-name my-ecs-stack --template-body file://ecs-mwaa-cfn.json \
 --parameters ParameterKey=SecurityGroups,ParameterValue=your-mwaa-security-group \
 ParameterKey=SubnetIds,ParameterValue=your-mwaa-subnet-1 \
 --capabilities CAPABILITY_IAM

```

Alternativ können Sie das folgende Shell-Skript verwenden. Das Skript ruft die erforderlichen Werte für die Sicherheitsgruppen und Subnetze Ihrer Umgebung ab, indem es den [get-environment](#) AWS CLIBefehl, erstellt dann den Stapel entsprechend. Gehen Sie wie folgt vor, um das Skript auszuführen.

- a. Kopieren Sie das Skript und speichern Sie es unter `ecs-stack-helper.sh` im selben Verzeichnis wie Ihre AWS CloudFormation-Vorlage.

```
#!/bin/bash

joinByString() {
 local separator="$1"
 shift
 local first="$1"
 shift
 printf "%s" "$first" "${@/#/$separator}"
}

response=$(aws mwa get-environment --name $1)

securityGroupId=$(echo "$response" | jq -r
'.Environment.NetworkConfiguration.SecurityGroupIds[]')
subnetIds=$(joinByString '\,' $(echo "$response" | jq -r
'.Environment.NetworkConfiguration.SubnetIds[]'))

aws cloudformation create-stack --stack-name $2 --template-body file://ecs-
cfn.json \
--parameters ParameterKey=SecurityGroups,ParameterValue=$securityGroupId \
ParameterKey=SubnetIds,ParameterValue=$subnetIds \
--capabilities CAPABILITY_IAM
```

- b. Führen Sie das Skript mit den folgenden Befehlen aus. Ersetzen Sie `environment-name` und `stack-name` mit Ihren Informationen.

```
$ chmod +x ecs-stack-helper.sh
$./ecs-stack-helper.sh environment-name stack-name
```

Wenn dies erfolgreich ist, wird die folgende Ausgabe angezeigt, in der Ihre neue AWS CloudFormation-Stapel-ID.

```
{
 "StackId": "arn:aws:cloudformation:us-west-2:123456789012:stack/my-ecs-
stack/123456e7-8ab9-01cd-b2fb-36cce63786c9"
}
```

Nach deinem AWS CloudFormation Stapel ist abgeschlossen und AWS hat Ihre Amazon ECS-Ressourcen bereitgestellt, Sie sind bereit, Ihre DAG zu erstellen und hochzuladen.

## Codebeispiel

1. Öffnen Sie eine Befehlszeile und navigieren Sie zu dem Verzeichnis, in dem Ihr DAG-Code gespeichert ist. Beispiele:

```
cd dags
```

2. Kopieren Sie den Inhalt des folgenden Codebeispiels und speichern Sie ihn lokal unter `mwa-ecs-operator.py` und laden Sie dann Ihre neue DAG auf Amazon S3 hoch.

```
from http import client
from airflow import DAG
from airflow.providers.amazon.aws.operators.ecs import ECSOperator
from airflow.utils.dates import days_ago
import boto3

CLUSTER_NAME="mwa-ecs-test-cluster" #Replace value for CLUSTER_NAME with your
information.
CONTAINER_NAME="mwa-ecs-test-container" #Replace value for CONTAINER_NAME with
your information.
LAUNCH_TYPE="FARGATE"

with DAG(
 dag_id = "ecs_fargate_dag",
 schedule_interval=None,
 catchup=False,
 start_date=days_ago(1)
) as dag:
 client=boto3.client('ecs')
 services=client.list_services(cluster=CLUSTER_NAME,launchType=LAUNCH_TYPE)

 service=client.describe_services(cluster=CLUSTER_NAME,services=services['serviceArns'])
```

```

ecs_operator_task = ECSOperator(
 task_id = "ecs_operator_task",
 dag=dag,
 cluster=CLUSTER_NAME,
 task_definition=service['services'][0]['taskDefinition'],
 launch_type=LAUNCH_TYPE,
 overrides={
 "containerOverrides": [
 {
 "name": CONTAINER_NAME,
 "command": ["ls", "-l", "/"],
 },
],
 },

 network_configuration=service['services'][0]['networkConfiguration'],
 awslogs_group="mwa-ecs-zero",
 awslogs_stream_prefix=f"ecs/{CONTAINER_NAME}",
)

```

### Note

Im Beispiel DAG, für `awslogs_group`, müssen Sie möglicherweise die Protokollgruppe mit dem Namen Ihrer Amazon ECS-Aufgabenprotokollgruppe ändern. Das Beispiel geht von einer Protokollgruppe mit dem Namen `ausmwa-ecs-zero`. Für `awslogs_stream_prefix`, verwenden Sie das Amazon ECS Task Log Stream-Präfix. Das Beispiel geht von einem Log-Stream-Präfix aus, `ecs`.

3. Führen Sie Folgendes aus AWS CLI-Befehl, um die DAG in den Bucket Ihrer Umgebung zu kopieren und dann die DAG mithilfe der Apache Airflow-Benutzeroberfläche auszulösen.

```
$ aws s3 cp your-dag.py s3://your-environment-bucket/dags/
```

4. Wenn dies erfolgreich ist, werden Sie in den Aufgabenprotokollen für eine Ausgabe ähnlich der folgenden sehen `ecs_operator_task` in `reecs_fargate_dag` TAG:

```
[2022-01-01, 12:00:00 UTC] {{ecs.py:300}} INFO - Running ECS Task -
Task definition: arn:aws:ecs:us-west-2:123456789012:task-definition/mwa-ecs-test-
task:1 - on cluster mwa-ecs-test-cluster
```

```

[2022-01-01, 12:00:00 UTC] {{ecs-operator-test.py:302}} INFO - ECSOperator
overrides:
{'containerOverrides': [{'name': 'mwa-ecs-test-container', 'command': ['ls', '-l',
'/']}]}}
.
.
.
[2022-01-01, 12:00:00 UTC] {{ecs.py:379}} INFO - ECS task ID is:
e012340b5e1b43c6a757cf012c635935
[2022-01-01, 12:00:00 UTC] {{ecs.py:313}} INFO - Starting ECS Task Log Fetcher
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] total
52
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC]
lrwxrwxrwx 1 root root 7 Jun 13 18:51 bin -> usr/bin
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] dr-xr-
xr-x 2 root root 4096 Apr 9 2019 boot
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x 5 root root 340 Jul 19 17:54 dev
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x 1 root root 4096 Jul 19 17:54 etc
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x 2 root root 4096 Apr 9 2019 home
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC]
lrwxrwxrwx 1 root root 7 Jun 13 18:51 lib -> usr/lib
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC]
lrwxrwxrwx 1 root root 9 Jun 13 18:51 lib64 -> usr/lib64
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x 2 root root 4096 Jun 13 18:51 local
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x 2 root root 4096 Apr 9 2019 media
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x 2 root root 4096 Apr 9 2019 mnt
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x 2 root root 4096 Apr 9 2019 opt
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] dr-xr-
xr-x 103 root root 0 Jul 19 17:54 proc
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] dr-xr-
x-\-\- 2 root root 4096 Apr 9 2019 root
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x 2 root root 4096 Jun 13 18:52 run
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC]
lrwxrwxrwx 1 root root 8 Jun 13 18:51 sbin -> usr/sbin
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-
xr-x 2 root root 4096 Apr 9 2019 srv

```

```
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] dr-xr-xr-x 13 root root 0 Jul 19 17:54 sys
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxrwxrwt 2 root root 4096 Jun 13 18:51 tmp
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-xr-x 13 root root 4096 Jun 13 18:51 usr
[2022-01-01, 12:00:00 UTC] {{ecs.py:119}} INFO - [2022-07-19, 17:54:03 UTC] drwxr-xr-x 18 root root 4096 Jun 13 18:52 var
.
.
.
[2022-01-01, 12:00:00 UTC] {{ecs.py:328}} INFO - ECS Task has been successfully executed
```

## Verwenden von dbt mit Amazon MWAA

Dieses Thema zeigt, wie Sie dbt und Postgres mit Amazon MWAA verwenden können. In den folgenden Schritten fügen Sie Ihrem `requirements.txt` die erforderlichen Abhängigkeiten hinzu und laden ein DBT-Beispielprojekt in den Amazon S3 S3-Bucket Ihrer Umgebung hoch. Anschließend überprüfen Sie anhand einer Beispiel-DAG, ob Amazon MWAA die Abhängigkeiten installiert hat, und verwenden die schließlich, `BashOperator` um das dbt-Projekt auszuführen.

### Themen

- [Version](#)
- [Voraussetzungen](#)
- [Abhängigkeiten](#)
- [Laden Sie ein dbt-Projekt auf Amazon S3 hoch](#)
- [Verwenden Sie eine DAG, um die Installation der DBT-Abhängigkeit zu überprüfen](#)
- [Verwenden Sie eine DAG, um ein DBT-Projekt auszuführen](#)

### Version

- Sie können das Codebeispiel auf dieser Seite mit Apache Airflow v2 und höher in [Python 3.10](#) verwenden.



## Voraussetzungen

Bevor Sie die folgenden Schritte ausführen können, benötigen Sie Folgendes:

- Eine [Amazon MWAA-Umgebung, die Apache Airflow v2.2.2](#) verwendet. Dieses Beispiel wurde mit Version 2.2.2 geschrieben und getestet. Möglicherweise müssen Sie das Beispiel ändern, um es mit anderen Apache Airflow-Versionen verwenden zu können.
- Ein DBT-Beispielprojekt. Um mit der Verwendung von dbt mit Amazon MWAA zu beginnen, können Sie einen Fork erstellen und das [dbt-Starter-Projekt aus dem dbt-labs-Repository](#) klonen. GitHub

## Abhängigkeiten

Um Amazon MWAA mit dbt zu verwenden, fügen Sie Ihrer Umgebung das folgende Startskript hinzu. Weitere Informationen finden Sie unter [Verwenden eines Startskripts mit Amazon MWAA](#).

```
#!/bin/bash

if [["${MWAA_AIRFLOW_COMPONENT}" != "worker"]]
then
 exit 0
fi

echo "-----"
echo "Installing virtual Python env"
echo "-----"

pip3 install --upgrade pip

echo "Current Python version:"
python3 --version
echo "..."

sudo pip3 install --user virtualenv
sudo mkdir python3-virtualenv
cd python3-virtualenv
sudo python3 -m venv dbt-env
sudo chmod -R 777 *

echo "-----"
echo "Activating venv in"
$DBT_ENV_PATH
```

```
echo "-----"

source dbt-env/bin/activate
pip3 list

echo "-----"
echo "Installing libraries..."
echo "-----"

do not use sudo, as it will install outside the venv
pip3 install dbt-redshift==1.6.1 dbt-postgres==1.6.1

echo "-----"
echo "Venv libraries..."
echo "-----"

pip3 list
dbt --version

echo "-----"
echo "Deactivating venv..."
echo "-----"

deactivate
```

In den folgenden Abschnitten laden Sie Ihr dbt-Projektverzeichnis auf Amazon S3 hoch und führen eine DAG aus, die überprüft, ob Amazon MWAA die erforderlichen dbt-Abhängigkeiten erfolgreich installiert hat.

## Laden Sie ein dbt-Projekt auf Amazon S3 hoch

Um ein dbt-Projekt mit Ihrer Amazon MWAA-Umgebung verwenden zu können, können Sie das gesamte Projektverzeichnis in den Ordner Ihrer Umgebung hochladen. Wenn die Umgebung aktualisiert wird, lädt Amazon MWAA das dbt-Verzeichnis in den lokalen Ordner herunter. `usr/local/airflow/dags/`

Um ein dbt-Projekt auf Amazon S3 hochzuladen

1. Navigieren Sie zu dem Verzeichnis, in dem Sie das dbt-Starterprojekt geklont haben.
2. Führen Sie den folgenden Amazon S3 AWS CLI S3-Befehl aus, um den Inhalt des Projekts mithilfe des `--recursive` Parameters rekursiv in den `dags` Ordner Ihrer Umgebung zu kopieren. Der Befehl erstellt ein Unterverzeichnis namens `dbt`, das Sie für all Ihre DBT-Projekte

verwenden können. Wenn das Unterverzeichnis bereits existiert, werden die Projektdateien in das bestehende Verzeichnis kopiert, und es wird kein neues Verzeichnis erstellt. Der Befehl erstellt auch ein Unterverzeichnis innerhalb des dbt Verzeichnisses für dieses spezielle Starterprojekt.

```
$ aws s3 cp dbt-starter-project s3://mwa-bucket/dags/dbt/dbt-starter-project --recursive
```

Sie können verschiedene Namen für Projektunterverzeichnisse verwenden, um mehrere DBT-Projekte innerhalb des übergeordneten Verzeichnisses zu organisieren. dbt

## Verwenden Sie eine DAG, um die Installation der DBT-Abhängigkeit zu überprüfen

Die folgende DAG verwendet einen `BashOperator` und einen Bash-Befehl, um zu überprüfen, ob Amazon MWAA die in angegebenen dbt-Abhängigkeiten erfolgreich installiert hat.

`requirements.txt`

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

with DAG(dag_id="dbt-installation-test", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
 cli_command = BashOperator(
 task_id="bash_command",
 bash_command="/usr/local/airflow/.local/bin/dbt --version"
)
```

Gehen Sie wie folgt vor, um die Task-Logs einzusehen und zu überprüfen, ob dbt und seine Abhängigkeiten installiert wurden.

1. Navigieren Sie zur Amazon MWAA-Konsole und wählen Sie dann Open Airflow UI aus der Liste der verfügbaren Umgebungen aus.
2. Suchen Sie auf der Apache Airflow-Benutzeroberfläche die `dbt-installation-test` DAG aus der Liste und wählen Sie dann das Datum unter der Last Run Spalte aus, an dem die letzte erfolgreiche Aufgabe geöffnet werden soll.

3. Wählen Sie in der Diagrammansicht die `bash_command` Aufgabe aus, um die Details der Aufgabeninstanz zu öffnen.
4. Wählen Sie Log, um die Task-Logs zu öffnen, und vergewissern Sie sich dann, dass die Logs die von uns angegebene DBT-Version erfolgreich auflisten. `requirements.txt`

## Verwenden Sie eine DAG, um ein DBT-Projekt auszuführen

Die folgende DAG verwendet `aBashOperator`, um die dbt-Projekte, die Sie auf Amazon S3 hochgeladen haben, aus dem lokalen `usr/local/airflow/dags/` Verzeichnis in das Verzeichnis mit Schreibzugriff zu `/tmp` kopieren, und führt dann das dbt-Projekt aus. Die Bash-Befehle gehen von einem DBT-Startprojekt mit dem Titel `aus.dbt-starter-project` Ändern Sie den Verzeichnisnamen entsprechend dem Namen Ihres Projektverzeichnisses.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

import os

DAG_ID = os.path.basename(__file__).replace(".py", "")

assumes all files are in a subfolder of DAGs called dbt

with DAG(dag_id=DAG_ID, schedule_interval=None, catchup=False, start_date=days_ago(1))
 as dag:
 cli_command = BashOperator(
 task_id="bash_command",
 bash_command="source /usr/local/airflow/python3-virtualenv/dbt-env/bin/
activate;\
cp -R /usr/local/airflow/dags/dbt /tmp;\
echo 'listing project files:';\
ls -R /tmp;\
cd /tmp/dbt/mwaa_dbt_test_project;\
/usr/local/airflow/python3-virtualenv/dbt-env/bin/dbt run --project-dir /tmp/dbt/
mwaa_dbt_test_project --profiles-dir ..;\
cat /tmp/dbt_logs/dbt.log;\
rm -rf /tmp/dbt/mwaa_dbt_test_project"
)
```

## AWS Blogs und Tutorials

- [Arbeiten mit Amazon EKS und Amazon MWAA für Apache Airflow v2.x](#)

# Bewährte Methoden für Amazon Managed Workflows für Apache Airflow

In diesem Handbuch werden die bewährten Methoden beschrieben, die wir für die Verwendung von Amazon Managed Workflows für Apache Airflow empfehlen.

Themen

- [Leistungsoptimierung für Apache Airflow auf Amazon MWAA](#)
- [Verwaltung von Python-Abhängigkeiten in requirements.txt](#)

## Leistungsoptimierung für Apache Airflow auf Amazon MWAA

Auf dieser Seite werden die bewährten Methoden beschrieben, die wir empfehlen, um die Leistung einer Amazon Managed Workflows for Apache Airflow-Umgebung zu optimieren. [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#)

Inhalt

- [Hinzufügen einer Apache Airflow-Konfigurationsoption](#)
- [Apache Airflow Scheduler](#)
  - [Parameter](#)
  - [Einschränkungen](#)
- [DAG-Ordner](#)
  - [Parameter](#)
- [DAG-Dateien](#)
  - [Parameter](#)
- [Aufgaben](#)
  - [Parameter](#)

## Hinzufügen einer Apache Airflow-Konfigurationsoption

Das folgende Verfahren führt Sie durch die Schritte zum Hinzufügen einer Airflow-Konfigurationsoption zu Ihrer Umgebung.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie Weiter aus.
5. Wählen Sie im Bereich mit den Airflow-Konfigurationsoptionen die Option Benutzerdefinierte Konfiguration hinzufügen aus.
6. Wählen Sie eine Konfiguration aus der Dropdownliste aus und geben Sie einen Wert ein, oder geben Sie eine benutzerdefinierte Konfiguration ein und geben Sie einen Wert ein.
7. Wählen Sie für jede Konfiguration, die Sie hinzufügen möchten, die Option Benutzerdefinierte Konfiguration hinzufügen aus.
8. Wählen Sie Speichern.

Weitere Informationen hierzu finden Sie unter [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#).

## Apache Airflow Scheduler

Der Apache Airflow Scheduler ist eine Kernkomponente von Apache Airflow. Ein Problem mit dem Scheduler kann verhindern, dass DAGs analysiert und Aufgaben geplant werden. Weitere Informationen zur Optimierung des Apache Airflow-Schedulers finden Sie unter [Feinabstimmung der Leistung Ihres Schedulers auf der Apache Airflow-Dokumentationswebsite](#).

### Parameter

In diesem Abschnitt werden die für den Apache Airflow Scheduler verfügbaren Konfigurationsoptionen und ihre Anwendungsfälle beschrieben.

#### Apache Airflow v2

| Version | Konfigurationsoption                                        | Standard | Beschreibung                                     | Anwendungsfall                                  |
|---------|-------------------------------------------------------------|----------|--------------------------------------------------|-------------------------------------------------|
| v2      | <a href="#">celery.py</a><br><a href="#">nc_parallelism</a> | 1        | Die Anzahl der Prozesse, die der Celery Executor | Sie können diese Option verwenden, um Warteschl |

| Version | Konfigurationsoption | Standard | Beschreibung                                          | Anwendungsfall                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------|----------------------|----------|-------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         |                      |          | verwendet , um den Aufgabenstatus zu synchronisieren. | <p>angenkonflikte zu vermeiden , indem Sie die vom Celery Executor verwendet en Prozesse einschränken. Standardmäßig ist ein Wert auf festgelegt, 1 um Fehler bei der Übermittlung von Aufgabenprotokollen an Logs zu CloudWatch verhindern. Wenn Sie den Wert auf festlegen, wird die maximale Anzahl von Prozessen verwendet, 0 was jedoch zu Fehlern bei der Übermittlung von Aufgabenprotokollen führen kann.</p> |



| Version | Konfigurationsoption                                                                      | Standard | Beschreibung                                                                                                                                  | Anwendungsfall                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------|-------------------------------------------------------------------------------------------|----------|-----------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#">scheduler</a><br><a href="#">.processo</a><br><a href="#">r_poll_interval</a> | 1        | Die Anzahl der Sekunden, die zwischen aufeinanderfolgenden Verarbeitungen von DAG-Dateien in der Scheduler-"Schleife" gewartet werden müssen. | Sie können diese Option verwenden, um CPU-Auslastung auf dem Scheduler zu verringern, indem Sie die Zeit verlängern, in der der Scheduler in den Ruhezustand versetzt wird, nachdem er die Ergebnisse der DAG-Analyse abgerufen, Aufgaben gesucht und in die Warteschlange gestellt hat, sowie Aufgaben in der Warteschlange im Executor ausgeführt hat. Wenn Sie diesen Wert erhöhen, wird die Anzahl der Scheduler-Threads verbraucht, |

| Version | Konfigurationsoption | Standard | Beschreibung | Anwendungsfall                                                                                                                                                                                                                                                                             |
|---------|----------------------|----------|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         |                      |          |              | die in einer Umgebung in scheduler.parsing_processes Apache Airflow v2 und Apache Airflow v1 ausgeführt werden. scheduler.max_threads Dies kann die Kapazität der Scheduler zum Parsen von DAGs verringern und die Zeit, die es dauert, bis DAGs auf dem Webserver erscheinen, verlängern. |

| Version | Konfigurationsoption                                     | Standard | Beschreibung                                                                                    | Anwendungsfall                                                                                                                                                             |
|---------|----------------------------------------------------------|----------|-------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#">scheduler.max_dagruns_to_create_per_loop</a> | 10       | Die maximale Anzahl von DAGs, für die pro Scheduler -"Schleife" erstellt werden sollen. DagRuns | Sie können diese Option verwenden, um Ressourcen für die Planung von Aufgaben freizugeben, indem Sie die maximale Anzahl DagRuns für die Scheduler -"Schleife" verringern. |

| Version | Konfigurationsoption                        | Standard | Beschreibung                                                                          | Anwendungsfall                                                                                                                                                                                                                                                                                                                                                                                                                           |
|---------|---------------------------------------------|----------|---------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#">scheduler.parsing.processes</a> | 2        | Die Anzahl der Threads, die der Scheduler parallel ausführen kann, um DAGs zu planen. | Sie können diese Option verwenden, um Ressourcen freizugeben, indem Sie die Anzahl der Prozesse verringern, die der Scheduler parallel zum Parsen von DAGs ausführt. Wir empfehlen, diese Zahl niedrig zu halten, wenn sich das DAG-Parsing auf die Aufgabenplanung auswirkt. Sie müssen einen Wert angeben, der unter der Anzahl der vCPUs in Ihrer Umgebung liegt. Weitere Informationen finden Sie unter <a href="#">Grenzwerte</a> . |

## Einschränkungen

In diesem Abschnitt werden die Grenzwerte beschrieben, die Sie bei der Anpassung der Standardparameter für den Scheduler berücksichtigen sollten.

`scheduler.parsing_processes`, `scheduler.max_threads`

Zwei Threads sind pro vCPU für eine Umgebungsklasse zulässig. Mindestens ein Thread muss für den Scheduler einer Umgebungsklasse reserviert sein. Wenn Sie eine Verzögerung bei der Planung von Aufgaben feststellen, müssen Sie möglicherweise Ihre [Umgebungsklasse](#) erhöhen. Eine große Umgebung verfügt beispielsweise über eine Fargate-Container-Instance mit 4 vCPUs als Scheduler. Das bedeutet, dass 7 insgesamt ein Maximum an Threads zur Verfügung steht, die für andere Prozesse verwendet werden können. Das heißt, zwei Threads multiplizieren vier vCPUs, minus einen für den Scheduler selbst. Der Wert, den Sie angeben, darf die Anzahl der für eine Umgebungsklasse verfügbaren Threads nicht überschreiten (wie unten dargestellt):

`scheduler.max_threads scheduler.parsing_processes`

- `mw1.small` — Der 1 Thread-Wert für andere Prozesse darf nicht überschritten werden. Der verbleibende Thread ist für den Scheduler reserviert.
- `mw1.medium` — Die Anzahl der 3 Threads für andere Prozesse darf nicht überschritten werden. Der verbleibende Thread ist für den Scheduler reserviert.
- `mw1.large` — Die Anzahl der 7 Threads für andere Prozesse darf nicht überschritten werden. Der verbleibende Thread ist für den Scheduler reserviert.

## DAG-Ordner

Der Apache Airflow Scheduler scannt kontinuierlich den DAGs-Ordner in Ihrer Umgebung. Alle enthaltenen `plugins.zip` Dateien oder Python (`.py`)-Dateien, die „Airflow“-Importanweisungen enthalten. Alle resultierenden Python-DAG-Objekte werden dann in eine `DagBagDatei` eingefügt, damit sie vom Scheduler verarbeitet werden kann, um zu bestimmen, welche Aufgaben, falls vorhanden, geplant werden müssen. Die Analyse von DAG-Dateien erfolgt unabhängig davon, ob die Dateien brauchbare DAG-Objekte enthalten.

## Parameter

In diesem Abschnitt werden die für den DAG-Ordner verfügbaren Konfigurationsoptionen und ihre Anwendungsfälle beschrieben.

## Apache Airflow v2

| Version | Konfigurationsoption                                | Standard     | Beschreibung                                                                                | Anwendungsfall                                                                                                                                                                                                                                                                                                                                            |
|---------|-----------------------------------------------------|--------------|---------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#">scheduler.dag_dir_list_interval</a>     | 300 Sekunden | Die Anzahl der Sekunden, für die der DAGs-Ordner nach neuen Dateien durchsucht werden soll. | Sie können diese Option verwenden, um Ressourcen freizugeben, indem Sie die Anzahl der Sekunden für die Analyse des DAGs-Ordners erhöhen. Wir empfehlen, diesen Wert zu erhöhen, wenn Sie lange Analysezeiten sehen. <code>total_parse_time</code> metrics, was möglicherweise auf eine große Anzahl von Dateien in Ihrem DAGs-Ordner zurückzuführen ist. |
| v2      | <a href="#">scheduler.min_file_process_interval</a> | 30 Sekunden  | Die Anzahl der Sekunden, nach denen der Scheduler eine DAG analysiert                       | Sie können diese Option verwenden, um Ressourcen freizugeben                                                                                                                                                                                                                                                                                              |

| Version | Konfigurationsoption | Standard | Beschreibung                                          | Anwendungsfall                                                                                                                                                                                                                                                                                            |
|---------|----------------------|----------|-------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         |                      |          | t und Aktualisierungen der DAG berücksichtigt werden. | en, indem Sie die Anzahl der Sekunden erhöhen, die der Scheduler wartet, bevor er eine DAG analysiert. Wenn Sie beispielsweise einen Wert von angeben30, wird die DAG-Datei alle 30 Sekunden analysiert. Wir empfehlen, diese Zahl hoch zu halten, um die CPU-Auslastung in Ihrer Umgebung zu verringern. |

## DAG-Dateien

Als Teil der Apache Airflow-Scheduler-Schleife werden einzelne DAG-Dateien analysiert, um DAG-Python-Objekte zu extrahieren. In Apache Airflow v2 und höher analysiert der Scheduler eine maximale Anzahl von [Parsing-Prozessen](#) gleichzeitig. Die in angegebene Anzahl von Sekunden `scheduler.min_file_process_interval` muss vergehen, bevor dieselbe Datei erneut analysiert wird.

## Parameter

In diesem Abschnitt werden die für Apache Airflow DAG-Dateien verfügbaren Konfigurationsoptionen und deren Anwendungsfälle beschrieben.

### Apache Airflow v2

| Version | Konfigurationsoption                            | Standard    | Beschreibung                                                                                                        | Anwendungsfall                                                                                                                                                                                                                                                                                                           |
|---------|-------------------------------------------------|-------------|---------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#">core.dag_file_processor_timeout</a> | 50 Sekunden | Die Anzahl der Sekunden, bevor der DagFileProcessor bei der Verarbeitung einer DAG-Datei das Timeout überschreitet. | Sie können diese Option verwenden, um Ressourcen freizugeben, indem Sie die Zeit verlängern, die bis zum Timeout des DagFileProcessors benötigt wird. Wir empfehlen, diesen Wert zu erhöhen, wenn in Ihren DAG-Verarbeitungsprotokollen Timeouts auftreten, die dazu führen, dass keine brauchbaren DAGs geladen werden. |



| Version | Konfigurationsoption                       | Standard    | Beschreibung                                                                  | Anwendungsfall                                                                                                                                                                                                                                                                                                                                                                                        |
|---------|--------------------------------------------|-------------|-------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#">core.dagbag_import_timeout</a> | 30 Sekunden | Die Anzahl der Sekunden vor dem Import einer Python-Datei wird überschritten. | Sie können diese Option verwenden, um Ressourcen freizugeben, indem Sie die Zeit erhöhen, die benötigt wird, bis der Scheduler beim Import einer Python-Datei zum Extrahieren der DAG-Objekte ein Timeout durchführt. Diese Option wird als Teil der Scheduler-"Schleife" verarbeitet und muss einen Wert enthalten, der unter dem angegebenen Wert liegt.<br><code>core.dagbag_import_timeout</code> |

| Version | Konfigurationsoption                                   | Standard | Beschreibung                                                                                      | Anwendungsfall                                                                                                                                                                                                                                                                                                                                                                                                |
|---------|--------------------------------------------------------|----------|---------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#">core.min_serialize_dag_update_interval</a> | 30       | Die Mindestanzahl von Sekunden, nach der serialisierte DAGs in der Datenbank aktualisiert werden. | Sie können diese Option verwenden, um Ressourcen freizugeben, indem Sie die Anzahl der Sekunden erhöhen, nach denen serialisierte DAGs in der Datenbank aktualisiert werden. Wir empfehlen, diesen Wert zu erhöhen, wenn Sie über eine große Anzahl von DAGs oder komplexe DAGs verfügen. Eine Erhöhung dieses Werts reduziert die Belastung des Schedulers und der Datenbank, wenn DAGs serialisiert werden. |

| Version | Konfigurationsoption                                          | Standard | Beschreibung                                                                                                                                 | Anwendungsfall                                                                                                                                                                                                                                                                                                                                                                                                            |
|---------|---------------------------------------------------------------|----------|----------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#"><u>core.min_serialize_dag_update_interval</u></a> | 10       | Die Anzahl der Sekunden, für die eine serialisierte DAG erneut aus der Datenbank abgerufen wird, wenn sie bereits in die geladen ist. DagBag | Sie können diese Option verwenden, um Ressourcen freizugeben, indem Sie die Anzahl der Sekunden erhöhen, für die eine serialisierte DAG erneut abgerufen wird. Der Wert muss höher als der unter angegebene Wert sein, <code>core.min_serialize_dag_update_interval</code> um die Schreibrate der Datenbank zu reduzieren. Eine Erhöhung dieses Werts reduziert die Belastung des Webservers und der Datenbank, wenn DAGs |

| Version | Konfigurationsoption | Standard | Beschreibung | Anwendungsfall       |
|---------|----------------------|----------|--------------|----------------------|
|         |                      |          |              | serialisiert werden. |

## Aufgaben

Sowohl der Apache Airflow Scheduler als auch die Worker sind an Aufgaben beteiligt, die in die Warteschlange gestellt und aus der Warteschlange entfernt werden. Der Scheduler versetzt analysierte Aufgaben, die zur Planung bereit sind, vom Status „Keine“ in den Status „Geplant“. Der Executor, der ebenfalls auf dem Scheduler-Container in Fargate läuft, stellt diese Aufgaben in die Warteschlange und setzt ihren Status auf In Warteschlange. Wenn die Mitarbeiter über Kapazitäten verfügen, nimmt er die Aufgabe aus der Warteschlange und setzt den Status auf Wird ausgeführt. Anschließend wird der Status auf Erfolgreich oder Fehlgeschlagen geändert, je nachdem, ob die Aufgabe erfolgreich war oder nicht.

## Parameter

In diesem Abschnitt werden die für Apache Airflow-Aufgaben verfügbaren Konfigurationsoptionen und ihre Anwendungsfälle beschrieben.

*Die Standardkonfigurationsoptionen, die Amazon MWAA überschreibt, sind rot markiert.*

### Apache Airflow v2

| Version | Konfigurationsoption    | Standard | Beschreibung                                                                           | Anwendungsfall                                                                                        |
|---------|-------------------------|----------|----------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------|
| v2      | <u>core.parallelism</u> | 10000    | Die maximale Anzahl von Task-Instanzen, die den Status „Wird ausgeführt“ haben können. | Sie können diese Option verwenden, um Ressourcen freizugeben, indem Sie die Anzahl der Task-Instanzen |

| Version | Konfigurationsoption | Standard | Beschreibung | Anwendungsfall                                                                                                                                                                                                                                                                                                                                             |
|---------|----------------------|----------|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         |                      |          |              | <p>anzahl erhöhen, die gleichzeitig ausgeführt werden können. Der angegebene Wert sollte der Anzahl der verfügbaren Worker „mal“ der Worker-Aufgabendichte entsprechen. Wir empfehlen, diesen Wert nur zu ändern, wenn Sie feststellen, dass eine große Anzahl von Aufgaben im Status „Wird ausgeführt“ oder „In Warteschlange“ stecken geblieben ist.</p> |

| Version | Konfigurationsoption                 | Standard | Beschreibung                                                                           | Anwendungsfall                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------|--------------------------------------|----------|----------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#">core.dag_concurrency</a> | 10000    | Die Anzahl der Task-Instanzen, die für jede DAG gleichzeitig ausgeführt werden dürfen. | Sie können diese Option verwenden, um Ressourcen freizugeben, indem Sie die Anzahl der Task-Instanzen erhöhen, die gleichzeitig ausgeführt werden dürfen. Wenn Sie beispielsweise einhundert DAGs mit zehn parallel Aufgaben haben und möchten, dass alle DAGs gleichzeitig ausgeführt werden, können Sie die maximale Parallelität als die Anzahl der verfügbaren Worker „mal“ der Worker-Aufgabendichte berechnen <i>celery.wo</i> |

| Version | Konfigurationsoption                                      | Standard | Beschreibung                                                                                                                             | Anwendungsfall                                                                                                                                                                           |
|---------|-----------------------------------------------------------|----------|------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         |                                                           |          |                                                                                                                                          | <i>rker_concurrency</i> , geteilt durch die Anzahl der DAGs (z. B. 100).                                                                                                                 |
| v2      | <a href="#">core.execute_tasks_new_python_interpreter</a> | True     | Bestimmt, ob Apache Airflow Aufgaben ausführt, indem es den übergeordneten Prozess forkt oder einen neuen Python-Prozess erstellt.       | Wenn diese Option auf gesetzt ist True, erkennt Apache Airflow Änderungen, die Sie an Ihren Plugins vornehmen, als neuen Python-Prozess, der zur Ausführung von Aufgaben erstellt wurde. |
| v2      | <a href="#">celery.worker_concurrency</a>                 | N/A      | Amazon MWAA überschreibt die Airflow-Basisinstallation für diese Option, um Workers als Teil seiner Autoscaling-Komponente zu skalieren. | <i>Jeder für diese Option angegebene Wert wird ignoriert.</i>                                                                                                                            |

| Version | Konfigurationsoption                    | Standard                                                                                            | Beschreibung                               | Anwendungsfall                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------|-----------------------------------------|-----------------------------------------------------------------------------------------------------|--------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| v2      | <a href="#">celery.worker_autoscale</a> | mw1.small - 5,0<br>mw1.mittel - 10,0<br>mw1.large - 20,0<br>mw1.xlarge - 40,0<br>mw1.2xlarge - 80,0 | Die Parallelität von Aufgaben für Workers. | Sie können diese Option verwenden, um Ressourcen freizugeben <i>enminimum</i> , indem Sie die Parallelität der <i>maximum</i> Aufgaben von Workers reduzieren. Mitarbeiter akzeptieren bis zu den konfigurierten <i>maximum</i> gleichzeitigen Aufgaben, unabhängig davon, ob genügend Ressourcen dafür zur Verfügung stehen. Wenn Aufgaben ohne ausreichende Ressourcen geplant werden, schlagen die Aufgaben sofort fehl. Es wird empfohlen, |



| Version | Konfigurationsoption | Standard | Beschreibung | Anwendungsfall                                                                                                                                                                      |
|---------|----------------------|----------|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         |                      |          |              | diesen Wert für resource intensive Aufgaben zu ändern, indem die Werte so reduziert werden, dass sie unter den Standardwerten liegen, um mehr Kapazität pro Aufgabe zu ermöglichen. |

## Verwaltung von Python-Abhängigkeiten in requirements.txt

Auf dieser Seite werden die bewährten Methoden beschrieben, die wir für die Installation und Verwaltung von Python-Abhängigkeiten in einer `requirements.txt` Datei für eine Amazon Managed Workflows for Apache Airflow-Umgebung empfehlen.

### Inhalt

- [Testen von DAGs mit dem Amazon MWAA CLI Utility](#)
- [Installation von Python-Abhängigkeiten mit dem PyPi .org-Anforderungsdateiformat](#)
  - [Option eins: Python-Abhängigkeiten aus dem Python-Paketindex](#)
  - [Option zwei: Python-Räder \(.whl\)](#)
    - [Verwenden der plugins.zip Datei in einem Amazon S3 S3-Bucket](#)
    - [Verwenden einer WHL-Datei, die auf einer URL gehostet wird](#)
    - [Erstellen von WHL-Dateien aus einer DAG](#)
  - [Option drei: Python-Abhängigkeiten, die auf einem privaten PyPi /PEP-503-konformen Repo gehostet werden](#)
- [Aktivieren von Protokollen auf der Amazon MWAA-Konsole](#)

- [Protokolle in der CloudWatch Logs-Konsole anzeigen](#)
- [Fehler in der Apache Airflow-Benutzeroberfläche anzeigen](#)
  - [Bei Apache Airflow anmelden](#)
- [Beispielszenarien requirements.txt](#)

## Testen von DAGs mit dem Amazon MWAA CLI Utility

- Das Befehlszeilenschnittstellenprogramm (CLI) repliziert eine Amazon Managed Workflows for Apache Airflow-Umgebung lokal.
- Die CLI erstellt lokal ein Docker-Container-Image, das einem Amazon MWAA-Produktionsimage ähnelt. Auf diese Weise können Sie eine lokale Apache Airflow-Umgebung ausführen, um DAGs, benutzerdefinierte Plugins und Abhängigkeiten zu entwickeln und zu testen, bevor Sie sie auf Amazon MWAA bereitstellen.
- Informationen zum Ausführen der CLI finden Sie unter [aws-mwaa-local-runner](#) on. GitHub

## Installation von Python-Abhängigkeiten mit dem PyPi .org-Anforderungsdateiformat

Im folgenden Abschnitt werden die verschiedenen Möglichkeiten beschrieben, Python-Abhängigkeiten gemäß dem PyPi .org [Requirements File Format](#) zu installieren.

### Option eins: Python-Abhängigkeiten aus dem Python-Paketindex

Im folgenden Abschnitt wird beschrieben, wie Python-Abhängigkeiten aus dem [Python-Paketindex](#) in einer `requirements.txt` Datei angegeben werden.

### Apache Airflow v2

1. Testen Sie lokal. Fügen Sie iterativ weitere Bibliotheken hinzu, um die richtige Kombination von Paketen und ihren Versionen zu finden, bevor Sie eine `requirements.txt` Datei erstellen. Informationen zum Ausführen des Amazon MWAA-CLI-Dienstprogramms finden Sie unter [aws-mwaa-local-runner](#) on. GitHub
2. Sehen Sie sich die Extras des Apache Airflow-Pakets an. Eine Liste der für Apache Airflow v2 auf Amazon MWAA installierten Pakete finden Sie auf der Website unter [Amazon MWAA Local Runner. requirements.txt](#) GitHub

3. Fügen Sie eine Beschränkungsanweisung hinzu. Fügen Sie die Einschränkungsdateri für Ihre Apache Airflow v2-Umgebung am Anfang Ihrer `requirements.txt` Datei hinzu. Apache Airflow-Einschränkungsdateien spezifizieren die Provider-Versionen, die zum Zeitpunkt einer Apache Airflow-Veröffentlichung verfügbar waren.

Ab Apache Airflow v2.7.2 muss Ihre Anforderungsdatei eine Erklärung enthalten. `--constraint` Wenn Sie keine Einschränkung angeben, gibt Amazon MWAA eine für Sie an, um sicherzustellen, dass die in Ihren Anforderungen aufgeführten Pakete mit der Version von Apache Airflow kompatibel sind, die Sie verwenden.

Ersetzen Sie im folgenden Beispiel `{environment-version}` durch die Versionsnummer Ihrer Umgebung und `{Python-version}` durch die *Version* von Python, die mit Ihrer Umgebung kompatibel ist.

Informationen zu der Version von Python, die mit Ihrer Apache Airflow-Umgebung kompatibel ist, finden Sie unter [Apache Airflow-Versionen](#).

```
--constraint "https://raw.githubusercontent.com/apache/airflow/
constraints-{Airflow-version}/constraints-{Python-version}.txt"
```

Wenn die Einschränkungsdateri feststellt, dass das `xyz==1.0` Paket nicht mit anderen Paketen in Ihrer Umgebung kompatibel ist, kann nicht verhindert `pip3 install` werden, dass inkompatible Bibliotheken in Ihrer Umgebung installiert werden. Wenn die Installation eines Pakets fehlschlägt, können Sie die Fehlerprotokolle für jede Apache Airflow-Komponente (den Scheduler, den Worker und den Webserver) im entsprechenden Protokollstream unter CloudWatch Logs einsehen. Weitere Informationen zu Protokolltypen finden Sie unter [the section called "Airflow-Protokolle anzeigen"](#)

4. Apache Airflow-Pakete. Fügen Sie die [Paket-Extras](#) und die Version (`==`) hinzu. Dies hilft zu verhindern, dass Pakete mit demselben Namen, aber unterschiedlicher Version in Ihrer Umgebung installiert werden.

```
apache-airflow[package-extra]==2.5.1
```

5. Python-Bibliotheken. Fügen Sie den Paketnamen und die Version (`==`) in Ihre `requirements.txt` Datei ein. Auf diese Weise wird verhindert, dass ein future aktuelles Update von [PyPi.org](#) automatisch angewendet wird.

```
library == version
```

## Example Boto3 und psycopg2-binary

Dieses Beispiel dient zu Demonstrationszwecken. Die Bibliotheken boto und psycopg2-binary sind in der Apache Airflow v2-Basisinstallation enthalten und müssen nicht in einer Datei angegeben werden. `requirements.txt`

```
boto3==1.17.54
boto==2.49.0
botocore==1.20.54
psycopg2-binary==2.8.6
```

[Wenn ein Paket ohne Version angegeben wird, installiert Amazon MWAA die neueste Version des Pakets von PyPi.org.](#) Diese Version kann zu Konflikten mit anderen Paketen in Ihrem führen. `requirements.txt`

## Apache Airflow v1

1. Testen Sie lokal. Fügen Sie iterativ weitere Bibliotheken hinzu, um die richtige Kombination von Paketen und ihren Versionen zu finden, bevor Sie eine `requirements.txt` Datei erstellen. Informationen zum Ausführen des Amazon MWAA-CLI-Dienstprogramms finden Sie unter [aws-mwaa-local-runner](#) on GitHub
2. Sehen Sie sich die Extras des Airflow-Pakets an. [Sehen Sie sich die Liste der für Apache Airflow v1.10.12 verfügbaren Pakete unter https://raw.githubusercontent.com/apache/airflow/constraints-1.10.12/constraints-3.7.txt an.](#)
3. Fügen Sie die Beschränkungsdatei hinzu. Fügen Sie die Einschränkungdatei für Apache Airflow v1.10.12 am Anfang Ihrer Datei hinzu. `requirements.txt` Wenn die Einschränkungdatei feststellt, dass das `xyz==1.0` Paket nicht mit anderen Paketen in Ihrer Umgebung kompatibel ist, kann sie nicht verhindern, dass inkompatible Bibliotheken in Ihrer Umgebung installiert werden. `pip3 install`

```
--constraint "https://raw.githubusercontent.com/apache/airflow/
constraints-1.10.12/constraints-3.7.txt"
```

4. Apache Airflow v1.10.12-Pakete. Fügen Sie die [Airflow-Paket-Extras und die Apache Airflow v1.10.12-Version](#) () hinzu. `==` Dadurch wird verhindert, dass Pakete mit demselben Namen, aber unterschiedlicher Version in Ihrer Umgebung installiert werden.

```
apache-airflow[package]==1.10.12
```

### Example Secure Shell (SSH)

Die folgende `requirements.txt` Beispieldatei installiert SSH für Apache Airflow v1.10.12.

```
apache-airflow[ssh]==1.10.12
```

5. Python-Bibliotheken. Fügen Sie den Paketnamen und die Version (==) in Ihre `requirements.txt` Datei ein. Auf diese Weise wird verhindert, dass ein future aktuelles Update von [PyPi.org](https://pypi.org) automatisch angewendet wird.

```
library == version
```

### Example Boto3

Die folgende `requirements.txt` Beispieldatei installiert die Boto3-Bibliothek für Apache Airflow v1.10.12.

```
boto3 == 1.17.4
```

[Wenn ein Paket ohne Version angegeben wird, installiert Amazon MWAA die neueste Version des Pakets von PyPi .org.](#) Diese Version kann zu Konflikten mit anderen Paketen in Ihrem führen. `requirements.txt`

## Option zwei: Python-Räder (.whl)

Ein Python-Rad ist ein Paketformat, das entwickelt wurde, um Bibliotheken mit kompilierten Artefakten auszuliefern. Wheel-Pakete als Methode zur Installation von Abhängigkeiten in Amazon MWAA bieten mehrere Vorteile:

- Schnellere Installation — Die WHL-Dateien werden als einzelne ZIP-Datei in den Container kopiert und dann lokal installiert, ohne dass jede Datei heruntergeladen werden muss.
- Weniger Konflikte — Sie können die Versionskompatibilität für Ihre Pakete im Voraus ermitteln. Daher ist es nicht erforderlich, rekursiv kompatible Versionen `pip` zu ermitteln.
- Höhere Stabilität — Bei extern gehosteten Bibliotheken können sich die nachgelagerten Anforderungen ändern, was zu Versionsinkompatibilität zwischen Containern in einer Amazon

MWAA-Umgebung führt. Da Abhängigkeiten nicht von einer externen Quelle abhängig sind, verfügt jeder Container über dieselben Bibliotheken, unabhängig davon, wann die einzelnen Container instanziiert werden.

Wir empfehlen die folgenden Methoden, um Python-Abhängigkeiten aus einem Python-Radarchiv (.whl) in Ihrem zu installieren `requirements.txt`.

### Methoden

- [Verwenden der `plugins.zip` Datei in einem Amazon S3 S3-Bucket](#)
- [Verwenden einer WHL-Datei, die auf einer URL gehostet wird](#)
- [Erstellen von WHL-Dateien aus einer DAG](#)

### Verwenden der `plugins.zip` Datei in einem Amazon S3 S3-Bucket

Der Apache Airflow-Scheduler, die Worker und der Webserver (für Apache Airflow v2.2.2 und höher) suchen beim Start auf dem AWS-verwalteten Fargate-Container für Ihre Umgebung unter nach benutzerdefinierten Plugins. `/usr/local/airflow/plugins/*` Dieser Prozess beginnt vor den Abhängigkeiten von Amazon MWAA `pip3 install -r requirements.txt` für Python und dem Start des Apache Airflow-Dienstes. Eine `plugins.zip` Datei kann für alle Dateien verwendet werden, die während der Ausführung der Umgebung nicht ständig geändert werden sollen oder für die Sie Benutzern, die DAGs schreiben, keinen Zugriff gewähren möchten. Zum Beispiel Raddateien für die Python-Bibliothek, Zertifikats-PEM-Dateien und YAML-Konfigurationsdateien.

Im folgenden Abschnitt wird beschrieben, wie Sie ein Rad, das sich in der `plugins.zip` Datei befindet, in Ihrem Amazon S3 S3-Bucket installieren.

1. Laden Sie die erforderlichen WHL-Dateien herunter, die Sie [pip download](#) mit Ihrem `requirements.txt` auf Amazon MWAA vorhandenen [Local-Runner](#) oder einem anderen [Amazon Linux 2-Container](#) verwenden können, um die erforderlichen Python-Wheel-Dateien aufzulösen und herunterzuladen.

```
$ pip3 download -r "$AIRFLOW_HOME/dags/requirements.txt" -d "$AIRFLOW_HOME/plugins"
$ cd "$AIRFLOW_HOME/plugins"
$ zip "$AIRFLOW_HOME/plugins.zip" *
```

2. Geben Sie den Pfad in Ihrem an. `requirements.txt` Geben Sie das Plugins-Verzeichnis oben in Ihrer Datei `requirements.txt` an [--find-links](#) und weisen Sie an, `pip` keine Installation aus anderen Quellen zu verwenden [--no-index](#), wie im Folgenden gezeigt

```
--find-links /usr/local/airflow/plugins
--no-index
```

### Example Rad in requirements.txt

Im folgenden Beispiel wird davon ausgegangen, dass Sie das Rad in eine `plugins.zip` Datei im Stammverzeichnis Ihres Amazon S3 S3-Buckets hochgeladen haben. Beispielsweise:

```
--find-links /usr/local/airflow/plugins
--no-index

numpy
```

Amazon MWAA ruft das `numpy-1.20.1-cp37-cp37m-manylinux1_x86_64.whl` Rad aus dem `plugins` Ordner ab und installiert es in Ihrer Umgebung.

### Verwenden einer WHL-Datei, die auf einer URL gehostet wird

Im folgenden Abschnitt wird beschrieben, wie Sie ein Rad installieren, das auf einer URL gehostet wird. Die URL muss entweder öffentlich zugänglich sein oder von der benutzerdefinierten Amazon VPC aus zugänglich sein, die Sie für Ihre Amazon MWAA-Umgebung angegeben haben.

- Geben Sie eine URL an. Geben Sie die URL zu einem Rad in Ihrem `anrequirements.txt`.

### Example Radarchiv auf einer öffentlichen URL

Im folgenden Beispiel wird ein Rad von einer öffentlichen Site heruntergeladen.

```
--find-links https://files.pythonhosted.org/packages/
--no-index
```

Amazon MWAA ruft das Rad von der von Ihnen angegebenen URL ab und installiert es in Ihrer Umgebung.

**Note**

Auf URLs kann nicht von privaten Webservern aus zugegriffen werden, die die Installationsanforderungen in Amazon MWAA v2.2.2 und höher erfüllen.

## Erstellen von WHL-Dateien aus einer DAG

Wenn Sie einen privaten Webserver haben, der Apache Airflow v2.2.2 oder höher verwendet, und Sie die Anforderungen nicht installieren können, weil Ihre Umgebung keinen Zugriff auf externe Repositories hat, können Sie die folgende DAG verwenden, um Ihre bestehenden Amazon MWAA-Anforderungen zu übernehmen und sie auf Amazon S3 zu packen:

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

S3_BUCKET = 'my-s3-bucket'
S3_KEY = 'backup/plugins_whl.zip'

with DAG(dag_id="create_whl_file", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
 cli_command = BashOperator(
 task_id="bash_command",
 bash_command=f"mkdir /tmp/whls;pip3 download -r /usr/local/airflow/
requirements/requirements.txt -d /tmp/whls;zip -j /tmp/plugins.zip /tmp/whls/*;aws s3
cp /tmp/plugins.zip s3://{S3_BUCKET}/{S3_KEY}"
)
```

Nachdem Sie die DAG ausgeführt haben, verwenden Sie diese neue Datei als Amazon MWAAplugins.zip, optional im Paket mit anderen Plug-ins. Aktualisieren Sie dann Ihre vorherige Version requirements.txt mit --find-links /usr/local/airflow/plugins und --no-index ohne Hinzufügen. --constraint

Mit dieser Methode können Sie dieselben Bibliotheken offline verwenden.



## Option drei: Python-Abhängigkeiten, die auf einem privaten PyPi /PEP-503-konformen Repo gehostet werden

Im folgenden Abschnitt wird beschrieben, wie Sie ein Apache Airflow-Extra installieren, das auf einer privaten URL mit Authentifizierung gehostet wird.

1. Fügen Sie Ihren Benutzernamen und Ihr Passwort als [Apache Airflow-Konfigurationsoptionen](#) hinzu. Beispielsweise:
  - `foo.user` : *YOUR\_USER\_NAME*
  - `foo.pass` : *YOUR\_PASSWORD*
2. Erstellen Sie Ihre `requirements.txt` Datei. Ersetzen Sie die Platzhalter im folgenden Beispiel durch Ihre private URL sowie den Benutzernamen und das Passwort, die Sie als [Apache Airflow-Konfigurationsoptionen](#) hinzugefügt haben. Beispielsweise:

```
--index-url https://${AIRFLOW__FOO__USER}:${AIRFLOW__FOO__PASS}@my.privatepypi.com
```

3. Fügen Sie Ihrer Datei weitere Bibliotheken hinzu `requirements.txt`. Beispielsweise:

```
--index-url https://${AIRFLOW__FOO__USER}:${AIRFLOW__FOO__PASS}@my.privatepypi.com
my-private-package==1.2.3
```

## Aktivieren von Protokollen auf der Amazon MWAA-Konsole

Die [Ausführungsrolle](#) für Ihre Amazon MWAA-Umgebung benötigt die Erlaubnis, Protokolle an Logs zu CloudWatch senden. Informationen zum Aktualisieren der Berechtigungen einer Ausführungsrolle finden Sie unter [Amazon MWAA-Ausführungsrolle](#)

Sie können Apache Airflow-Protokolle auf der CRITICAL Ebene INFO, WARNINGERROR, oder aktivieren. Wenn Sie eine Protokollebene wählen, sendet Amazon MWAA Protokolle für diese Stufe und alle höheren Schweregrade. Wenn Sie beispielsweise Protokolle auf der INFO Ebene aktivieren, sendet Amazon MWAA INFO Protokolle und WARNINGERROR, und Protokollebenen an CRITICAL CloudWatch Logs. Wir empfehlen, die Apache Airflow-Protokolle auf der INFO Ebene zu aktivieren, auf der der Scheduler die für den empfangenen Protokolle einsehen kann. `requirements.txt`

## Airflow scheduler logs

### Log level

Specify which types of task events to log

|                                                          |   |
|----------------------------------------------------------|---|
| <b>INFO</b><br>Log info and higher-severity events       | ▲ |
| <b>CRITICAL</b><br>Log critical events only              |   |
| <b>ERROR</b><br>Log error and higher-severity events     |   |
| <b>WARNING</b><br>Log warning and higher-severity events |   |
| <b>INFO</b><br>Log info and higher-severity events       |   |

## Protokolle in der CloudWatch Logs-Konsole anzeigen

Sie können die Apache Airflow-Protokolle für den Scheduler einsehen, Ihre Workflows planen und Ihren Ordner analysieren. In den folgenden Schritten wird beschrieben, wie Sie die Protokollgruppe für den Scheduler auf der Amazon MWAA-Konsole öffnen und Apache Airflow-Protokolle in der Logs-Konsole anzeigen. CloudWatch

Um Protokolle für ein anzuzeigen **requirements.txt**

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich Überwachung die Protokollgruppe Airflow Scheduler aus.
4. Wählen Sie unter **requirements\_install\_ip** Log-Streams die Option Log Streams aus.
5. Sie sollten die Liste der Pakete, die in der Umgebung installiert wurden, unter `finden/usr/local/airflow/.local/bin` finden. Beispielsweise:

```
Collecting appdirs==1.4.4 (from -r /usr/local/airflow/.local/bin (line 1))
```

```
Downloading https://files.pythonhosted.org/
packages/3b/00/2344469e2084fb28kjdsfiuyweb47389789vxbmbnjhsdgf5463acd6cf5e3db69324/
appdirs-1.4.4-py2.py3-none-any.whl
Collecting astroid==2.4.2 (from -r /usr/local/airflow/.local/bin (line 2))
```

6. Sehen Sie sich die Liste der Pakete an und überprüfen Sie, ob bei der Installation eines dieser Pakete ein Fehler aufgetreten ist. Wenn etwas schief gelaufen ist, wird möglicherweise ein Fehler ähnlich dem folgenden angezeigt:

```
2021-03-05T14:34:42.731-07:00
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/
airflow/.local/bin (line 4))
```

## Fehler in der Apache Airflow-Benutzeroberfläche anzeigen

Möglicherweise möchten Sie auch Ihre Apache Airflow-Benutzeroberfläche überprüfen, um festzustellen, ob ein Fehler möglicherweise mit einem anderen Problem zusammenhängt. Der häufigste Fehler, auf den Sie bei Apache Airflow auf Amazon MWAA stoßen können, ist:

```
Broken DAG: No module named x
```

Wenn Sie diesen Fehler in Ihrer Apache Airflow-Benutzeroberfläche sehen, fehlt Ihnen wahrscheinlich eine erforderliche Abhängigkeit in Ihrer Datei `requirements.txt`

### Bei Apache Airflow anmelden

Sie benötigen [Zugriffsrichtlinie für die Apache Airflow-Benutzeroberfläche: AmazonMWAA Access WebServer](#) Berechtigungen für Ihr AWS Konto in AWS Identity and Access Management (IAM), um Ihre Apache Airflow-Benutzeroberfläche anzeigen zu können.

Um auf Ihre Apache Airflow-Benutzeroberfläche zuzugreifen

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie „Airflow-Benutzeroberfläche öffnen“.

## Beispielszenarien `requirements.txt`

Sie können verschiedene Formate in Ihrem `requirements.txt` kombinieren. Das folgende Beispiel verwendet eine Kombination der verschiedenen Möglichkeiten zur Installation von Extras.

Example Extras auf PyPi .org und eine öffentliche URL

Sie müssen `--index-url` diese Option verwenden, wenn Sie Pakete von PyPi .org angeben, zusätzlich zu Paketen auf einer öffentlichen URL, wie z. B. benutzerdefinierte PEP 503-konforme Repo-URLs.

```
aws-batch == 0.6
phoenix-letter >= 0.3

--index-url http://dist.repoze.org/zope2/2.10/simple
zopelib
```

# Überwachung und Metriken für Amazon Managed Workflows for Apache Airflow

Die Überwachung ist ein wichtiger Bestandteil der Aufrechterhaltung der Zuverlässigkeit, Verfügbarkeit und Leistung von Amazon Managed Workflows for Apache Airflow und Ihrer AWS Lösung. Wir empfehlen, Überwachungsdaten aus allen Teilen Ihrer AWS Lösung zu sammeln, damit Sie einen Fehler an mehreren Stellen leichter debuggen können, falls einer auftritt. In diesem Thema wird beschrieben, welche Ressourcen zur Überwachung Ihrer Amazon MWAA-Umgebung und zur Reaktion auf potenzielle Ereignisse zur AWS Verfügung stehen.

## Note

Apache Airflow-Metriken und Protokollierung unterliegen den [CloudWatch Standardpreisen von Amazon](#).

Weitere Informationen zur Überwachung von Apache Airflow finden Sie unter [Logging & Monitoring](#) auf der Apache Airflow-Dokumentationswebsite.

## Sections

- [Überblick über die Überwachung auf Amazon MWAA](#)
- [Audit-Logs anzeigen AWS CloudTrail](#)
- [Airflow-Protokolle in Amazon anzeigen CloudWatch](#)
- [Überwachen von Dashboards und Alarmen auf Amazon MWAA](#)
- [Apache Airflow v2-Umgebungsmetriken in CloudWatch](#)
- [Container-, Warteschlangen- und Datenbankmetriken für Amazon MWAA](#)

## Überblick über die Überwachung auf Amazon MWAA

Auf dieser Seite werden die AWS Services beschrieben, die zur Überwachung einer Amazon Managed Workflows for Apache Airflow-Umgebung verwendet werden.

## Inhalt

- [CloudWatch Überblick über Amazon](#)

- [AWS CloudTrail Überblick](#)

## CloudWatch Überblick über Amazon

CloudWatch ist ein Metrik-Repository für AWS Services, mit dem Sie Statistiken abrufen können, die auf den von einem Service veröffentlichten [Metriken](#) und [Dimensionen](#) basieren. Sie können diese Metriken verwenden, um [Alarme](#) zu konfigurieren, Statistiken zu berechnen und die Daten dann in einem [Dashboard](#) zu präsentieren, das Ihnen hilft, den Zustand Ihrer Umgebung in der CloudWatch Amazon-Konsole zu beurteilen.

Apache Airflow ist bereits dafür eingerichtet, [StatsD-Metriken](#) für eine Amazon Managed Workflows für Apache Airflow-Umgebung an Amazon zu senden. CloudWatch

Weitere Informationen finden Sie unter [Was ist Amazon CloudWatch?](#) .

## AWS CloudTrail Überblick

CloudTrail ist ein Audit-Service, der eine Aufzeichnung der Aktionen bereitstellt, die von einem Benutzer, einer Rolle oder einem AWS Service in Amazon MWAA ausgeführt wurden. Anhand der von gesammelten Informationen können Sie die Anfrage CloudTrail, die an Amazon MWAA gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details, die in den Audit-Protokollen verfügbar sind, ermitteln.

Weitere Informationen finden Sie unter [Was ist? AWS CloudTrail](#) .

## Audit-Logs anzeigen AWS CloudTrail

AWS CloudTrail ist in Ihrem AWS Konto aktiviert, wenn Sie es erstellen. CloudTrail protokolliert die Aktivität einer IAM-Entität oder eines AWS Dienstes, z. B. Amazon Managed Workflows für Apache Airflow, die als CloudTrail Ereignis aufgezeichnet wird. Sie können den Ereignisverlauf der letzten 90 Tage in der Konsole anzeigen, suchen und herunterladen. CloudTrail erfasst alle Ereignisse auf der Amazon MWAA-Konsole und alle Aufrufe von Amazon MWAA-APIs. Es erfasst keine schreibgeschützten Aktionen wie, oder die Aktion. `GetEnvironment PublishMetrics` Auf dieser Seite wird beschrieben, wie CloudTrail Sie Ereignisse für Amazon MWAA überwachen können.

### Inhalt

- [Einen Trail erstellen in CloudTrail](#)

- [Ereignisse mit dem CloudTrail Ereignisverlauf anzeigen](#)
- [Beispielpfad für CreateEnvironment](#)
- [Als nächstes](#)

## Einen Trail erstellen in CloudTrail

Sie müssen einen Trail erstellen, um eine laufende Aufzeichnung von Ereignissen in Ihrem AWS Konto anzuzeigen, einschließlich Ereignisse für Amazon MWAA. Ein Trail ermöglicht CloudTrail die Übermittlung von Protokolldateien an einen Amazon S3 S3-Bucket. Wenn Sie keinen Trail erstellen, können Sie den verfügbaren Ereignisverlauf trotzdem in der CloudTrail Konsole einsehen. Anhand der von gesammelten Informationen können Sie beispielsweise die Anfrage CloudTrail, die an Amazon MWAA gestellt wurde, die IP-Adresse, von der aus die Anfrage gestellt wurde, wer die Anfrage gestellt hat, wann sie gestellt wurde, und weitere Details ermitteln. Weitere Informationen finden Sie im Abschnitt [Einen Trail für Ihr AWS Konto erstellen](#).

## Ereignisse mit dem CloudTrail Ereignisverlauf anzeigen

Sie können Betriebs- und Sicherheitsvorfälle der letzten 90 Tage in der CloudTrail Konsole beheben, indem Sie den Ereignisverlauf einsehen. Sie können beispielsweise Ereignisse im Zusammenhang mit der Erstellung, Änderung oder Löschung von Ressourcen (wie IAM-Benutzern oder anderen AWS Ressourcen) in Ihrem AWS Konto für jede Region einzeln anzeigen. Weitere Informationen finden Sie unter [Ereignisse mit CloudTrail Ereignisverlauf anzeigen](#).

1. Öffnen Sie die [CloudTrail-Konsole](#).
2. Wählen Sie „Ereignisverlauf“.
3. Wählen Sie die Ereignisse aus, die Sie sich ansehen möchten, und klicken Sie dann auf Veranstaltungsdetails vergleichen.

## Beispielpfad für **CreateEnvironment**

Ein Trail ist eine Konfiguration, durch die Ereignisse als Protokolldateien an den von Ihnen angegebenen Amazon-S3-Bucket übermittelt werden.

CloudTrail Protokolldateien enthalten einen oder mehrere Protokolleinträge. Ein Ereignis stellt eine einzelne Anforderung aus einer beliebigen Quelle dar und enthält Informationen über die angeforderte Aktion, z. B. Datum und Uhrzeit der Aktion oder Anforderungsparameter. CloudTrail

Protokolldateien sind kein geordneter Stack-Trace der öffentlichen API-Aufrufe und erscheinen auch nicht in einer bestimmten Reihenfolge. Das folgende Beispiel ist ein Protokolleintrag für die `CreateEnvironment` Aktion, die aufgrund fehlender Berechtigungen verweigert wurde. Die Werte in `AirflowConfigurationOptions` wurden aus Datenschutzgründen redigiert.

```
{
 "eventVersion": "1.05",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "00123456ABC7DEF8HIJK",
 "arn": "arn:aws:sts::012345678901:assumed-role/root/myuser",
 "accountId": "012345678901",
 "accessKeyId": "",
 "sessionContext": {
 "sessionIssuer": {
 "type": "Role",
 "principalId": "00123456ABC7DEF8HIJK",
 "arn": "arn:aws:iam::012345678901:role/user",
 "accountId": "012345678901",
 "userName": "user"
 },
 "webIdFederationData": {},
 "attributes": {
 "mfaAuthenticated": "false",
 "creationDate": "2020-10-07T15:51:52Z"
 }
 }
 },
 "eventTime": "2020-10-07T15:52:58Z",
 "eventSource": "airflow.amazonaws.com",
 "eventName": "CreateEnvironment",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "205.251.233.178",
 "userAgent": "PostmanRuntime/7.26.5",
 "errorCode": "AccessDenied",
 "requestParameters": {
 "SourceBucketArn": "arn:aws:s3:::my-bucket",
 "ExecutionRoleArn": "arn:aws:iam::012345678901:role/AirflowTaskRole",
 "AirflowConfigurationOptions": "****",
 "DagS3Path": "sample_dag.py",
 "NetworkConfiguration": {
 "SecurityGroupIds": [
 "sg-01234567890123456"
]
 }
 }
}
```



```
],
 "SubnetIds": [
 "subnet-01234567890123456",
 "subnet-65432112345665431"
]
 },
 "Name": "test-cloudtrail"
},
"responseElements": {
 "message": "Access denied."
},
"requestID": "RequestID",
"eventID": "EventID",
"readOnly": false,
"eventType": "AwsApiCall",
"recipientAccountId": "012345678901"
}
```

## Als nächstes

- Erfahren Sie unter [CloudTrail Unterstützte AWS Dienste und Integrationen](#), wie Sie andere Dienste für die in den CloudTrail Protokollen gesammelten Ereignisdaten konfigurieren.
- Erfahren Sie [unter Konfiguration von Amazon SNS-Benachrichtigungen für CloudTrail, wie Sie benachrichtigt werden, wenn neue Protokolldateien in einem Amazon S3-Bucket CloudTrail veröffentlicht werden](#).

## Airflow-Protokolle in Amazon anzeigen CloudWatch

Amazon MWAA kann Apache Airflow-Protokolle an Amazon senden. CloudWatch Sie können Protokolle für mehrere Umgebungen von einem einzigen Standort aus einsehen, um Verzögerungen oder Workflow-Fehler bei Apache Airflow leicht zu identifizieren, ohne dass zusätzliche Tools von Drittanbietern erforderlich sind. Apache Airflow-Protokolle müssen auf der Amazon Managed Workflows for Apache Airflow-Konsole aktiviert sein, um die Apache Airflow-DAG-Verarbeitung, die Aufgaben, den Webserver und die Worker-Anmeldungen anzeigen zu können. CloudWatch

### Inhalt

- [Preisgestaltung](#)
- [Bevor Sie beginnen](#)

- [Typen von Protokollen](#)
- [Apache Airflow-Protokolle aktivieren](#)
- [Apache Airflow-Protokolle anzeigen](#)
- [Beispiel für Scheduler-Protokolle](#)
- [Als nächstes](#)

## Preisgestaltung

- Es fallen Standardgebühren für CloudWatch Logs an. Weitere Informationen finden Sie unter [CloudWatch Preise](#).

## Bevor Sie beginnen

- Sie müssen über eine Rolle verfügen, die Logins einsehen kann CloudWatch. Weitere Informationen finden Sie unter [Zugreifen auf eine Amazon MWAA-Umgebung](#).

## Typen von Protokollen

Amazon MWAA erstellt für jede Airflow-Protokollierungsoption, die Sie aktivieren, eine Protokollgruppe und überträgt die Protokolle an die Protokollgruppen, die einer CloudWatch Umgebung zugeordnet sind. Protokollgruppen werden im folgenden Format benannt: `YourEnvironmentName-LogType` Wenn Ihre Umgebung beispielsweise benannt ist `Airflow-v202-Public`, werden Apache Airflow-Taskprotokolle an `Airflow-v202-Public-Task` gesendet.

| Protokolltyp                                   | Beschreibung                                                                                      |
|------------------------------------------------|---------------------------------------------------------------------------------------------------|
| <code>YourEnvironmentName-DAGProcessing</code> | Die Protokolle des DAG-Prozessor-Managers (der Teil des Schedulers, der DAG-Dateien verarbeitet). |
| <code>YourEnvironmentName-Scheduler</code>     | Die vom Airflow-Scheduler generierten Protokolle.                                                 |
| <code>YourEnvironmentName-Task</code>          | Die Aufgabenprotokolle, die eine DAG generiert                                                    |

| Protokolltyp                          | Beschreibung                                                           |
|---------------------------------------|------------------------------------------------------------------------|
| YourEnvironmentName- <b>WebServer</b> | Die von der Airflow-Weboberfläche generierten Protokolle.              |
| YourEnvironmentName- <b>Worker</b>    | Die im Rahmen der Workflow- und DAG-Ausführung generierten Protokolle. |

## Apache Airflow-Protokolle aktivieren

Sie können Apache Airflow-Protokolle auf der Ebene INFO, WARNINGERROR, oder CRITICAL aktivieren. Wenn Sie eine Protokollebene wählen, sendet Amazon MWAA Protokolle für diese Stufe und alle höheren Schweregrade. Wenn Sie beispielsweise Protokolle auf der INFO Ebene aktivieren, sendet Amazon MWAA INFO Protokolle und WARNINGERROR, und Protokollebenen an CRITICAL CloudWatch Logs.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie Bearbeiten aus.
4. Wählen Sie Weiter aus.
5. Wählen Sie eine oder mehrere der folgenden Protokollierungsoptionen:
  - a. Wählen Sie im Bereich Überwachung die Protokollgruppe Airflow Scheduler aus.
  - b. Wählen Sie im Bereich Überwachung die Airflow-Webserver-Protokollgruppe aus.
  - c. Wählen Sie im Bereich Überwachung die Airflow-Worker-Protokollgruppe aus.
  - d. Wählen Sie im Bereich Überwachung die Airflow DAG-Verarbeitungsprotokollgruppe aus.
  - e. Wählen Sie im Bereich Überwachung die Protokollgruppe Airflow Task aus.
  - f. Wählen Sie die Protokollierungsebene unter Protokollebene aus.
6. Wählen Sie Weiter aus.
7. Wählen Sie Speichern.

## Apache Airflow-Protokolle anzeigen

Im folgenden Abschnitt wird beschrieben, wie Apache Airflow-Protokolle in der CloudWatch Konsole angezeigt werden.

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich Überwachung eine Protokollgruppe aus.
4. Wählen Sie ein Protokoll im Protokollstream aus.

## Beispiel für Scheduler-Protokolle

Sie können Apache Airflow-Protokolle für den Scheduler einsehen, um Ihre Workflows zu planen und Ihren Ordner zu analysieren. In den folgenden Schritten wird beschrieben, wie Sie die Protokollgruppe für den Scheduler auf der Amazon MWAA-Konsole öffnen und Apache Airflow-Protokolle in der Logs-Konsole anzeigen. CloudWatch

Um Protokolle für ein anzuzeigen **requirements.txt**

1. Öffnen Sie die [Seite Umgebungen](#) auf der Amazon MWAA-Konsole.
2. Wählen Sie eine Umgebung aus.
3. Wählen Sie im Bereich Überwachung die Protokollgruppe Airflow Scheduler aus.
4. Wählen Sie unter **requirements\_install\_ip** Log-Streams die Option Log Streams aus.
5. Sie sollten die Liste der Pakete, die in der Umgebung installiert wurden, unter finden `/usr/local/airflow/.local/bin`. Beispielsweise:

```
Collecting appdirs==1.4.4 (from -r /usr/local/airflow/.local/bin (line 1))
Downloading https://files.pythonhosted.org/
packages/3b/00/2344469e2084fb28kjdsfiuyweb47389789vxbmnbjhsdgf5463acd6cf5e3db69324/
appdirs-1.4.4-py2.py3-none-any.whl
Collecting astroid==2.4.2 (from -r /usr/local/airflow/.local/bin (line 2))
```

6. Sehen Sie sich die Liste der Pakete an und überprüfen Sie, ob bei der Installation eines dieser Pakete ein Fehler aufgetreten ist. Wenn etwas schief gelaufen ist, wird möglicherweise ein Fehler ähnlich dem folgenden angezeigt:

```
2021-03-05T14:34:42.731-07:00
```

```
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/airflow/.local/bin (line 4))
No matching distribution found for LibraryName==1.0.0 (from -r /usr/local/airflow/.local/bin (line 4))
```

## Als nächstes

- Erfahren Sie [unter Amazon-Alarme verwenden, wie Sie einen CloudWatch CloudWatch Alarm konfigurieren](#).
- Erfahren Sie unter [CloudWatch Dashboards verwenden](#), wie Sie ein CloudWatch Dashboard erstellen.

## Überwachen von Dashboards und Alarmen auf Amazon MWAA

Sie können in Amazon ein benutzerdefiniertes Dashboard erstellen CloudWatch und Alarme für eine bestimmte Metrik hinzufügen, um den Status einer Amazon Managed Workflows for Apache Airflow-Umgebung zu überwachen. Wenn sich ein Alarm auf einem Dashboard befindet, wird er rot, wenn er sich im ALARM Status befindet, sodass Sie den Zustand einer Amazon MWAA-Umgebung leichter proaktiv überwachen können.

Apache Airflow stellt Metriken für eine Reihe von Prozessen bereit, darunter die Anzahl der DAG-Prozesse, die Größe des DAG-Beutels, aktuell ausgeführte Aufgaben, Aufgabenfehler und Erfolge. Wenn Sie eine Umgebung erstellen, ist Airflow so konfiguriert, dass automatisch Metriken für eine Amazon MWAA-Umgebung an gesendet werden. CloudWatch Auf dieser Seite wird beschrieben, wie Sie ein Status-Dashboard für die Airflow-Metriken in CloudWatch einer Amazon MWAA-Umgebung erstellen.

### Inhalt

- [Metriken](#)
- [Übersicht über die Alarmstatus](#)
- [Beispiel für benutzerdefinierte Dashboards und Alarme](#)
  - [Über diese Metriken](#)
  - [Über das Armaturenbrett](#)
  - [Mithilfe von Tutorials AWS](#)
  - [Verwenden AWS CloudFormation](#)

- [Löschen von Metriken und Dashboards](#)
- [Als nächstes](#)

## Metriken

Sie können ein benutzerdefiniertes Dashboard und einen Alarm für alle Metriken erstellen, die für Ihre Apache Airflow-Version verfügbar sind. Jede Metrik entspricht einem Apache Airflow Key Performance Indicator (KPI). Eine Liste von Metriken finden Sie unter:

- [Apache Airflow v2-Umgebungsmetriken in CloudWatch](#)

## Übersicht über die Alarmstatus

Ein Metrikalarm kann die folgenden Status aufweisen:

- OK – Die Metrik oder der Ausdruck liegt innerhalb des festgelegten Schwellenwerts.
- ALARM – Die Metrik oder der Ausdruck liegt außerhalb des festgelegten Schwellenwerts.
- INSUFFICIENT\_DATA – Der Alarm wurde soeben gestartet; die Metrik ist nicht verfügbar oder es sind nicht genügend Daten verfügbar, damit die Metrik den Alarmstatus bestimmen kann.

## Beispiel für benutzerdefinierte Dashboards und Alarme

Sie können ein benutzerdefiniertes Monitoring-Dashboard erstellen, das Diagramme mit ausgewählten Metriken für Ihre Amazon MWAA-Umgebung anzeigt.

### Über diese Metriken

In der folgenden Liste werden alle Metriken beschrieben, die im benutzerdefinierten Dashboard anhand des Tutorials und der Vorlagendefinitionen in diesem Abschnitt erstellt wurden.

- QueuedTasks- Die Anzahl der Aufgaben mit dem Status „Warteschlange“. Entspricht der `executor.queued_tasks` Apache Airflow-Metrik.
- TasksPending- Die Anzahl der Aufgaben, die im Executor noch ausstehen. Entspricht der `scheduler.tasks.pending` Apache Airflow-Metrik.

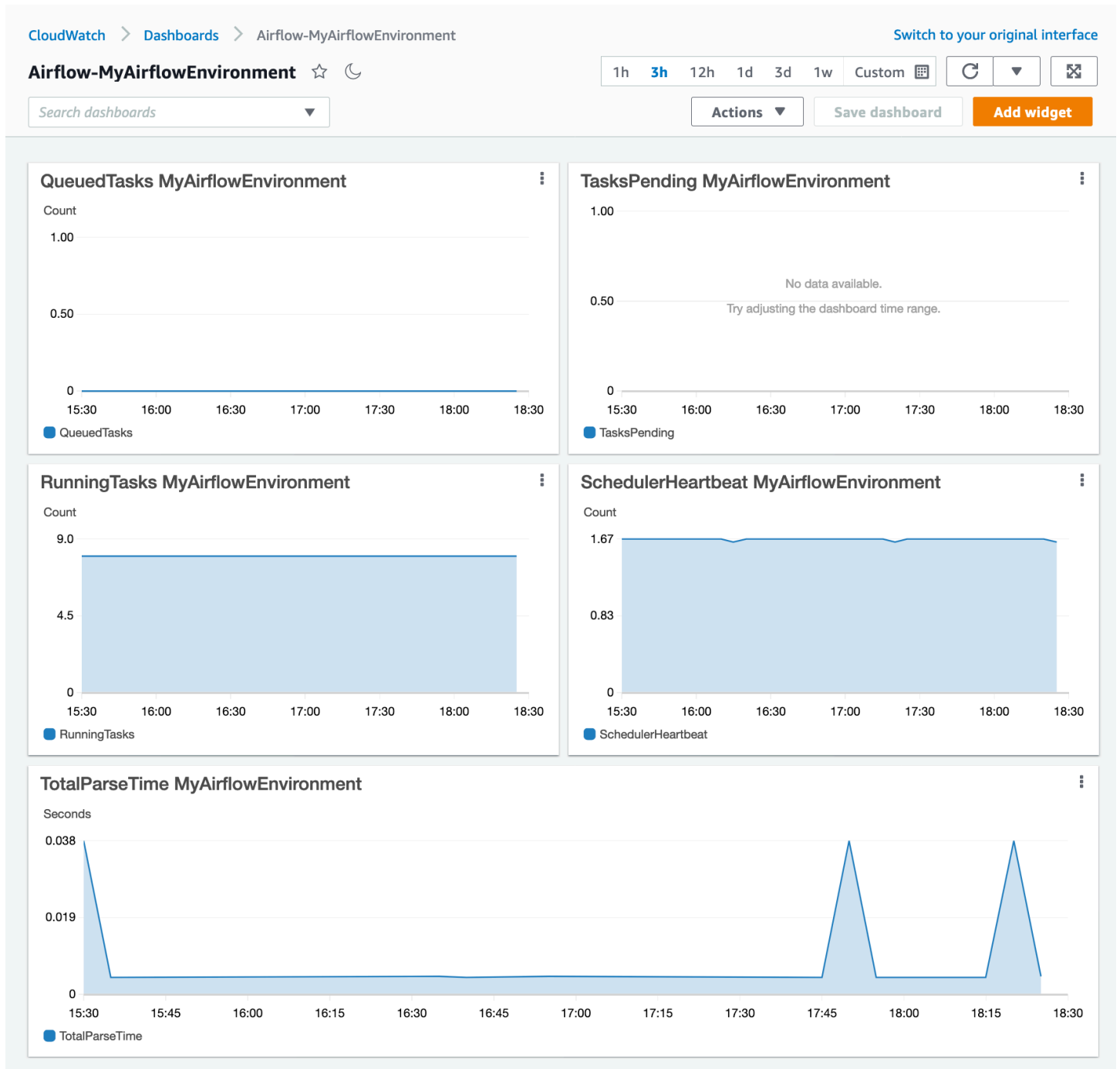
**Note**

Gilt nicht für Apache Airflow v2.2 und höher.

- **RunningTasks**- Die Anzahl der Aufgaben, die im Executor ausgeführt werden. Entspricht der `executor.running_tasks` Apache Airflow-Metrik.
- **SchedulerHeartbeat**— Die Anzahl der Check-ins, die Apache Airflow für den Scheduler-Job durchführt. Entspricht den `scheduler_heartbeat` Apache Airflow-Metriken.
- **TotalParseZeit** — Die Anzahl der Sekunden, die benötigt wurden, um alle DAG-Dateien einmal zu scannen und zu importieren. Entspricht der `dag_processing.total_parse_time` Apache Airflow-Metrik.

## Über das Armaturenbrett

Die folgende Abbildung zeigt das Monitoring-Dashboard, das mit dem Tutorial und der Vorlagendefinition in diesem Abschnitt erstellt wurde.



## Mithilfe von Tutorials AWS

Sie können das folgende AWS Tutorial verwenden, um automatisch ein Integritätsstatus-Dashboard für alle Amazon MWAA-Umgebungen zu erstellen, die derzeit bereitgestellt werden. Außerdem werden in allen Amazon CloudWatch MWAA-Umgebungen Alarme für kranke Mitarbeiter und für Heartbeat-Ausfälle im Terminplaner generiert.



- [CloudWatch Dashboard-Automatisierung für Amazon MWA](#)

## Verwenden AWS CloudFormation

Sie können die AWS CloudFormation Vorlagendefinition in diesem Abschnitt verwenden, um ein Überwachungs-Dashboard zu erstellen und dann Alarme auf der CloudWatch Konsole hinzuzufügen CloudWatch, um Benachrichtigungen zu erhalten, wenn eine Metrik einen bestimmten Schwellenwert überschreitet. Informationen zum Erstellen des Stacks mithilfe dieser Vorlagendefinition finden Sie unter [Einen Stack auf der AWS CloudFormation Konsole](#) erstellen. Informationen zum Hinzufügen eines Alarms zum Dashboard finden Sie unter [Alarme verwenden](#).

```
AWSTemplateFormatVersion: "2010-09-09"
Description: Creates MWA Cloudwatch Dashboard
Parameters:
 DashboardName:
 Description: Enter the name of the CloudWatch Dashboard
 Type: String
 EnvironmentName:
 Description: Enter the name of the MWA Environment
 Type: String
Resources:
 BasicDashboard:
 Type: AWS::CloudWatch::Dashboard
 Properties:
 DashboardName: !Ref DashboardName
 DashboardBody:
 Fn::Sub: '{
 "widgets": [
 {
 "type": "metric",
 "x": 0,
 "y": 0,
 "width": 12,
 "height": 6,
 "properties": {
 "view": "timeSeries",
 "stacked": true,
 "metrics": [
 [
 "AmazonMWA",
 "QueuedTasks",
 "Function",
```

```
 "Executor",
 "Environment",
 "${EnvironmentName}"
]
],
"region": "${AWS::Region}",
"title": "QueuedTasks ${EnvironmentName}",
"period": 300
}
},
{
 "type": "metric",
 "x": 0,
 "y": 6,
 "width": 12,
 "height": 6,
 "properties": {
 "view": "timeSeries",
 "stacked": true,
 "metrics": [
 [
 "AmazonMWAA",
 "RunningTasks",
 "Function",
 "Executor",
 "Environment",
 "${EnvironmentName}"
]
],
 "region": "${AWS::Region}",
 "title": "RunningTasks ${EnvironmentName}",
 "period": 300
 }
},
{
 "type": "metric",
 "x": 12,
 "y": 6,
 "width": 12,
 "height": 6,
 "properties": {
 "view": "timeSeries",
 "stacked": true,
 "metrics": [
```

```
 [
 "AmazonMWAA",
 "SchedulerHeartbeat",
 "Function",
 "Scheduler",
 "Environment",
 "${EnvironmentName}"
]
],
 "region": "${AWS::Region}",
 "title": "SchedulerHeartbeat ${EnvironmentName}",
 "period": 300
}
},
{
 "type": "metric",
 "x": 12,
 "y": 0,
 "width": 12,
 "height": 6,
 "properties": {
 "view": "timeSeries",
 "stacked": true,
 "metrics": [
 [
 "AmazonMWAA",
 "TasksPending",
 "Function",
 "Scheduler",
 "Environment",
 "${EnvironmentName}"
]
],
 "region": "${AWS::Region}",
 "title": "TasksPending ${EnvironmentName}",
 "period": 300
 }
},
{
 "type": "metric",
 "x": 0,
 "y": 12,
 "width": 24,
 "height": 6,
```

```
 "properties": {
 "view": "timeSeries",
 "stacked": true,
 "region": "${AWS::Region}",
 "metrics": [
 [
 "AmazonMWA",
 "TotalParseTime",
 "Function",
 "DAG Processing",
 "Environment",
 "${EnvironmentName}"
]
],
 "title": "TotalParseTime ${EnvironmentName}",
 "period": 300
 }
]
}'
```

## Löschen von Metriken und Dashboards

Wenn Sie eine Amazon MWAA-Umgebung löschen, wird auch das entsprechende Dashboard gelöscht. CloudWatch Metriken werden für fünfzehn (15) Monate gespeichert und können nicht gelöscht werden. Die CloudWatch Konsole beschränkt die Suche nach Metriken auf zwei (2) Wochen nach der letzten Erfassung einer Metrik, um sicherzustellen, dass die aktuellsten Instances für Ihre Amazon MWAA-Umgebung angezeigt werden. Weitere Informationen finden Sie in den [CloudWatch häufig gestellten Fragen zu Amazon](#).

## Als nächstes

- Erfahren Sie, wie Sie eine DAG erstellen, die die Amazon Aurora PostgreSQL-Metadatenbank für Ihre Umgebung abfragt und benutzerdefinierte Metriken veröffentlicht. CloudWatch [Verwenden einer DAG zum Schreiben benutzerdefinierter Metriken](#)

## Apache Airflow v2-Umgebungsmetriken in CloudWatch

Apache Airflow v2 ist bereits dafür eingerichtet, [StatsD-Metriken](#) für eine Amazon Managed Workflows for Apache Airflow-Umgebung zu sammeln und an Amazon zu senden. CloudWatch

Die vollständige Liste der von Apache Airflow gesendeten Metriken ist auf der Seite [Metriken](#) im Apache Airflow-Referenzhandbuch verfügbar. Auf dieser Seite werden die Apache Airflow-Metriken beschrieben, die in der Konsole verfügbar sind CloudWatch, und wie Sie auf Metriken in der Konsole zugreifen können. CloudWatch

## Inhalt

- [Bedingungen](#)
- [Dimensionen](#)
- [Zugreifen auf Metriken in der Konsole CloudWatch](#)
- [Apache Airflow-Metriken sind verfügbar in CloudWatch](#)
  - [Apache Airflow-Zähler](#)
  - [Apache Luftstrommessgeräte](#)
  - [Apache Airflow Timer](#)
- [Auswahl der Metriken, die gemeldet werden](#)
- [Als nächstes](#)

## Bedingungen

### Namespace

Ein Namespace ist ein Container für die CloudWatch Metriken eines AWS Dienstes. Für Amazon MWAA lautet der Namespace AmazonMWAA.

### CloudWatch Metriken

Eine CloudWatch Metrik stellt einen nach der Zeit geordneten Satz von Datenpunkten dar, die spezifisch für sind. CloudWatch

### Apache Airflow-Metriken

Die spezifischen [Metriken](#) für Apache Airflow.

### Dimension

Eine Dimension ist ein Name-Wert-Paar, das zur Identifizierung einer Metrik beiträgt.

## Einheit

Eine Statistik hat eine Maßeinheit. Für Amazon MWAA umfassen die Einheiten Anzahl, Sekunden und Millisekunden. Für Amazon MWAA werden die Einheiten auf der Grundlage der Einheiten in den ursprünglichen Airflow-Metriken festgelegt.

## Dimensionen

In diesem Abschnitt wird die Gruppierung von CloudWatch Dimensionen für Apache Airflow-Metriken unter beschrieben. CloudWatch

| Dimension     | Beschreibung                                                                                  |  |  |  |
|---------------|-----------------------------------------------------------------------------------------------|--|--|--|
| DAG           | Zeigt einen bestimmte n Apache Airflow DAG-Namen an.                                          |  |  |  |
| DAG-Dateiname | Zeigt einen bestimmte n Apache Airflow DAG-Dateinamen an.                                     |  |  |  |
| Funktion      | Diese Dimension wird verwendet, um die Gruppierung von Metriken in zu verbessern. CloudWatch  |  |  |  |
| Aufgabe       | Zeigt einen Apache Airflow-Job an, der vom Scheduler ausgeführt wird. Hat immer den Wert Job. |  |  |  |
| Operator      | Zeigt einen bestimmten Apache Airflow-Operator an.                                            |  |  |  |

| Dimension | Beschreibung                                                                 |  |  |  |
|-----------|------------------------------------------------------------------------------|--|--|--|
| Pool      | Weist auf einen bestimmten Apache Airflow-Workerpool hin.                    |  |  |  |
| Aufgabe   | Weist auf eine bestimmte Apache Airflow-Aufgabe hin.                         |  |  |  |
| HostName  | Gibt den Hostnamen für einen bestimmten laufenden Apache Airflow-Prozess an. |  |  |  |

## Zugreifen auf Metriken in der Konsole CloudWatch

In diesem Abschnitt wird beschrieben, wie Sie auf Leistungsmetriken CloudWatch für eine bestimmte DAG zugreifen.

So zeigen Sie Leistungskennzahlen für eine Dimension an




1. Öffnen Sie die [Seite „Metriken“](#) in der CloudWatch Konsole.
2. Verwenden Sie die AWS Regionsauswahl, um Ihre Region auszuwählen.
3. Wählen Sie den AmazonMWAA-Namespace.
4. Wählen Sie auf der Registerkarte Alle Metriken eine Dimension aus. Zum Beispiel DAG, Umgebung.
5. Wählen Sie eine CloudWatch Metrik für eine Dimension. Zum Beispiel TaskInstanceErfolge oder TaskInstanceDauer. Wählen Sie Alle Suchergebnisse grafisch darstellen aus.
6. Wählen Sie die Registerkarte Graphische Metriken, um Leistungsstatistiken für Apache Airflow-Metriken wie DAG, Umgebung und Aufgabe anzuzeigen.

## Apache Airflow-Metriken sind verfügbar in CloudWatch




In diesem Abschnitt werden die Apache Airflow-Metriken und -Dimensionen beschrieben, die an gesendet werden. CloudWatch

## Apache Airflow-Zähler



Die Apache Airflow-Metriken in diesem Abschnitt enthalten Daten zu [Apache](#) Airflow Counters.

| CloudWatch metrisch                                                                                                                                                                                                                                                      | Apache Airflow-Metrik             | Einheit | Dimension              |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|---------|------------------------|
| SLA-Meldung verpasst                                                                                                                                                                                                                                                     | sla_missed                        | Anzahl  | Funktion,<br>Scheduler |
| <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; background-color: #E6F2FF;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.4.3 und höher.</p> </div>   |                                   |         |                        |
| SLA-Callback ist fehlgeschlagen                                                                                                                                                                                                                                          | sla_callback_notification_failure | Anzahl  | Funktion,<br>Scheduler |
| <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; background-color: #E6F2FF;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.4.3 und höher.</p> </div> |                                   |         |                        |
| Aktualisierungen                                                                                                                                                                                                                                                         | dataset.updates                   | Anzahl  | Funktion,<br>Scheduler |
| <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; background-color: #E6F2FF;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.6.3 und höher.</p> </div> |                                   |         |                        |
| Verwaist                                                                                                                                                                                                                                                                 | dataset.verwaist                  | Anzahl  | Funktion,<br>Scheduler |








| CloudWatch metrisch                                                                                                                                       | Apache Airflow-Metrik                       | Einheit | Dimension                  |  |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------|---------|----------------------------|--|
| <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.6.3 und höher.</p>   |                                             |         |                            |  |
| FailedCeleryTaskExecution                                                                                                                                 | celery.execute_command.failure              | Anzahl  | Funktion, Sellerie         |  |
| <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.4.3 und höher.</p>   |                                             |         |                            |  |
| FilePathQueueUpdateZählen                                                                                                                                 | dag_processing.file_path_queue_update_count | Anzahl  | Funktion, Scheduler        |  |
| <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.6.3 und höher.</p> |                                             |         |                            |  |
| CriticalSectionBeschäftigt                                                                                                                                | scheduler.critical_section_busy             | Anzahl  | Funktion, Scheduler        |  |
| DagBagGröße                                                                                                                                               | dagbag_size                                 | Anzahl  | Funktion, DAG-Verarbeitung |  |
| DagCallbackAusnahmen                                                                                                                                      | dag.callback_exceptions                     | Anzahl  | DAG, Alle                  |  |

| CloudWatch metrisch               | Apache Airflow-Metrik                              | Einheit | Dimension                                                         |
|-----------------------------------|----------------------------------------------------|---------|-------------------------------------------------------------------|
| SLA ist gescheitert EmailAttempts | sla_email_notification_failed                      | Anzahl  | Funktion, Scheduler                                               |
| TaskInstanceFertig                | bis fertig. {tag_id}. {Aufgaben-ID}. {Bundesstaat} | Anzahl  | TAG, {tag_id}<br>Aufgabe, {task_id}<br>Bundesstaat, {Bundesstaat} |
| JobEnd                            | {Jobname}_Ende                                     | Anzahl  | Job, {job_name}                                                   |
| JobHeartbeatFehlschlag            | {job_name}_heartbeat_failure                       | Anzahl  | Job, {job_name}                                                   |
| JobStart                          | {Jobname}_Start                                    | Anzahl  | Job, {job_name}                                                   |
| ManagerStalls                     | dag_processing.manager_stalls                      | Anzahl  | Funktion, DAG-Verarbeitung                                        |
| OperatorFailures                  | operator_failures_{operator_name}                  | Anzahl  | Betreiber, {operator_name}                                        |
| OperatorSuccesses                 | operator_successes_{operator_name}                 | Anzahl  | Betreiber, {operator_name}                                        |

| CloudWatch metrisch                                                                                                                                                                                                                                                      | Apache Airflow-Metrik                | Einheit | Dimension                  |  |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------|---------|----------------------------|--|
| OtherCallbackZählen                                                                                                                                                                                                                                                      | dag_processing.other_callback_count  | Anzahl  | Funktion, Scheduler        |  |
| <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; background-color: #E6F2FF;"> <p> <b>Note</b><br/>Verfügbar in Apache Airflow v2.6.3 und höher.</p> </div>    |                                      |         |                            |  |
| Prozesse                                                                                                                                                                                                                                                                 | dag_processing.processes             | Anzahl  | Funktion, DAG-Verarbeitung |  |
| SchedulerHeartbeat                                                                                                                                                                                                                                                       | scheduler_heartbeat                  | Anzahl  | Funktion, Scheduler        |  |
| StartedTaskInstanzen                                                                                                                                                                                                                                                     | zum Starten. {tag_id}. {Aufgaben-ID} | Anzahl  | DAG, Alle Aufgabe, Alle    |  |
| SlaCallbackZählen                                                                                                                                                                                                                                                        | dag_processing.sla_callback_count    | Anzahl  | Funktion, Scheduler        |  |
| <div style="border: 1px solid #0070C0; border-radius: 10px; padding: 10px; background-color: #E6F2FF;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.6.3 und höher.</p> </div> |                                      |         |                            |  |

| CloudWatch metrisch               | Apache Airflow-Metrik                     | Einheit | Dimension                      |
|-----------------------------------|-------------------------------------------|---------|--------------------------------|
| TasksKilledExtern                 | scheduler<br>.tasks.killed_external       | Anzahl  | Funktion,<br>Scheduler         |
| TaskTimeoutFehler                 | celery.task_timeout_error                 | Anzahl  | Funktion,<br>Sellerie          |
| TaskInstanceCreatedUsingBetreiber | task_instance_created-<br>{Operator name} | Anzahl  | Betreiber<br>, {operator_name} |
| TaskInstancePreviouslySucceeded   | vorherig_erfolgreich                      | Anzahl  | DAG, Alle<br>Aufgabe, Alle     |
| TaskInstanceMisserfolge           | ti_failures                               | Anzahl  | DAG, Alle<br>Aufgabe, Alle     |
| TaskInstanceErfolge               | ti_successs                               | Anzahl  | DAG, Alle<br>Aufgabe, Alle     |
| TaskRemovedVon Dag                | task_from_dag entfernt.<br>{tag_id}       | Anzahl  | TAG,<br>{dag_id}               |
| TaskRestoredZu DAG                | task_restored_to_dag.<br>{tag_id}         | Anzahl  | TAG,<br>{dag_id}               |

| CloudWatch metrisch                                                                                                                                                                                                                                                                                       | Apache Airflow-Metrik        | Einheit | Dimension           |  |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------|---------|---------------------|--|
| TriggersSucceeded<br><br><div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.7.2 und höher.</p> </div>           | triggers.erfolgreich         | Anzahl  | Funktion, Auslöser  |  |
| TriggersFailed<br><br><div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.7.2 und höher.</p> </div>              | triggers.failed              | Anzahl  | Funktion, Auslöser  |  |
| TriggersBlockedMainThread<br><br><div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.7.2 und höher.</p> </div> | triggers.blocked_main_thread | Anzahl  | Funktion, Auslöser  |  |
| TriggerHeartbeat<br><br><div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.8.1 und höher.</p> </div>          | triggerer_heartbeat          | Anzahl  | Funktion, Triggerer |  |


| CloudWatch metrisch                                                                                                                                                                                                                                         | Apache Airflow-Metrik                         | Einheit | Dimension                  |  |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------|---------|----------------------------|--|
| TaskInstanceCreatedUsingBetreiber                                                                                                                                                                                                                           | airflow.task_instance_created_{operator_name} | Anzahl  | Betreiber, {operator_name} |  |
| <div style="border: 1px solid #add8e6; border-radius: 10px; padding: 10px; margin: 10px 0;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.7.2 und höher.</p> </div> |                                               |         |                            |  |
| ZombiesKilled                                                                                                                                                                                                                                               | Zombies_getötet                               | Anzahl  | DAG, Alle Aufgabe, Alle    |  |



## Apache Luftstrommessgeräte

Die Apache Airflow-Metriken in diesem Abschnitt enthalten Daten zu [Apache Airflow Gauges](#).

| CloudWatch metrisch    | Apache Airflow-Metrik        | Einheit | Dimension                  |  |
|------------------------|------------------------------|---------|----------------------------|--|
| DAG-Fehler FileRefresh | dag_file_refresh_error       | Anzahl  | Funktion, DAG-Verarbeitung |  |
| ImportErrors           | dag_processing.import_errors | Anzahl  | Funktion, DAG-Verarbeitung |  |

| CloudWatch metrisch | Apache Airflow-Metrik                    | Einheit  | Dimension                               |  |
|---------------------|------------------------------------------|----------|-----------------------------------------|--|
| Exception Failures  | smart_sensor.operator.exception_failures | Anzahl   | Funktion, intelligenter Sensor-Operator |  |
| ExecutedTasks       | smart_sensor.operator.executed_tasks     | Anzahl   | Funktion, intelligenter Sensor-Operator |  |
| InfraFailures       | smart_sensor.operator.infra_failures     | Anzahl   | Funktion, intelligenter Sensor-Operator |  |
| LoadedTasks         | smart_sensor.operator.loaded_tasks       | Anzahl   | Funktion, intelligenter Sensoroperator  |  |
| TotalParseZeit      | dag_processing.total_parse_time          | Sekunden | Funktion, DAG-Verarbeitung              |  |
| Triggered DagLäufe  | dataset.triggered_dagruns                | Anzahl   | Funktion, Scheduler                     |  |

 **Note**  
Verfügbar in Apache Airflow v2.6.3 und höher.

| CloudWatch metrisch                                                                                                                                                                                                                                                                                    | Apache Airflow-Metrik                                   | Einheit  | Dimension                                               |  |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------|----------|---------------------------------------------------------|--|
| TriggersRunning<br><br><div style="border: 1px solid #007bff; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> <b>Note</b><br/>Verfügbar in Apache Airflow v2.7.2 und höher.</p> </div>           | löst aus. Wird ausgeführt.<br><i>{Hostname}</i>         | Anzahl   | Funktion, Auslöser<br><br>HostName, <i>{Hostname}</i> } |  |
| PoolDeferredSteckplätze<br><br><div style="border: 1px solid #007bff; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> <b>Note</b><br/>Verfügbar in Apache Airflow v2.7.2 und höher.</p> </div> | pool.deferred_slots.<br>{pool_name}                     | Anzahl   | Schwimmbad, {Poolname}                                  |  |
| TAG FileProcessing LastRun SecondsAgo                                                                                                                                                                                                                                                                  | dag_processing.last_run.seconds_ago.<br>{dag_Dateiname} | Sekunden | DAG-Dateiname, {dag_filename}                           |  |
| OpenSlots                                                                                                                                                                                                                                                                                              | executor.open_slots                                     | Anzahl   | Funktion, Executor                                      |  |




| CloudWatch metrisch     | Apache Airflow-Metrik                  | Einheit | Dimension                                     |  |
|-------------------------|----------------------------------------|---------|-----------------------------------------------|--|
| OrphanedTasksAdoptiert  | scheduler<br>.orphaned_tasks.adoptiert | Anzahl  | Funktion,<br>Scheduler                        |  |
| OrphanedTasksGelöscht   | scheduler<br>.orphaned_tasks.clear     | Anzahl  | Funktion,<br>Scheduler                        |  |
| PokedExceptions         | smart_sensor_operator.poked_exception  | Anzahl  | Funktion,<br>intelligenter<br>Sensor-Operator |  |
| PokedSuccess            | smart_sensor_operator.poked_success    | Anzahl  | Funktion,<br>intelligenter<br>Sensoroperator  |  |
| PokedTasks              | smart_sensor_operator.poked_tasks      | Anzahl  | Funktion,<br>intelligenter<br>Sensor-Operator |  |
| PoolFailures            | pool.open_slots.<br>{Poolname}         | Anzahl  | Schwimmbad,<br>{Poolname}                     |  |
| PoolStarvingAufgaben    | pool.starving_tasks.<br>{Poolname}     | Anzahl  | Schwimmbad,<br>{Poolname}                     |  |
| PoolOpenSpielautomaten  | pool.open_slots.<br>{Poolname}         | Anzahl  | Schwimmbad,<br>{Poolname}                     |  |
| PoolQueueSpielautomaten | pool.queued_slots.<br>{Poolname}       | Anzahl  | Schwimmbad,<br>{Poolname}                     |  |

| CloudWatch metrisch                                                                                                                                                                                                                                                           | Apache Airflow-Metrik             | Einheit | Dimension                  |  |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------|---------|----------------------------|--|
| PoolRunningSpielautomaten                                                                                                                                                                                                                                                     | pool.running_slots.<br>{Poolname} | Anzahl  | Schwimmbad,<br>{Poolname}  |  |
| Processor Timeouts                                                                                                                                                                                                                                                            | dag_processing.processor_timeouts | Anzahl  | Funktion, DAG-Verarbeitung |  |
| QueuedTasks                                                                                                                                                                                                                                                                   | executor.queued_tasks             | Anzahl  | Funktion, Executor         |  |
| RunningTasks                                                                                                                                                                                                                                                                  | executor.running_tasks            | Anzahl  | Funktion, Executor         |  |
| TasksExecutable                                                                                                                                                                                                                                                               | scheduler.tasks.executable        | Anzahl  | Funktion, Scheduler        |  |
| TasksPending                                                                                                                                                                                                                                                                  | scheduler.tasks.pending           | Anzahl  | Funktion, Scheduler        |  |
| <div data-bbox="113 1134 365 1543" style="border: 1px solid #00a0e3; border-radius: 10px; padding: 10px;"> <p> <b>Note</b></p> <p>Gilt nicht für Apache Airflow v2.2 und höher.</p> </div> |                                   |         |                            |  |
| TasksRunning                                                                                                                                                                                                                                                                  | scheduler.tasks.running           | Anzahl  | Funktion, Scheduler        |  |
| TasksStarving                                                                                                                                                                                                                                                                 | scheduler.tasks.starving          | Anzahl  | Funktion, Scheduler        |  |



| CloudWatch metrisch | Apache Airflow-Metrik              | Einheit | Dimension              |
|---------------------|------------------------------------|---------|------------------------|
| TasksWithoutDagRun  | scheduler<br>.tasks.without_dagrun | Anzahl  | Funktion,<br>Scheduler |

## Apache Airflow Timer

Die Apache Airflow-Metriken in diesem Abschnitt enthalten Daten über [Apache](#) Airflow Timer.

| CloudWatch metrisch                                                                                                                                                                                                                                                      | Apache Airflow-Metrik                         | Einheit       | Dimension                  |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------|---------------|----------------------------|
| DBDAGs sammeln                                                                                                                                                                                                                                                           | collect_db_dags                               | Millisekunden | Funktion, DAG-Verarbeitung |
| CriticalSectionDauer                                                                                                                                                                                                                                                     | scheduler<br>.critical_section_duration       | Millisekunden | Funktion,<br>Scheduler     |
| CriticalSectionQueryDuration                                                                                                                                                                                                                                             | scheduler<br>.critical_section_query_duration | Millisekunden | Funktion,<br>Scheduler     |
| <div style="border: 1px solid #00a0e3; border-radius: 10px; padding: 10px; background-color: #e1f5fe;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.5.1 und höher.</p> </div> |                                               |               |                            |
| DAG DependencyCheck                                                                                                                                                                                                                                                      | dagrun.<br>Abhängigk                          | Millisekunden | TAG, {dag_id}              |

| CloudWatch metrisch             | Apache Airflow-Metrik                              | Einheit       | Dimension                        |  |
|---------------------------------|----------------------------------------------------|---------------|----------------------------------|--|
|                                 | tagsprüfung.<br>{tag_id}                           |               |                                  |  |
| TAG DurationFailed              | dagrun.duration.ist<br>fehlgeschlagen.<br>{dag_id} | Millisekunden | TAG, {dag_id}                    |  |
| TAG DurationSuccess             | Tag Lauf. Dauer.<br>Erfolg. {tag_id}               | Millisekunden | TAG, {dag_id}                    |  |
| TAG FileProcessing LastDuration | dag_processing.last_duration.<br>{dag_Dateiname}   | Sekunden      | DAG-Dateiname,<br>{dag_filename} |  |
| DAG ScheduledDelay              | dagrun.schedule_delay.<br>{tag_id}                 | Millisekunden | TAG, {dag_id}                    |  |
| FirstTask SchedulingDelay       | Dagrun.<br>{tag_id}.first_task_scheduling_delay    | Millisekunden | TAG, {dag_id}                    |  |

| CloudWatch metrisch                                                                                                                                                                                                                                                      | Apache Airflow-Metrik                                       | Einheit       | Dimension                                  |  |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------|---------------|--------------------------------------------|--|
| Scheduler LoopDauer                                                                                                                                                                                                                                                      | scheduler<br>.schedule<br>r_loop_duration                   | Millisekunden | Funktion,<br>Scheduler                     |  |
| <div style="border: 1px solid #007bff; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.5.1 und höher.</p> </div>   |                                                             |               |                                            |  |
| TaskInstanceDauer                                                                                                                                                                                                                                                        | Tag. {tag_id}.<br>{task_id} .dauer                          | Millisekunden | Tag, {tag_id}<br><br>Aufgabe,<br>{task_id} |  |
| TaskInstanceQueued Duration                                                                                                                                                                                                                                              | Tag. {dag_id}.<br>{task_id}<br>. Dauer der<br>Warteschlange | Millisekunden | DAG, {dag_id}<br><br>Aufgabe,<br>{task_id} |  |
| <div style="border: 1px solid #007bff; border-radius: 10px; padding: 10px; background-color: #e6f2ff;"> <p> <b>Note</b><br/>Verfügbar für Apache Airflow v2.7.2 und höher.</p> </div> |                                                             |               |                                            |  |

| CloudWatch metrisch           | Apache Airflow-Metrik                             | Einheit       | Dimension                                  |
|-------------------------------|---------------------------------------------------|---------------|--------------------------------------------|
| TaskInstanceScheduledDuration | Tag. {dag_id}.<br>{task_id}<br>.geplante<br>Dauer | Millisekunden | Tag, {tag_id}<br><br>Aufgabe,<br>{task_id} |

**Note**  
Verfügbar für Apache Airflow v2.7.2 und höher.

## Auswahl der Metriken, die gemeldet werden

Mithilfe der folgenden Amazon MWAA-Konfigurationsoptionen können Sie wählen CloudWatch, welche Apache Airflow-Metriken an Apache Airflow gesendet oder von Apache Airflow blockiert werden:

- **metrics.metrics\_allow\_list**— Eine Liste von kommagetrennten Präfixen, anhand derer Sie auswählen können, an welche Metriken von Ihrer Umgebung ausgegeben werden. CloudWatch Verwenden Sie diese Option, wenn Sie möchten, dass Apache Airflow nicht alle verfügbaren Metriken sendet und stattdessen eine Teilmenge von Elementen auswählt. z. B. scheduler, executor, dagrun.
- **metrics.metrics\_block\_list**— Eine Liste von kommagetrennten Präfixen, um Metriken herauszufiltern, die mit den Elementen der Liste beginnen. z. B. scheduler, executor, dagrun.

Wenn Sie sowohl als auch metrics.metrics\_allow\_list konfigurieren, ignoriert Apache metrics.metrics\_block\_list Airflow. metrics.metrics\_block\_list Wenn Sie konfigurieren, metrics.metrics\_block\_list aber nicht metrics.metrics\_allow\_list, filtert Apache Airflow die Elemente heraus, die Sie angeben. metrics.metrics\_block\_list

**Note**

Die Optionen `metrics.metrics_allow_list` und die `metrics.metrics_block_list` Konfigurationsoptionen gelten nur für Apache Airflow v2.6.3 und höher. Verwenden Sie für frühere Versionen von Apache Airflow stattdessen `metrics.statsd_allow_list` und `metrics.statsd_block_list`.

## Als nächstes

- Erkunden Sie den Amazon MWAA-API-Betrieb, der zur Veröffentlichung von Umweltgesundheitsmetriken verwendet wird, unter [PublishMetrics](#)

## Container-, Warteschlangen- und Datenbankmetriken für Amazon MWAA

Zusätzlich zu den Apache Airflow-Metriken können Sie die zugrunde liegenden Komponenten Ihrer Amazon Managed Workflows für Apache Airflow-Umgebungen mithilfe CloudWatch von Amazon überwachen. Dabei werden Rohdaten gesammelt und Daten zu lesbaren, nahezu in Echtzeit verfügbaren Metriken verarbeitet. Mit diesen Umgebungsmetriken erhalten Sie einen besseren Einblick in wichtige Leistungsindikatoren, sodass Sie Ihre Umgebungen entsprechend dimensionieren und Probleme mit Ihren Workflows beheben können. Diese Metriken gelten für alle unterstützten Apache Airflow-Versionen auf Amazon MWAA.

Amazon MWAA stellt die CPU- und Speicherauslastung für jeden Amazon Elastic Container Service (Amazon ECS) -Container und jede Amazon Aurora PostgreSQL-Instance sowie Amazon Simple Queue Service (Amazon SQS) -Metriken für die Anzahl der Nachrichten und das Alter der ältesten Nachricht, Amazon Relational Database Service (Amazon RDS) -Metriken für Datenbankverbindungen, Festplattenwarteschlangentiefe, Schreibvorgänge, Latenz und Durchsatz sowie Amazon RDS Proxy-Metriken bereit. Zu diesen Metriken gehören auch die Anzahl der Basisarbeiter, zusätzlichen Worker, Scheduler und Webserver.

Diese Statistiken werden 15 Monate lang aufbewahrt, sodass Sie auf historische Informationen zugreifen und einen besseren Überblick darüber erhalten, warum ein Zeitplan ausfällt, und die zugrunde liegenden Probleme beheben können. Sie können auch Alarme einrichten, die auf bestimmte Grenzwerte achten und Benachrichtigungen senden oder Aktivitäten auslösen, wenn

diese Grenzwerte erreicht werden. Weitere Informationen finden Sie im [CloudWatch Amazon-Benutzerhandbuch](#).

## Themen

- [Bedingungen](#)
- [Dimensionen](#)
- [Zugreifen auf Metriken in der Konsole CloudWatch](#)
- [Liste der Metriken](#)

## Bedingungen

### Namespace

Ein Namespace ist ein Container für die CloudWatch Metriken eines AWS Dienstes. Für Amazon MWAAs lautet der Namespace `AWS/MWAA`.

### CloudWatch Metriken

Eine CloudWatch Metrik stellt einen nach der Zeit geordneten Satz von Datenpunkten dar, die spezifisch für sind. CloudWatch

### Dimension

Eine Dimension ist ein Name-Wert-Paar, das zur Identifizierung einer Metrik beiträgt.

### Einheit

Eine Statistik hat eine Maßeinheit. Bei Amazon MWAAs enthalten Einheiten die Anzahl.

## Dimensionen

In diesem Abschnitt wird die Gruppierung der CloudWatch Dimensionen für Amazon MWAAs-Metriken beschrieben. CloudWatch

| Dimension | Beschreibung                                                                                                              |
|-----------|---------------------------------------------------------------------------------------------------------------------------|
| Cluster   | Metriken für die mindestens drei Amazon ECS-Container, die eine Amazon MWAAs-Umgebung zur Ausführung von Apache Airflow-K |



| Dimension     | Beschreibung                                                                                                                                                                                                                                                                                                                                                            |
|---------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|               | omponenten verwendet: Scheduler, Worker und Webserver.                                                                                                                                                                                                                                                                                                                  |
| Warteschlange | Metriken für die Amazon SQS SQS-Warteschlangen, die den Scheduler von den Workern entkoppeln. Wenn Mitarbeiter die Nachrichten lesen, gelten sie als während des Fluges und sind für andere Mitarbeiter nicht verfügbar. Nachrichten können von anderen Mitarbeitern gelesen werden, sofern sie nicht vor Ablauf der Sichtbarkeitsdauer von 12 Stunden gelöscht werden. |
| Datenbank     | Metrikt die von Amazon MWAA verwendet in Aurora-Cluster. Dazu gehören Metriken für die primäre Datenbank-Instance und eine Read Replica zur Unterstützung der Lesevorgänge. Amazon MWAA veröffentlicht Datenbankmetriken für READER- und WRITER-Instances.                                                                                                              |

## Zugreifen auf Metriken in der Konsole CloudWatch

In diesem Abschnitt wird beschrieben, wie Sie in auf Ihre Amazon MWAA-Metriken zugreifen können. CloudWatch

Um Leistungskennzahlen für eine Dimension anzuzeigen

1. Öffnen Sie die [Seite „Metriken“](#) in der CloudWatch Konsole.
2. Verwenden Sie die AWS Regionsauswahl, um Ihre Region auszuwählen.
3. Wählen Sie den AWS/MWAA-Namespace.
4. Wählen Sie auf der Registerkarte Alle Metriken eine Dimension aus. Zum Beispiel Cluster.
5. Wählen Sie eine CloudWatch Metrik für eine Dimension aus. Zum Beispiel NumSchedulersoder CPUUtilization. Wählen Sie dann Alle Suchergebnisse grafisch darstellen aus.
6. Wählen Sie den Tab Graphische Metriken, um die Leistungskennzahlen anzuzeigen.

## Liste der Metriken

In den folgenden Tabellen sind die Cluster-, Warteschlangen- und Datenbankservice-Metriken für Amazon MWAA aufgeführt. Um Beschreibungen für Metriken anzuzeigen, die direkt von Amazon ECS, Amazon SQS oder Amazon RDS ausgegeben werden, wählen Sie den entsprechenden Dokumentationslink.

Themen

- [Cluster-Metriken](#)
- [Datenbankmetriken](#)
- [Warteschlangenmetriken](#)
- [Application-Load-Balancer-Metriken](#)

### Cluster-Metriken

Die folgenden Metriken gelten für jeden Scheduler, Basisworker, zusätzlichen Worker und Webserver. Weitere Informationen und Beschreibungen der einzelnen Cluster-Metriken finden Sie unter [Verfügbare Metriken und Dimensionen](#) im Amazon ECS Developer Guide.

| Namespace | Metrik            | Einheit |
|-----------|-------------------|---------|
| AWS/MWAA  | CPUUtilization    | Prozent |
| AWS/MWAA  | MemoryUtilization | Prozent |

### Bewertung der Anzahl zusätzlicher Worker- und Webserver-Container

Sie können die in der Cluster-Dimension bereitgestellten Komponentenmetriken verwenden, wie im folgenden Verfahren beschrieben, um zu beurteilen, wie viele zusätzliche Worker oder Webserver eine Umgebung zu einem bestimmten Zeitpunkt verwendet. Sie können dies tun, indem Sie entweder die CPU-Auslastung oder die MemoryUtilizationMetrik grafisch darstellen und den Statistiktyp auf Stichprobenanzahl festlegen. Der resultierende Wert ist die Gesamtzahl der RUNNING Aufgaben für die Komponente. `AdditionalWorker` Wenn Sie die Anzahl der zusätzlichen Worker-Instanzen kennen, die von Ihrer Umgebung genutzt werden, können Sie besser einschätzen, wie Ihre Umgebung skaliert, und Sie können die Anzahl der zusätzlichen Worker optimieren.

## Workers

Um die Anzahl der zusätzlichen Mitarbeiter zu ermitteln, die AWS Management Console

1. Wählen Sie den AWS/MWAA-Namespace.
2. Wählen Sie auf der Registerkarte Alle Metriken die Cluster-Dimension aus.
3. Wählen Sie unter der Cluster-Dimension für entweder die AdditionalWorkerCPUUtilization oder die MemoryUtilizationMetrik aus.
4. Stellen Sie auf der Registerkarte Graphische Metriken die Option Zeitraum auf 1 Minute und Statistik auf Stichprobenanzahl ein.

## Web servers

Um die Anzahl der zusätzlichen Webserver mit dem zu ermitteln AWS Management Console

1. Wählen Sie den AWS/MWAA-Namespace.
2. Wählen Sie auf der Registerkarte Alle Metriken die Cluster-Dimension aus.
3. Wählen Sie unter der Cluster-Dimension für entweder die AdditionalWebserverCPUUtilization oder die MemoryUtilizationMetrik aus.
4. Stellen Sie auf der Registerkarte Graphische Metriken die Option Zeitraum auf 1 Minute und Statistik auf Stichprobenanzahl ein.

Weitere Informationen finden Sie unter [Anzahl der RUNNING Serviceaufgaben](#) im Amazon Elastic Container Service Developer Guide.

## Datenbankmetriken

Die folgenden Metriken gelten für jede Datenbank-Instance, die der Amazon MWAA-Umgebung zugeordnet ist.

| Namespace | Metrik              | Einheit |
|-----------|---------------------|---------|
| AWS/MWAA  | CPUUtilization      | Prozent |
| AWS/MWAA  | DatabaseConnections | Anzahl  |
| AWS/MWAA  | DiskQueueDepth      | Anzahl  |

| Namespace | Metrik          | Einheit                 |
|-----------|-----------------|-------------------------|
| AWS/MWAA  | FreeableMemory  | Bytes                   |
| AWS/MWAA  | VolumeWriteIOPS | Zählt alle fünf Minuten |
| AWS/MWAA  | WriteIOPS       | Anzahl pro Sekunde      |
| AWS/MWAA  | WriteLatency    | Sekunden                |
| AWS/MWAA  | WriteThroughput | Bytes pro Sekunde       |

## Warteschlangenmetriken

Weitere Informationen zu Einheiten und Beschreibungen für die folgenden Warteschlangenmetriken finden Sie unter [Verfügbare CloudWatch Metriken für Amazon SQS](#) im Amazon Simple Queue Service Developer Guide.

| Namespace | Metrik                     | Einheit  |
|-----------|----------------------------|----------|
| AWS/MWAA  | ApproximateAgeOfOldestTask | Sekunden |
| AWS/MWAA  | RunningTasks               | Anzahl   |
| AWS/MWAA  | QueuedTasks                | Anzahl   |

## Application-Load-Balancer-Metriken

Die Application Load Balancer Balancer-Metriken gelten für die Webserver, die in Ihrer Umgebung ausgeführt werden. Amazon MWAA verwendet diese Metriken, um Ihre Webserver basierend auf der Menge des Datenverkehrs zu skalieren. Weitere Informationen zu Einheiten und Beschreibungen der folgenden Load Balancer-Metriken finden Sie unter [CloudWatch Metriken für Ihren Application Load Balancer](#) im Application Load Balancers User Guide.

| Namespace | Metrik                | Einheit |
|-----------|-----------------------|---------|
| AWS/MWAA  | ActiveConnectionCount | Anzahl  |

# Sicherheit in Amazon Managed Workflows für Apache Airflow

Cloud-Sicherheit hat höchste AWS Priorität. Als AWS Kunde profitieren Sie von einer Rechenzentrums- und Netzwerkarchitektur, die darauf ausgelegt sind, die Anforderungen der sicherheitssensibelsten Unternehmen zu erfüllen.

Sicherheit ist eine gemeinsame Verantwortung zwischen Ihnen AWS und Ihnen (dem Kunden). Das [Modell der geteilten Verantwortung](#) beschreibt dies als Sicherheit der Cloud selbst und Sicherheit in der Cloud:

- Sicherheit der Cloud — AWS ist verantwortlich für den Schutz der Infrastruktur, die AWS Dienste in der AWS Cloud ausführt. AWS bietet Ihnen auch Dienste, die Sie sicher nutzen können. Externe Prüfer testen und verifizieren regelmäßig die Wirksamkeit unserer Sicherheitsmaßnahmen im Rahmen der [AWS](#). Weitere Informationen zu den Compliance-Programmen, die für Amazon MWAA gelten, finden Sie unter [AWS Services in Scope by Compliance Program AWS](#) Program.
- Sicherheit in der Cloud — Ihre Verantwortung richtet sich nach dem AWS Service, den Sie nutzen. 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.

Diese Dokumentation hilft Ihnen zu verstehen, wie Sie das Modell der gemeinsamen Verantwortung bei der Verwendung von Amazon Managed Workflows für Apache Airflow anwenden können. Es zeigt Ihnen, wie Sie Amazon MWAA konfigurieren, um Ihre Sicherheits- und Compliance-Ziele zu erreichen. Sie erfahren auch, wie Sie andere AWS Dienste nutzen können, mit denen Sie Ihre Amazon MWAA-Ressourcen überwachen und sichern können.

In diesem Abschnitt:

- [Datenschutz in Amazon Managed Workflows für Apache Airflow](#)
- [AWS Identity and Access Management](#)
- [Konformitätsprüfung für Amazon Managed Workflows für Apache Airflow](#)
- [Resilienz in Amazon Managed Workflows für Apache Airflow](#)
- [Infrastruktursicherheit in Amazon MWAA](#)
- [Konfiguration und Schwachstellenanalyse in Amazon MWAA](#)
- [Bewährte Sicherheitsmethoden auf Amazon MWAA](#)

# Datenschutz in Amazon Managed Workflows für Apache Airflow

Das [Modell der AWS gemeinsamen Verantwortung](#) gilt für den Datenschutz in Amazon Managed Workflows for Apache Airflow. Wie in diesem Modell beschrieben, AWS ist verantwortlich für den Schutz der globalen Infrastruktur, auf der AWS Cloud alle Systeme laufen. Sie sind dafür verantwortlich, die Kontrolle über Ihre in dieser Infrastruktur gehosteten Inhalte zu behalten. Dieser Inhalt enthält die Sicherheitskonfigurations- und Verwaltungsaufgaben für die von Ihnen verwendeten AWS-Services. Weitere Informationen zum Datenschutz finden Sie unter [Häufig gestellte Fragen zum Datenschutz](#). Informationen zum Datenschutz in Europa finden Sie im Blog-Beitrag [AWS -Modell der geteilten Verantwortung und in der DSGVO](#) im AWS -Sicherheitsblog.

Aus Datenschutzgründen empfehlen wir, dass Sie Ihre AWS-Konto Anmeldeinformationen schützen und individuelle Benutzerkonten mit AWS Identity and Access Management (IAM) einrichten. So erhält jeder Benutzer nur die Berechtigungen, die zum Durchführen seiner Aufgaben erforderlich sind. Außerdem sollten Sie die Daten mit folgenden Methoden schützen:

- Verwenden Sie für jedes Konto die Multi-Faktor-Authentifizierung (MFA).
- Verwenden Sie SSL/TLS, um mit Ressourcen zu kommunizieren. AWS Wir empfehlen TLS 1.2 oder höher.
- Richten Sie die API und die Protokollierung von Benutzeraktivitäten mit ein. AWS CloudTrail
- Verwenden Sie AWS Verschlüsselungslösungen zusammen mit allen Standardsicherheitskontrollen innerhalb der AWS Dienste.
- Verwenden Sie erweiterte verwaltete Sicherheitsservices wie Amazon Macie, die dabei helfen, in Amazon S3 gespeicherte persönliche Daten zu erkennen und zu sichern.

Wir empfehlen dringend, in Freitextfeldern wie z. B. im Feld Name keine vertraulichen oder sensiblen Informationen wie die E-Mail-Adressen Ihrer Kunden einzugeben. Dies gilt auch, wenn Sie mit Amazon MWAA oder anderen AWS Services über die Konsole AWS CLI, API oder AWS SDKs arbeiten. Alle Daten, die Sie in Tags (Markierungen) oder Freiformfelder eingeben, die für Namen verwendet werden, können für Abrechnungs- oder Diagnoseprotokolle verwendet werden. Wenn Sie eine URL für einen externen Server bereitstellen, empfehlen wir dringend, Sie keine Anmeldeinformationen zur Validierung Ihrer Anforderung an den betreffenden Server in die URL einzuschließen.

## Verschlüsselung auf Amazon MWAA

In den folgenden Themen wird beschrieben, wie Amazon MWAA Ihre Daten im Speicher und bei der Übertragung schützt. Anhand dieser Informationen erfahren Sie, wie Amazon MWAA integriert ist, AWS KMS um Daten im Ruhezustand zu verschlüsseln und wie Daten bei der Übertragung mit dem Transport Layer Security (TLS) -Protokoll verschlüsselt werden.

### Themen

- [Verschlüsselung im Ruhezustand](#)
- [Verschlüsselung während der Übertragung](#)

### Verschlüsselung im Ruhezustand

Bei Amazon MWAA handelt es sich bei Daten im Ruhezustand um Daten, die der Service auf persistenten Medien speichert.

Sie können einen [AWS eigenen Schlüssel](#) für die Verschlüsselung von Daten im Ruhezustand verwenden oder optional einen vom [Kunden verwalteten Schlüssel](#) für zusätzliche Verschlüsselung bereitstellen, wenn Sie eine Umgebung erstellen. Wenn Sie sich für die Verwendung eines vom Kunden verwalteten KMS-Schlüssels entscheiden, muss sich dieser in demselben Konto befinden wie die anderen AWS Ressourcen und Dienste, die Sie in Ihrer Umgebung verwenden.

Um einen vom Kunden verwalteten KMS-Schlüssel zu verwenden, müssen Sie die erforderliche Richtlinienerklärung für den CloudWatch Zugriff auf Ihre Schlüsselrichtlinie beifügen. Wenn Sie einen vom Kunden verwalteten KMS-Schlüssel für Ihre Umgebung verwenden, fügt Amazon MWAA in Ihrem Namen vier [Zuschüsse hinzu](#). Weitere Informationen zu den Zuschüssen, die Amazon MWAA einem vom Kunden verwalteten KMS-Schlüssel zuweist, finden Sie unter [Vom Kunden verwaltete Schlüssel für die Datenverschlüsselung](#).

Wenn Sie keinen vom Kunden verwalteten KMS-Schlüssel angeben, verwendet Amazon MWAA standardmäßig einen AWS eigenen KMS-Schlüssel zum Verschlüsseln und Entschlüsseln Ihrer Daten. Wir empfehlen die Verwendung eines AWS eigenen KMS-Schlüssels zur Verwaltung der Datenverschlüsselung auf Amazon MWAA.



**Note**

Sie zahlen für die Speicherung und Nutzung von AWS eigenen oder vom Kunden verwalteten KMS-Schlüsseln auf Amazon MWAA. Weitere Informationen finden Sie unter [AWS KMS - Preisgestaltung](#).

## Verschlüsselungsartefakte

Sie geben die Verschlüsselungsartefakte an, die für die Verschlüsselung im Ruhezustand verwendet werden, indem Sie bei der Erstellung Ihrer Amazon MWAA-Umgebung einen [AWS eigenen Schlüssel oder einen vom Kunden verwalteten Schlüssel](#) angeben. Amazon MWAA fügt die benötigten [Zuschüsse](#) zu Ihrem angegebenen Schlüssel hinzu.

**Amazon S3** — Amazon S3 S3-Daten werden auf Objektebene mit serverseitiger Verschlüsselung (SSE) verschlüsselt. Die Amazon S3 S3-Verschlüsselung und Entschlüsselung erfolgt im Amazon S3 S3-Bucket, in dem Ihr DAG-Code und die unterstützenden Dateien gespeichert sind. Objekte werden verschlüsselt, wenn sie auf Amazon S3 hochgeladen werden, und entschlüsselt, wenn sie in Ihre Amazon MWAA-Umgebung heruntergeladen werden. Wenn Sie einen vom Kunden verwalteten KMS-Schlüssel verwenden, verwendet Amazon MWAA ihn standardmäßig zum Lesen und Entschlüsseln der Daten in Ihrem Amazon S3 S3-Bucket.

**CloudWatch Protokolle** — Wenn Sie einen AWS eigenen KMS-Schlüssel verwenden, werden die an Logs gesendeten Apache Airflow-Protokolle mit serverseitiger Verschlüsselung (SSE) mit dem eigenen KMS-Schlüssel von CloudWatch Logs verschlüsselt. CloudWatch AWS Wenn Sie einen vom Kunden verwalteten KMS-Schlüssel verwenden, müssen Sie Ihrem [KMS-Schlüssel eine Schlüsselrichtlinie](#) hinzufügen, damit CloudWatch Logs Ihren Schlüssel verwenden kann.

**Amazon SQS** — Amazon MWAA erstellt eine Amazon SQS SQS-Warteschlange für Ihre Umgebung. Amazon MWAA verarbeitet die Verschlüsselung von Daten, die an und aus der Warteschlange übergeben werden, mit serverseitiger Verschlüsselung (SSE) entweder mit einem AWS eigenen KMS-Schlüssel oder einem von Ihnen angegebenen, vom Kunden verwalteten KMS-Schlüssel. Sie müssen Ihrer Ausführungsrolle Amazon SQS SQS-Berechtigungen hinzufügen, unabhängig davon, ob Sie einen AWS eigenen oder einen vom Kunden verwalteten KMS-Schlüssel verwenden.

**Aurora PostgreSQL** — Amazon MWAA erstellt einen PostgreSQL-Cluster für Ihre Umgebung. Aurora PostgreSQL verschlüsselt den Inhalt entweder mit einem AWS eigenen oder einem vom Kunden verwalteten KMS-Schlüssel mithilfe serverseitiger Verschlüsselung (SSE). Wenn Sie einen vom

Kunden verwalteten KMS-Schlüssel verwenden, fügt Amazon RDS dem Schlüssel mindestens zwei Grants hinzu: eine für den Cluster und eine für die Datenbank-Instance. Amazon RDS kann zusätzliche Zuschüsse gewähren, wenn Sie Ihren vom Kunden verwalteten KMS-Schlüssel in mehreren Umgebungen verwenden möchten. Weitere Informationen finden Sie unter [Datenschutz in Amazon RDS](#).

## Verschlüsselung während der Übertragung

Daten, die übertragen werden, werden als Daten bezeichnet, die bei der Übertragung im Netzwerk abgefangen werden können.

Transport Layer Security (TLS) verschlüsselt die Amazon MWAA-Objekte, die zwischen den Apache Airflow-Komponenten Ihrer Umgebung und anderen in Amazon MWAA integrierten AWS Diensten wie Amazon S3 übertragen werden. Weitere Informationen zur Amazon S3 S3-Verschlüsselung finden Sie unter [Schutz von Daten durch Verschlüsselung](#).

## Verwendung von vom Kunden verwalteten Schlüsseln zur Verschlüsselung

Sie können optional einen vom [Kunden verwalteten Schlüssel](#) für die Datenverschlüsselung in Ihrer Umgebung bereitstellen. Sie müssen den vom Kunden verwalteten KMS-Schlüssel in derselben Region wie Ihre Amazon MWAA-Umgebungsinstanz und Ihr Amazon S3 S3-Bucket erstellen, in dem Sie Ressourcen für Ihre Workflows speichern. Wenn sich der von Ihnen angegebene vom Kunden verwaltete KMS-Schlüssel in einem anderen Konto befindet als dem, das Sie zur Konfiguration einer Umgebung verwenden, müssen Sie den Schlüssel mit seinem ARN für den kontoübergreifenden Zugriff angeben. Weitere Informationen zum Erstellen von Schlüsseln finden Sie unter [Creating Keys](#) im AWS Key Management Service Developer Guide.

### Was wird unterstützt

| AWS KMS Funktion                                                         | Unterstützt |  |
|--------------------------------------------------------------------------|-------------|--|
| Eine <a href="#">AWS KMS Schlüssel-ID</a> oder ein <a href="#">ARN</a> . | Ja          |  |
| Ein <a href="#">AWS KMS Schlüsselalias</a> .                             | Nein        |  |
| Ein <a href="#">Schlüssel AWS KMS für mehrere Regionen</a> .             | Nein        |  |

## Zuschüsse für die Verschlüsselung verwenden

In diesem Thema werden die Zuschüsse beschrieben, die Amazon MWAA in Ihrem Namen an einen vom Kunden verwalteten KMS-Schlüssel anhängt, um Ihre Daten zu verschlüsseln und zu entschlüsseln.

### Funktionsweise

[Es gibt zwei ressourcenbasierte Zugriffskontrollmechanismen, die von einem vom Kunden verwalteten KMS-Schlüssel unterstützt werden: AWS KMS eine Schlüsselrichtlinie und eine Gewährung.](#)

Eine Schlüsselrichtlinie wird verwendet, wenn die Berechtigung größtenteils statisch ist und im synchronen Dienstmodus verwendet wird. Eine Erteilung wird verwendet, wenn dynamischere und detailliertere Berechtigungen erforderlich sind, z. B. wenn ein Dienst unterschiedliche Zugriffsberechtigungen für sich selbst oder andere Konten definieren muss.

Amazon MWAA verwendet vier Zuschussrichtlinien und hängt sie an Ihren vom Kunden verwalteten KMS-Schlüssel an. Dies ist auf die detaillierten Berechtigungen zurückzuführen, die für eine Umgebung erforderlich sind, um ruhende Daten aus CloudWatch Logs, Amazon SQS-Warteschlangen, Aurora PostgreSQL-Datenbankdatenbank, Secrets Manager, Amazon S3 S3-Bucket und DynamoDB-Tabellen zu verschlüsseln.

Wenn Sie eine Amazon MWAA-Umgebung erstellen und einen vom Kunden verwalteten KMS-Schlüssel angeben, hängt Amazon MWAA die Gewährungsrichtlinien an Ihren vom Kunden verwalteten KMS-Schlüssel an. Diese Richtlinien ermöglichen es Amazon MWAA, Ihren vom Kunden verwalteten KMS-Schlüssel `airflow.region}.amazonaws.com` zu verwenden, um in Ihrem Namen Ressourcen zu verschlüsseln, die Amazon MWAA gehören.

Amazon MWAA erstellt in Ihrem Namen zusätzliche Zuschüsse und fügt diese an einen bestimmten KMS-Schlüssel an. Dazu gehören Richtlinien zur Einstellung eines Zuschusses, wenn Sie Ihre Umgebung löschen, zur Verwendung Ihres vom Kunden verwalteten KMS-Schlüssels für Client-Side Encryption (CSE) und für die AWS Fargate Ausführungsrolle, die auf Geheimnisse zugreifen muss, die durch Ihren vom Kunden verwalteten Schlüssel in Secrets Manager geschützt sind.

### Richtlinien für Zuschüsse

Amazon MWAA fügt in Ihrem Namen einem vom Kunden verwalteten KMS-Schlüssel die folgenden [ressourcenbasierten Policy-Grants](#) hinzu. Diese Richtlinien ermöglichen es dem Empfänger und dem Auftraggeber (Amazon MWAA), die in der Richtlinie definierten Aktionen durchzuführen.

### Zuschuss 1: Wird zur Erstellung von Ressourcen auf Datenebene verwendet

```
{
 "Name": "mwa-grant-for-env-mgmt-role-environment name",
 "GranteePrincipal": "airflow.region.amazonaws.com",
 "RetiringPrincipal": "airflow.region.amazonaws.com",
 "Operations": [
 "kms:Encrypt",
 "kms:Decrypt",
 "kms:ReEncrypt*",
 "kms:GenerateDataKey*",
 "kms:CreateGrant",
 "kms:DescribeKey",
 "kms:RetireGrant"
]
}
```

### Grant 2: wird für den **ControllerLambdaExecutionRole** Zugriff verwendet

```
{
 "Name": "mwa-grant-for-lambda-exec-environment name",
 "GranteePrincipal": "airflow.region.amazonaws.com",
 "RetiringPrincipal": "airflow.region.amazonaws.com",
 "Operations": [
 "kms:Encrypt",
 "kms:Decrypt",
 "kms:ReEncrypt*",
 "kms:GenerateDataKey*",
 "kms:DescribeKey",
 "kms:RetireGrant"
]
}
```

### Grant 3: wird für den **CfnManagementLambdaExecutionRole** Zugriff verwendet

```
{
 "Name": " mwa-grant-for-cfn-mgmt-environment name",
 "GranteePrincipal": "airflow.region.amazonaws.com",
 "RetiringPrincipal": "airflow.region.amazonaws.com",
 "Operations": [
```

```

 "kms:Encrypt",
 "kms:Decrypt",
 "kms:ReEncrypt*",
 "kms:GenerateDataKey*",
 "kms:DescribeKey"
]
}

```

Grant 4: Wird für die Fargate-Ausführungsrolle verwendet, um auf Backend-Geheimnisse zuzugreifen

```

{
 "Name": "mwa-fargate-access-for-environment name",
 "GranteePrincipal": "airflow.region.amazonaws.com",
 "RetiringPrincipal": "airflow.region.amazonaws.com",
 "Operations": [
 "kms:Encrypt",
 "kms:Decrypt",
 "kms:ReEncrypt*",
 "kms:GenerateDataKey*",
 "kms:DescribeKey",
 "kms:RetireGrant"
]
}

```

## Anhängen wichtiger Richtlinien an einen vom Kunden verwalteten Schlüssel

Wenn Sie Ihren eigenen, vom Kunden verwalteten KMS-Schlüssel mit Amazon MWAA verwenden möchten, müssen Sie dem Schlüssel die folgende Richtlinie beifügen, damit Amazon MWAA ihn zur Verschlüsselung Ihrer Daten verwenden kann.

Wenn der vom Kunden verwaltete KMS-Schlüssel, den Sie für Ihre Amazon MWAA-Umgebung verwendet haben, noch nicht für die Verwendung konfiguriert ist CloudWatch, müssen Sie die [Schlüsselrichtlinie](#) aktualisieren, um verschlüsselte CloudWatch Protokolle zuzulassen. Weitere Informationen finden Sie unter dem Dienst „[Protokolldaten verschlüsseln](#)“. CloudWatch AWS Key Management Service

Das folgende Beispiel stellt eine wichtige Richtlinie für CloudWatch Logs dar. Ersetzen Sie die für die Region bereitgestellten Beispielwerte.

```

{

```

```
"Effect": "Allow",
"Principal": {
 "Service": "logs.us-west-2.amazonaws.com"
},
"Action": [
 "kms:Encrypt*",
 "kms:Decrypt*",
 "kms:ReEncrypt*",
 "kms:GenerateDataKey*",
 "kms:Describe*"
],
"Resource": "*",
"Condition": {
 "ArnLike": {
 "kms:EncryptionContext:aws:logs:arn": "arn:aws:logs:us-west-2:*:*"
 }
}
}
```

## AWS Identity and Access Management

AWS Identity and Access Management (IAM) ist ein AWS Service, der einem Administrator hilft, den Zugriff auf Ressourcen sicher zu kontrollieren. AWS IAM-Administratoren kontrollieren, wer authentifiziert (angemeldet) und autorisiert werden kann (über Berechtigungen verfügt), um Amazon Managed Workflows für Apache Airflow-Ressourcen zu verwenden. IAM ist ein AWS Service, den Sie ohne zusätzliche Kosten nutzen können.

Dieses Thema bietet einen grundlegenden Überblick darüber, wie Amazon MWAA AWS Identity and Access Management (IAM) verwendet. Weitere Informationen zur Verwaltung des Zugriffs auf Amazon MWAA finden Sie unter [Verwaltung des Zugriffs auf eine Amazon MWAA-Umgebung](#)

### Inhalt

- [Zielgruppe](#)
- [Authentifizierung mit Identitäten](#)
- [Verwalten des Zugriffs mit Richtlinien](#)
- [Benutzern die Berechtigung zur Anzeige eigener Berechtigungen erteilen](#)
- [Fehlerbehebung bei Amazon Managed Workflows für Identität und Zugriff auf Apache Airflow](#)
- [So funktioniert Amazon MWAA mit IAM](#)

## Zielgruppe

Wie Sie AWS Identity and Access Management (IAM) verwenden, hängt von der Arbeit ab, die Sie in Amazon MWAA ausführen.

**Servicebenutzer** — Wenn Sie den Amazon MWAA-Service für Ihre Arbeit verwenden, stellt Ihnen Ihr Administrator die Anmeldeinformationen und Berechtigungen zur Verfügung, die Sie benötigen. Da Sie für Ihre Arbeit mehr Amazon MWAA-Funktionen verwenden, benötigen Sie möglicherweise zusätzliche Berechtigungen. Wenn Sie die Funktionsweise der Zugriffskontrolle nachvollziehen, wissen Sie bereits, welche Berechtigungen Sie von Ihrem Administrator anfordern müssen. Wenn Sie auf eine Funktion in Amazon MWAA nicht zugreifen können, finden Sie weitere Informationen unter [Fehlerbehebung bei Amazon Managed Workflows für Identität und Zugriff auf Apache Airflow](#)

**Service-Administrator** — Wenn Sie in Ihrem Unternehmen für die Amazon MWAA-Ressourcen verantwortlich sind, haben Sie wahrscheinlich vollen Zugriff auf Amazon MWAA. Es ist Ihre Aufgabe, zu bestimmen, auf welche Funktionen und Ressourcen von Amazon MWAA 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 mit Amazon MWAA verwenden kann, finden Sie unter [So funktioniert Amazon MWAA mit IAM](#)

**IAM-Administrator** — Wenn Sie ein IAM-Administrator sind, möchten Sie vielleicht mehr darüber erfahren, wie Sie Richtlinien schreiben können, um den Zugriff auf Amazon MWAA zu verwalten. Beispiele für identitätsbasierte Amazon MWAA-Richtlinien, die Sie in IAM verwenden können, finden Sie unter [Beispiele für identitätsbasierte Amazon MWAA-Richtlinien](#)

## Authentifizierung mit Identitäten

Authentifizierung ist die Art und Weise, wie Sie sich mit Ihren Identitätsdaten anmelden. AWS Sie müssen als IAM-Benutzer authentifiziert (angemeldet AWS) sein oder eine IAM-Rolle annehmen. Root-Benutzer des AWS-Kontos

Sie können sich AWS als föderierte Identität anmelden, indem Sie Anmeldeinformationen verwenden, die über eine Identitätsquelle bereitgestellt wurden. AWS IAM Identity Center (IAM Identity Center) -Benutzer, die Single Sign-On-Authentifizierung Ihres Unternehmens und Ihre Google- oder Facebook-Anmeldeinformationen sind Beispiele für föderierte Identitäten. Wenn Sie sich als Verbundidentität anmelden, hat der Administrator vorher mithilfe von IAM-Rollen einen Identitätsverbund eingerichtet. Wenn Sie über den Verbund darauf zugreifen AWS, übernehmen Sie indirekt eine Rolle.

Je nachdem, welcher Benutzertyp Sie sind, können Sie sich beim AWS Management Console oder beim AWS Zugangsportal anmelden. Weitere Informationen zur Anmeldung finden Sie AWS unter [So melden Sie sich bei Ihrem an AWS-Konto](#) im AWS-Anmeldung Benutzerhandbuch.

Wenn Sie AWS programmgesteuert darauf zugreifen, AWS stellt es ein Software Development Kit (SDK) und eine Befehlszeilenschnittstelle (CLI) bereit, um Ihre Anfragen mithilfe Ihrer Anmeldeinformationen kryptografisch zu signieren. Wenn Sie keine AWS Tools verwenden, müssen Sie Anfragen selbst signieren. Weitere Informationen zur Verwendung der empfohlenen Methode, um Anfragen selbst zu [signieren, finden Sie im IAM-Benutzerhandbuch unter AWS API-Anfragen](#) signieren.

Unabhängig von der verwendeten Authentifizierungsmethode müssen Sie möglicherweise zusätzliche Sicherheitsinformationen angeben. AWS empfiehlt beispielsweise, die Multi-Faktor-Authentifizierung (MFA) zu verwenden, um die Sicherheit Ihres Kontos zu erhöhen. 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 einen erstellen AWS-Konto, beginnen Sie mit einer Anmeldeidentität, die vollständigen Zugriff auf alle AWS-Services Ressourcen im Konto hat. Diese Identität wird als AWS-Konto Root-Benutzer bezeichnet. Sie können darauf zugreifen, indem Sie sich mit der E-Mail-Adresse und dem Passwort anmelden, mit denen Sie das Konto erstellt haben. 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 innerhalb von Ihnen AWS-Konto, die über spezifische Berechtigungen für eine einzelne Person oder Anwendung verfügt. 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 innerhalb Ihres Unternehmens AWS-Konto , die über bestimmte Berechtigungen verfügt. Sie ist einem IAM-Benutzer vergleichbar, ist aber nicht mit einer bestimmten Person verknüpft. Sie können vorübergehend eine IAM-Rolle in der übernehmen, AWS Management Console indem Sie die Rollen [wechseln](#). Sie können eine Rolle übernehmen, 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. Bei einigen können Sie AWS-Services jedoch eine Richtlinie direkt an eine Ressource anhängen (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 Funktionen 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 verwenden, um Aktionen auszuführen AWS, gelten Sie als Principal. 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 aufruft AWS-Service, in Kombination mit der Anfrage, Anfragen an AWS-Service nachgelagerte Dienste zu stellen. FAS-Anfragen werden nur gestellt, wenn ein Dienst eine Anfrage erhält, für deren Abschluss Interaktionen mit anderen AWS-Services oder Ressourcen erforderlich sind. 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.
- **Dienstbezogene Rolle** — Eine dienstbezogene Rolle ist eine Art von Servicerolle, die mit einer verknüpft ist. AWS-Service Der Service kann die Rolle übernehmen, um eine Aktion in Ihrem Namen auszuführen. Servicebezogene Rollen erscheinen in Ihrem Dienst AWS-Konto und gehören dem Dienst. Ein IAM-Administrator kann die Berechtigungen für Service-verknüpfte Rollen anzeigen, aber nicht bearbeiten.
- **Anwendungen, die auf Amazon EC2 ausgeführt werden** — Sie können eine IAM-Rolle verwenden, um temporäre Anmeldeinformationen für Anwendungen zu verwalten, die auf einer EC2-Instance ausgeführt werden und API-Anfragen stellen AWS CLI . AWS Das ist eher zu empfehlen, als Zugriffsschlüssel innerhalb der EC2-Instance zu speichern. Um einer EC2-Instance eine AWS

Rolle zuzuweisen und sie allen ihren Anwendungen zur Verfügung zu stellen, erstellen Sie ein Instance-Profil, das an die Instance angehängt ist. 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

Sie kontrollieren den Zugriff, AWS indem Sie Richtlinien erstellen und diese an AWS Identitäten oder Ressourcen anhängen. Eine Richtlinie ist ein Objekt, AWS das, wenn es einer Identität oder Ressource zugeordnet ist, deren Berechtigungen definiert. AWS wertet diese Richtlinien aus, wenn ein Prinzipal (Benutzer, Root-Benutzer oder Rollensitzung) eine Anfrage stellt. Berechtigungen in den Richtlinien bestimmen, ob die Anforderung zugelassen oder abgelehnt wird. Die meisten Richtlinien werden 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 angeben, wer auf was Zugriff hat. 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 Rolleninformationen von der AWS Management Console AWS CLI, der oder der 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 System zuordnen können AWS-Konto. Zu den verwalteten Richtlinien gehören AWS verwaltete Richtlinien und vom 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](#). Zu den Prinzipalen können Konten, Benutzer, Rollen, Verbundbenutzer oder gehören. AWS-Services

Ressourcenbasierte Richtlinien sind Richtlinien innerhalb dieses Diensts. Sie können AWS verwaltete 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 und Amazon VPC sind Beispiele für Services, die ACLs unterstützen. AWS WAF 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 verbreitete 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. Eine explizite Zugriffsverweigerung in einer dieser Richtlinien setzt eine Zugriffserlaubnis außer Kraft. Weitere Informationen über Berechtigungsgrenzen finden Sie unter [Berechtigungsgrenzen für IAM-Entitäten](#) im IAM-Benutzerhandbuch.
- **Service Control Policies (SCPs)** — SCPs sind JSON-Richtlinien, die die maximalen Berechtigungen für eine Organisation oder Organisationseinheit (OU) in festlegen. AWS Organizations AWS Organizations ist ein Dienst zur Gruppierung und zentralen Verwaltung mehrerer Objekte AWS-Konten , die Ihrem Unternehmen gehören. Wenn Sie innerhalb einer Organisation alle Features aktivieren, können Sie Service-Kontrollrichtlinien (SCPs) auf alle oder einzelne Ihrer Konten anwenden. Das SCP schränkt die Berechtigungen für Entitäten in Mitgliedskonten ein, einschließlich der einzelnen Entitäten. Root-Benutzer des AWS-Kontos Weitere Informationen zu Organizations 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 darüber, wie AWS bestimmt wird,

ob eine Anfrage zulässig ist, wenn mehrere Richtlinientypen betroffen sind, finden Sie im IAM-Benutzerhandbuch unter [Bewertungslogik für Richtlinien](#).

## Benutzern die Berechtigung zur Anzeige eigener Berechtigungen erteilen

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 umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der AWS CLI OR-API. AWS

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ViewOwnUserInfo",
 "Effect": "Allow",
 "Action": [
 "iam:GetUserPolicy",
 "iam:ListGroupsWithUser",
 "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": "*"
 }
]
}
```

## Fehlerbehebung bei Amazon Managed Workflows für Identität und Zugriff auf Apache Airflow

Verwenden Sie die folgenden Informationen, um häufig auftretende Probleme zu diagnostizieren und zu beheben, die bei der Arbeit mit Amazon MWAA und IAM auftreten können.

### Ich bin nicht berechtigt, eine Aktion in Amazon MWAA durchzuführen

Wenn Ihnen AWS Management Console mitgeteilt wird, dass Sie nicht berechtigt sind, eine Aktion durchzuführen, müssen Sie sich an Ihren Administrator wenden, um Unterstützung zu erhalten. Ihr Administrator ist die Person, die Ihnen Ihren Benutzernamen und Ihr Passwort bereitgestellt hat.

### Ich bin nicht berechtigt, iam auszuführen: PassRole

Wenn Sie eine Fehlermeldung erhalten, dass Sie nicht berechtigt sind, die `iam:PassRole` Aktion durchzuführen, müssen Ihre Richtlinien aktualisiert werden, damit Sie eine Rolle an Amazon MWAA übergeben können.

Einige AWS-Services ermöglichen es Ihnen, eine bestehende Rolle an diesen Service zu übergeben, anstatt eine neue Servicerolle oder eine dienstbezogene Rolle zu erstellen. 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 Amazon MWAA 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.

Wenn Sie Hilfe benötigen, wenden Sie sich an Ihren Administrator. AWS Ihr Administrator hat Ihnen Ihre Anmeldeinformationen zur Verfügung gestellt.



## Ich möchte Personen außerhalb meines AWS Kontos den Zugriff auf meine Amazon MWAA-Ressourcen ermöglichen

Sie können eine Rolle erstellen, die Benutzer in anderen Konten oder Personen außerhalb Ihrer Organisation für den Zugriff auf Ihre Ressourcen verwenden können. Sie können festlegen, wem die Übernahme der Rolle anvertraut wird. Im Fall von Diensten, die ressourcenbasierte Richtlinien oder Zugriffskontrolllisten (Access Control Lists, ACLs) verwenden, können Sie diese Richtlinien verwenden, um Personen Zugriff auf Ihre Ressourcen zu gewähren.

Weitere Informationen dazu finden Sie hier:

- Informationen darüber, ob Amazon MWAA diese Funktionen unterstützt, finden Sie unter [So funktioniert Amazon MWAA mit IAM](#)
- Informationen dazu, wie Sie Zugriff auf Ihre Ressourcen gewähren können, AWS-Konten die Ihnen gehören, finden Sie im IAM-Benutzerhandbuch unter [Gewähren des Zugriffs auf einen IAM-Benutzer in einem anderen AWS-Konto , den Sie besitzen](#).
- Informationen dazu, wie Sie Dritten Zugriff auf Ihre Ressourcen gewähren können AWS-Konten, finden Sie [AWS-Konten im IAM-Benutzerhandbuch unter Gewähren des Zugriffs für Dritte](#).
- Informationen dazu, wie Sie über einen Identitätsverbund Zugriff gewähren, finden Sie unter [Gewähren von Zugriff für extern authentifizierte Benutzer \(Identitätsverbund\)](#) im IAM-Benutzerhandbuch.
- Informationen zum Unterschied zwischen der Verwendung von Rollen und ressourcenbasierten Richtlinien für den kontoübergreifenden Zugriff finden Sie unter [So unterscheiden sich IAM-Rollen von ressourcenbasierten Richtlinien](#) im IAM-Benutzerhandbuch.

## So funktioniert Amazon MWAA mit IAM

Amazon MWAA verwendet identitätsbasierte IAM-Richtlinien, um Berechtigungen für Amazon MWAA-Aktionen und -Ressourcen zu erteilen. Empfohlene Beispiele für benutzerdefinierte IAM-Richtlinien, mit denen Sie den Zugriff auf Ihre Amazon MWAA-Ressourcen steuern können, finden Sie unter [the section called “Zugreifen auf eine Amazon MWAA-Umgebung”](#)

Einen allgemeinen Überblick darüber, wie Amazon MWAA und andere AWS Services mit IAM zusammenarbeiten, finden Sie unter [AWS Services That Work with IAM im IAM-Benutzerhandbuch](#).



## Identitätsbasierte Amazon MWAA-Richtlinien

Mit identitätsbasierten IAM-Richtlinien können Sie angeben, welche Aktionen und Ressourcen zugelassen oder abgelehnt werden. Darüber hinaus können Sie die Bedingungen festlegen, unter denen Aktionen zugelassen oder abgelehnt werden. Amazon MWAA unterstützt bestimmte Aktionen, Ressourcen und Bedingungsschlüssel.

Die folgenden Schritte zeigen, wie Sie mithilfe der IAM-Konsole eine neue JSON-Richtlinie erstellen können. Diese Richtlinie gewährt Lesezugriff auf Ihre Amazon MWAA-Ressourcen.

So verwenden Sie den JSON-Richtlinienditor zum Erstellen einer Richtlinie

1. [Melden Sie sich bei der an AWS Management Console und öffnen Sie die IAM-Konsole unter `https://console.aws.amazon.com/iam/`.](https://console.aws.amazon.com/iam/)
2. Wählen Sie im Navigationsbereich auf der linken Seite Policies (Richtlinien).

Wenn Sie zum ersten Mal Policies (Richtlinien) auswählen, erscheint die Seite Welcome to Managed Policies (Willkommen bei verwalteten Richtlinien). Wählen Sie Get Started.

3. Wählen Sie oben auf der Seite Create policy (Richtlinie erstellen) aus.
4. Wählen Sie im Bereich Policy editor (Richtlinien-Editor) die Option JSON aus.
5. Geben Sie folgendes JSON-Richtliniendokument ein:

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

6. Wählen Sie Weiter aus.

**Note**

Sie können jederzeit zwischen den Editoroptionen Visual und JSON wechseln. Wenn Sie jedoch Änderungen vornehmen oder im Visual-Editor Weiter wählen, strukturiert IAM Ihre Richtlinie möglicherweise um, um sie für den visuellen Editor zu optimieren. Weitere Informationen finden Sie unter [Richtlinienrestrukturierung](#) im IAM-Benutzerhandbuch.

7. Geben Sie auf der Seite Prüfen und erstellen unter Richtliniennamen einen Namen und unter Beschreibung (optional) eine Beschreibung für die Richtlinie ein, die Sie erstellen. Überprüfen Sie Permissions defined in this policy (In dieser Richtlinie definierte Berechtigungen), um die Berechtigungen einzusehen, die von Ihrer Richtlinie gewährt werden.
8. Wählen Sie Create policy (Richtlinie erstellen) aus, um Ihre neue Richtlinie zu speichern.

Informationen zu sämtlichen Elementen, die Sie in einer JSON-Richtlinie verwenden, finden Sie in der [IAM-Referenz für JSON-Richtlinienelemente](#) im IAM-Benutzerhandbuch.

## Aktionen

Administratoren können mithilfe von AWS JSON-Richtlinien angeben, wer Zugriff auf was hat. Das heißt, welcher Prinzipal kann Aktionen für welche Ressourcen und unter welchen Bedingungen ausführen.

Das Element `Action` einer JSON-Richtlinie beschreibt die Aktionen, mit denen Sie den Zugriff in einer Richtlinie zulassen oder verweigern können. Richtlinienaktionen haben normalerweise denselben Namen wie der zugehörige AWS API-Vorgang. Es gibt einige Ausnahmen, z. B. Aktionen, die nur mit Genehmigung durchgeführt werden können und für die es keinen passenden API-Vorgang gibt. Es gibt auch einige Operationen, die mehrere Aktionen in einer Richtlinie erfordern. Diese zusätzlichen Aktionen werden als abhängige Aktionen bezeichnet.

Schließen Sie Aktionen in eine Richtlinie ein, um Berechtigungen zur Durchführung der zugeordneten Operation zu erteilen.

Richtlinienanweisungen müssen entweder ein `Action`- oder ein `NotAction`-Element enthalten. Das Element `Action` listet die Aktionen auf, die im Rahmen der Richtlinie zulässig sind. Das Element `NotAction` listet die Aktionen auf, die nicht zulässig sind.

Die für Amazon MWAA definierten Aktionen spiegeln Aufgaben wider, die Sie mit Amazon MWAA ausführen können. Richtlinienaktionen in Detective haben das folgende Präfix: `airflow:`.

Sie können Platzhalter (\*) verwenden, um mehrere Aktionen anzugeben. Anstatt diese Aktionen separat aufzulisten, können Sie beispielsweise Zugriff auf alle Aktionen gewähren, die mit dem Wort enden. `environment`

Eine Liste der Amazon MWAA-Aktionen finden Sie unter [Von Amazon Managed Workflows for Apache Airflow definierte Aktionen](#) im IAM-Benutzerhandbuch.

## Beispiele für identitätsbasierte Amazon MWAA-Richtlinien

Informationen zu den Amazon MWAA-Richtlinien finden Sie unter [Verwaltung des Zugriffs auf eine Amazon MWAA-Umgebung](#)

Standardmäßig sind IAM-Benutzer und -Rollen nicht berechtigt, Amazon MWAA-Ressourcen zu erstellen oder zu ändern. Sie können auch keine Aufgaben mit der AWS Management Console, AWS CLI, oder API ausführen.

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.

### Important

Wir empfehlen, IAM-Rollen und temporäre Anmeldeinformationen zu verwenden, um Zugriff auf Ihre Amazon MWAA-Ressourcen zu gewähren. Vermeiden Sie es, Berechtigungsrichtlinien direkt an Ihre IAM-Benutzer anzuhängen.

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](#)
- [Verwenden der Amazon MWAA-Konsole](#)
- [Benutzern die Berechtigung zur Anzeige eigener Berechtigungen erteilen](#)

## Bewährte Methoden für Richtlinien

Identitätsbasierte Richtlinien legen fest, ob jemand Amazon MWAA-Ressourcen in Ihrem Konto erstellen, darauf zugreifen oder diese löschen kann. Dies kann zusätzliche Kosten für Ihr verursachen AWS-Konto. Befolgen Sie beim Erstellen oder Bearbeiten identitätsbasierter Richtlinien die folgenden Anleitungen und Empfehlungen:

- Beginnen Sie mit AWS verwalteten Richtlinien und wechseln Sie zu Berechtigungen mit den geringsten Rechten — Verwenden Sie die AWS verwalteten Richtlinien, die Berechtigungen für viele gängige Anwendungsfälle gewähren, um Ihren Benutzern und Workloads zunächst Berechtigungen zu gewähren. Sie sind in Ihrem verfügbar. AWS-Konto Wir empfehlen Ihnen, die Berechtigungen weiter zu reduzieren, indem Sie vom AWS Kunden verwaltete Richtlinien definieren, die speziell auf Ihre Anwendungsfälle zugeschnitten sind. Weitere Informationen finden Sie unter [AWS -verwaltete Richtlinien](#) oder [AWS -verwaltete Richtlinien für Auftrags-Funktionen](#) 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 Serviceaktionen zu gewähren, wenn diese für einen bestimmten Zweck verwendet werden AWS-Service, z. AWS CloudFormation B. 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 [Richtlinienvvalidierung zum IAM Access Analyzer](#) im IAM-Benutzerhandbuch.

- Multi-Faktor-Authentifizierung (MFA) erforderlich — Wenn Sie ein Szenario haben, das IAM-Benutzer oder einen Root-Benutzer in Ihrem System erfordert AWS-Konto, 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.

## Verwenden der Amazon MWAA-Konsole

Um die Amazon MWAA-Konsole verwenden zu können, muss der Benutzer oder die Rolle Zugriff auf die entsprechenden Aktionen haben, die den entsprechenden Aktionen in der API entsprechen.

Informationen zu den Amazon MWAA-Richtlinien finden Sie unter. [Verwaltung des Zugriffs auf eine Amazon MWAA-Umgebung](#)

Benutzern die Berechtigung zur Anzeige eigener Berechtigungen erteilen

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 umfasst Berechtigungen zum Ausführen dieser Aktion auf der Konsole oder programmgesteuert mithilfe der API oder. AWS CLI AWS

```
{
 "Version": "2012-10-17",
 "Statement": [
 {
 "Sid": "ViewOwnUserInfo",
 "Effect": "Allow",
 "Action": [
 "iam:GetUserPolicy",
 "iam:ListGroupsWithUser",
 "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": "*"
 }
]
```

## Konformitätsprüfung für Amazon Managed Workflows für Apache Airflow

Informationen darüber, ob AWS-Service ein [AWS-Services in den Geltungsbereich bestimmter Compliance-Programme fällt, finden Sie unter Umfang nach Compliance-Programm AWS-Services unter](#) . Wählen Sie dort das Compliance-Programm aus, an dem Sie interessiert sind. Allgemeine Informationen finden Sie unter [AWS Compliance-Programme AWS](#) .

Sie können Prüfberichte von Drittanbietern unter herunterladen AWS Artifact. Weitere Informationen finden Sie unter [Berichte herunterladen unter](#) .

Ihre Verantwortung für die Einhaltung der Vorschriften bei der Nutzung AWS-Services hängt von der Vertraulichkeit Ihrer Daten, den Compliance-Zielen Ihres Unternehmens und den geltenden Gesetzen und Vorschriften ab. AWS stellt die folgenden Ressourcen zur Verfügung, die Sie bei der Einhaltung der Vorschriften unterstützen:

- [Schnellstartanleitungen zu Sicherheit und Compliance](#) — In diesen Bereitstellungsleitfäden werden architektonische Überlegungen erörtert und Schritte für die Implementierung von Basisumgebungen beschrieben AWS , bei denen Sicherheit und Compliance im Mittelpunkt stehen.
- [Architecting for HIPAA Security and Compliance on Amazon Web Services](#) — In diesem Whitepaper wird beschrieben, wie Unternehmen HIPAA-fähige Anwendungen erstellen AWS können.

**Note**

AWS-Services Nicht alle sind HIPAA-fähig. Weitere Informationen finden Sie in der [Referenz für HIPAA-berechtigte Services](#).

- [AWS Compliance-Ressourcen](#) — Diese Sammlung von Arbeitsmapen und Leitfäden gilt möglicherweise für Ihre Branche und Ihren Standort.
- [AWS Leitfäden zur Einhaltung von Vorschriften für Kunden](#) — Verstehen Sie das Modell der gemeinsamen Verantwortung aus dem Blickwinkel der Einhaltung von Vorschriften. In den Leitfäden werden die bewährten Verfahren zur Sicherung zusammengefasst AWS-Services und die Leitlinien den Sicherheitskontrollen in verschiedenen Frameworks (einschließlich des National Institute of Standards and Technology (NIST), des Payment Card Industry Security Standards Council (PCI) und der International Organization for Standardization (ISO)) zugeordnet.
- [Evaluierung von Ressourcen anhand von Regeln](#) im AWS Config Entwicklerhandbuch — Der AWS Config Service bewertet, wie gut Ihre Ressourcenkonfigurationen den internen Praktiken, Branchenrichtlinien und Vorschriften entsprechen.
- [AWS Security Hub](#)— Dies AWS-Service bietet einen umfassenden Überblick über Ihren internen Sicherheitsstatus. AWS Security Hub verwendet Sicherheitskontrollen, um Ihre AWS -Ressourcen zu bewerten und Ihre Einhaltung von Sicherheitsstandards und bewährten Methoden zu überprüfen. Eine Liste der unterstützten Services und Kontrollen finden Sie in der [Security-Hub-Steuerungsreferenz](#).
- [Amazon GuardDuty](#) — Dies AWS-Service erkennt potenzielle Bedrohungen für Ihre Workloads AWS-Konten, Container und Daten, indem es Ihre Umgebung auf verdächtige und böswillige Aktivitäten überwacht. GuardDuty kann Ihnen helfen, verschiedene Compliance-Anforderungen wie PCI DSS zu erfüllen, indem es die in bestimmten Compliance-Frameworks vorgeschriebenen Anforderungen zur Erkennung von Eindringlingen erfüllt.
- [AWS Audit Manager](#)— Auf diese AWS-Service Weise können Sie Ihre AWS Nutzung kontinuierlich überprüfen, um das Risikomanagement und die Einhaltung von Vorschriften und Industriestandards zu vereinfachen.

## Resilienz in Amazon Managed Workflows für Apache Airflow

Die AWS globale Infrastruktur basiert auf AWS Regionen und Availability Zones. Regionen stellen mehrere physisch getrennte und isolierte Availability Zones bereit, die über hoch redundante Netzwerke mit niedriger Latenz und hohen Durchsätzen 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.

Weitere Informationen zu AWS Regionen und Availability Zones finden Sie unter [AWS Globale Infrastruktur](#).

## Infrastruktursicherheit in Amazon MWAA

Als verwalteter Service ist Amazon Managed Workflows for Apache Airflow durch AWS globale Netzwerksicherheit geschützt. Informationen zu AWS Sicherheitsdiensten und zum AWS Schutz der Infrastruktur finden Sie unter [AWS Cloud-Sicherheit](#). Informationen zum Entwerfen Ihrer AWS Umgebung unter Verwendung der bewährten Methoden für die Infrastruktursicherheit finden Sie unter [Infrastructure Protection](#) in Security Pillar AWS Well-Architected Framework.

Sie verwenden AWS veröffentlichte API-Aufrufe, um über das Netzwerk auf Amazon MWAA 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.

## Konfiguration und Schwachstellenanalyse in Amazon MWAA

Konfiguration und IT-Steuerung liegen in der gemeinsamen Verantwortung AWS von Ihnen, unserem Kunden.

Amazon Managed Workflows for Apache Airflow patcht und aktualisiert Apache Airflow regelmäßig in Ihren Umgebungen. Sie sollten sicherstellen, dass die entsprechenden Zugriffsrichtlinien für Ihre VPCs verwendet werden.



Weitere Informationen finden Sie in den folgenden Ressourcen:

- [Konformitätsprüfung für Amazon Managed Workflows für Apache Airflow](#)
- [Modell der übergreifenden Verantwortlichkeit](#)
- [Amazon Web Services – Übersicht über Sicherheitsverfahren](#)
- [Infrastruktursicherheit in Amazon MWAA](#)
- [Bewährte Sicherheitsmethoden auf Amazon MWAA](#)

## Bewährte Sicherheitsmethoden auf Amazon MWAA

Amazon MWAA bietet eine Reihe von Sicherheitsfunktionen, die Sie bei der Entwicklung und Implementierung Ihrer eigenen Sicherheitsrichtlinien berücksichtigen sollten. Die folgenden bewährten Methoden sind allgemeine Richtlinien und keine vollständige Sicherheitslösung. Da diese bewährten Methoden für Ihre Umgebung möglicherweise nicht angemessen oder ausreichend sind, sollten Sie sie als hilfreiche Überlegungen und nicht als bindend ansehen.

- Verwenden Sie Richtlinien für Berechtigungen, die am wenigsten zulässig sind. Erteilen Sie Berechtigungen nur für die Ressourcen oder Aktionen, die Benutzer zur Ausführung von Aufgaben benötigen.
- Wird verwendet AWS CloudTrail , um die Benutzeraktivitäten in Ihrem Konto zu überwachen.
- Stellen Sie sicher, dass die Amazon S3 S3-Bucket-Richtlinie und die Objekt-ACLs den Benutzern aus der zugehörigen Amazon MWAA-Umgebung Berechtigungen zum Platzieren von Objekten in den Bucket gewähren. Dadurch wird sichergestellt, dass Benutzer mit der Berechtigung, Workflows zum Bucket hinzuzufügen, auch berechtigt sind, die Workflows in Airflow auszuführen.
- Verwenden Sie die Amazon S3 S3-Buckets, die Amazon MWAA-Umgebungen zugeordnet sind. Ihr Amazon S3 S3-Bucket kann einen beliebigen Namen haben. Speichern Sie keine anderen Objekte im Bucket und verwenden Sie den Bucket nicht mit einem anderen Service.

## Bewährte Sicherheitsmethoden in Apache Airflow

Apache Airflow ist nicht mandantenfähig. Zwar gibt es [Maßnahmen zur Zugriffskontrolle](#), um einige Funktionen auf bestimmte Benutzer zu beschränken, die [Amazon MWAA implementiert](#), aber DAG-Ersteller haben die Möglichkeit, DAGs zu schreiben, die die Benutzerrechte von Apache Airflow ändern und mit der zugrunde liegenden Metadatenbank interagieren können.

Wir empfehlen die folgenden Schritte, wenn Sie mit Apache Airflow auf Amazon MWAA arbeiten, um sicherzustellen, dass die Metadaten und DAGs Ihrer Umgebung sicher sind.

- Verwenden Sie separate Umgebungen für separate Teams mit DAG-Schreibzugriff oder der Möglichkeit, Dateien zu Ihrem Amazon S3 /dags S3-Ordner hinzuzufügen, vorausgesetzt, alles, auf das über die [Amazon MWAA-Ausführungsrolle](#) oder [Apache Airflow-Verbindungen](#) zugegriffen werden kann, ist auch für Benutzer zugänglich, die in die Umgebung schreiben können.
- Bieten Sie keinen direkten Zugriff auf Amazon S3 S3-DAGs-Ordner. Verwenden Sie stattdessen CI/CD-Tools, um DAGs in Amazon S3 zu schreiben, wobei ein Validierungsschritt sicherstellt, dass der DAG-Code den Sicherheitsrichtlinien Ihres Teams entspricht.
- Verhindern Sie den Benutzerzugriff auf den Amazon S3 S3-Bucket Ihrer Umgebung. Verwenden Sie stattdessen eine DAG-Factory, die DAGs auf der Grundlage einer YAML-, JSON- oder anderen Definitionsdatei generiert, die an einem anderen Ort als Ihrem Amazon MWAA Amazon S3 S3-Bucket gespeichert ist, in dem Sie DAGs speichern.
- Speichern Sie Geheimnisse im [Secrets Manager](#). Dadurch werden Benutzer, die DAGs schreiben können, zwar nicht daran gehindert, Geheimnisse zu lesen, sie werden jedoch daran gehindert, die von Ihrer Umgebung verwendeten Geheimnisse zu ändern.

## Erkennung von Änderungen an den Benutzerrechten von Apache Airflow

Sie können CloudWatch Logs Insights verwenden, um zu erkennen, dass DAGs die Benutzerrechte von Apache Airflow ändern. Zu diesem Zweck können Sie eine EventBridge geplante Regel, eine Lambda-Funktion und CloudWatch Logs Insights verwenden, um Benachrichtigungen an CloudWatch Metriken zu senden, wenn eines Ihrer DAGs die Benutzerrechte von Apache Airflow ändert.

### Voraussetzungen

Um die folgenden Schritte durchzuführen, benötigen Sie Folgendes:

- Eine Amazon MWAA-Umgebung, in der alle Apache Airflow-Protokolltypen auf Protokollebene aktiviert sind. INFO Weitere Informationen finden Sie unter [the section called “Airflow-Protokolle anzeigen”](#).

Um Benachrichtigungen für Änderungen der Apache Airflow-Benutzerrechte zu konfigurieren

1. [Erstellen Sie eine Lambda-Funktion](#), die die folgende CloudWatch Logs Insights-Abfragezeichenfolge für die fünf Protokollgruppen der Amazon MWAA-Umgebung (DAGProcessing,, Scheduler TaskWebServer, und) ausführt. Worker

```
fields @log, @timestamp, @message | filter @message like "add-role" | stats count()
by @log
```

2. [Erstellen Sie eine EventBridge Regel, die nach einem Zeitplan ausgeführt wird](#), wobei die Lambda-Funktion, die Sie im vorherigen Schritt erstellt haben, das Ziel der Regel ist. Konfigurieren Sie Ihren Zeitplan mithilfe eines Cron- oder Rate-Ausdrucks so, dass er in regelmäßigen Intervallen ausgeführt wird.

# Apache Airflow-Versionen auf Amazon Managed Workflows für Apache Airflow

Auf dieser Seite werden die von Amazon Managed Workflows for Apache Airflow unterstützten Apache Airflow-Versionen und die Strategien beschrieben, die wir für ein Upgrade auf die neueste Version empfehlen.

## Themen

- [Informationen zu Amazon MWAA-Versionen](#)
- [Aktuelle Version](#)
- [Apache Airflow-Versionen](#)
- [Apache Airflow-Komponenten](#)
- [Aktualisieren der Apache Airflow-Version](#)
- [Veraltete Versionen von Apache Airflow](#)
- [Unterstützung der Apache Airflow-Version und häufig gestellte Fragen](#)

## Informationen zu Amazon MWAA-Versionen

Amazon MWAA erstellt Container-Images, die Apache Airflow-Versionen mit anderen gängigen Binärdateien und Python-Bibliotheken bündeln. Das Image verwendet die Apache Airflow-Basisinstallation für die von Ihnen angegebene Version. Wenn Sie eine Umgebung erstellen, geben Sie eine zu verwendende Image-Version an. Sobald eine Umgebung erstellt wurde, verwendet sie weiterhin die angegebene Image-Version, bis Sie sie auf eine neuere Version aktualisieren.

## Aktuelle Version

Amazon MWAA unterstützt mehr als eine Apache Airflow-Version. Wenn Sie beim Erstellen einer Umgebung keine Image-Version angeben, erstellt Amazon MWAA eine Umgebung mit der neuesten unterstützten Version von Apache Airflow.

## Apache Airflow-Versionen

Die folgenden Apache Airflow-Versionen werden auf Amazon Managed Workflows für Apache Airflow unterstützt.

**Note**

- Ab Apache Airflow v2.2.2 unterstützt Amazon MWAA die Installation von Python-Anforderungen, Anbieterpaketen und benutzerdefinierten Plugins direkt auf dem Apache Airflow-Webserver.
- Ab Apache Airflow v2.7.2 muss Ihre Anforderungsdatei eine `---constraint`Anweisung enthalten. Wenn Sie keine Einschränkung angeben, gibt Amazon MWAA eine für Sie an, um sicherzustellen, dass die in Ihren Anforderungen aufgeführten Pakete mit der von Ihnen verwendeten Version von Apache Airflow kompatibel sind.

Weitere Informationen zum Einrichten von Einschränkungen in Ihrer Anforderungsdatei finden Sie unter [Installieren von Python-Abhängigkeiten](#).

| Apache Airflow-Version | Leitfaden für Apache Airflow                               | Apache Airflow-Einschränkungen                            | Python-Version              |
|------------------------|------------------------------------------------------------|-----------------------------------------------------------|-----------------------------|
| <a href="#">v2.8.1</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.8.1</a> | <a href="#">Apache Airflow v2.8.1-Einschränkungsdatei</a> | <a href="#">Python 3.11</a> |
| <a href="#">v2.7.2</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.7.2</a> | <a href="#">Apache Airflow v2.7.2-Einschränkungsdatei</a> | <a href="#">Python 3.11</a> |
| <a href="#">v2.6.3</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.6.3</a> | <a href="#">Apache Airflow v2.6.3-Einschränkungsdatei</a> | <a href="#">Python 3.10</a> |
| <a href="#">v2.5.1</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.5.1</a> | <a href="#">Apache Airflow v2.5.1-Einschränkungsdatei</a> | <a href="#">Python 3.10</a> |
| <a href="#">v2.4.3</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.4.3</a> | <a href="#">Apache Airflow v2.4.3-Einschränkungsdatei</a> | <a href="#">Python 3.10</a> |

| Apache Airflow-Version | Leitfaden für Apache Airflow                               | Apache Airflow-Einschränkungen                            | Python-Version             |
|------------------------|------------------------------------------------------------|-----------------------------------------------------------|----------------------------|
| <a href="#">v2.2.2</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.2.2</a> | <a href="#">Apache Airflow v2.2.2-Einschränkungsdatei</a> | <a href="#">Python 3.7</a> |
| <a href="#">v2.0.2</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.0.2</a> | <a href="#">Apache Airflow v2.0.2-Einschränkungsdatei</a> | <a href="#">Python 3.7</a> |

Weitere Informationen zur Migration Ihrer selbstverwalteten Apache-Airflow-Bereitstellungen oder zur Migration einer vorhandenen Amazon-MWAA-Umgebung, einschließlich Anweisungen zum Sichern Ihrer Metadatendatenbank, finden Sie im [Amazon-MWAA-Migrationshandbuch](#).

## Apache Airflow-Komponenten

In diesem Abschnitt wird die Anzahl der Apache-Airflow-Scheduler und -Worker beschrieben, die für jede Apache-Airflow-Version auf Amazon MWAA verfügbar sind, und eine Liste der wichtigsten Apache-Airflow-Features, die die Version angeben, die jedes Feature unterstützt.

### Scheduler

| Apache Airflow-Version      | Scheduler (Standard) | Scheduler (min.) | Scheduler (max.) |
|-----------------------------|----------------------|------------------|------------------|
| Apache Airflow v2 und höher | 2                    | 2                | 5                |

## Worker

| Airflow-Version   | Auftragnehmer (min.) | Worker (max.) | Worker (Standard) |
|-------------------|----------------------|---------------|-------------------|
| Apache Airflow v2 | 1                    | 25            | 10                |

## Aktualisieren der Apache Airflow-Version

Amazon MWAA unterstützt Nebenversions-Upgrades. Das bedeutet, dass Sie Ihre Umgebung von Version `x.y.z` auf aktualisieren können `x.2.z`, aber nicht auf eine neue Hauptversion, z. B. von `1.y.z` auf `2.y.z`.

### Note

Sie können die Apache Airflow-Version für Ihre Umgebung nicht herabstufen.

Weitere Informationen und detaillierte Anweisungen zum Aktualisieren Ihrer Workflow-Ressourcen und zum Aktualisieren der Umgebung auf eine neue Version finden Sie unter [the section called "Aktualisierung der Version"](#).

## Veraltete Versionen von Apache Airflow

In der folgenden Tabelle sind die veralteten Versionen von Apache Airflow in Amazon MWAA sowie die Erstveröffentlichung und das Ende des Supports für jede Version aufgeführt. Weitere Informationen zur Migration auf eine neuere Version finden Sie im [Amazon MWAA Migration Guide](#).

| Apache Airflow-Version | Veröffentlichungsdatum von Apache Airflow | Verfügbarkeitsdatum von Amazon MWAA | Datum des eingeschränkten Supports für Amazon MWAA | Ende des Supports für Amazon MWAA |
|------------------------|-------------------------------------------|-------------------------------------|----------------------------------------------------|-----------------------------------|
| v1.10.12               | 25. August 2020                           | 24. November 2021                   | 21. August 2023                                    | 21. Februar 2024                  |

| Apache Airflow-Version | Veröffentlichungsdatum von Apache Airflow | Verfügbarkeitsdatum von Amazon MWAA | Datum des eingeschränkten Supports für Amazon MWAA | Ende des Supports für Amazon MWAA |
|------------------------|-------------------------------------------|-------------------------------------|----------------------------------------------------|-----------------------------------|
| v2.0.2                 | 19. April 2021                            | 25. Mai 2021                        | 23. November 2023                                  | 29. April 2024                    |
| v2.2.2                 | 15. November 2022                         | 27. Januar 2022                     | 25. Januar 2024                                    | 27. Juni 2024                     |

## Unterstützung der Apache Airflow-Version und häufig gestellte Fragen

In Übereinstimmung mit dem Apache Airflow Community-[Versionsprozess und der Versionsrichtlinie](#) verpflichtet sich Amazon MWAA, mindestens drei Nebenversionen von Apache Airflow gleichzeitig zu unterstützen. Wir kündigen das Datum des Support-Laufzeitendes einer bestimmten Apache Airflow-Nebenversion mindestens 90 Tage vor dem Support-Laufzeitende an.

### Häufig gestellte Fragen

F: Wie lange unterstützt Amazon MWAA eine Apache Airflow-Version?

A: Amazon MWAA unterstützt eine Apache-Airflow-Nebenversion mindestens 12 Monate nach der ersten Verfügbarkeit.

F: Werde ich benachrichtigt, wenn der Support für eine Apache Airflow-Version auf Amazon MWAA endet?

A: Ja. Wenn in Amazon MWAA-Umgebungen in Ihrem Konto die Version ausgeführt wird, die sich dem Ende des Supports nähert, sendet Amazon MWAA eine Benachrichtigung über mit AWS Health Dashboard dem Ende des Supports.

F: Was passiert am begrenzten Support-Datum?

A: Am begrenzten Support-Datum können Sie keine neuen Amazon MWAA-Umgebungen mit der zugehörigen Version mehr erstellen. Ihre vorhandenen Umgebungen bleiben bis zum Ende des Supports verfügbar.

F: Was passiert am Datum des Support-Laufzeitendes?



A: Am Ende des Supports können Sie weiterhin auf Ihre vorhandenen Amazon MWAA-Umgebungen zugreifen, in denen die zugehörige, veraltete Version von Apache Airflow auf eigenes Risiko ausgeführt wird. Anweisungen zum Upgrade auf eine neuere Version von Apache Airflow auf Amazon MWAA finden Sie im [Amazon-MWAA-Migrationshandbuch](#).

### Important

Sie sind dafür verantwortlich, Ihre Amazon MWAA-Versionen auf dem neuesten Stand zu halten. AWS fordert alle Kunden auf, ihre Amazon MWAA-Umgebungen auf die neueste Version zu aktualisieren, um von den aktuellsten Sicherheits-, Datenschutz- und Verfügbarkeitsvorkehrungen zu profitieren. Wenn Sie Ihre Umgebung mit einer nicht unterstützten Version oder Software über das Veralterungsdatum hinaus betreiben, das als Legacy-Version bezeichnet wird, besteht eine höhere Wahrscheinlichkeit von Sicherheits-, Datenschutz- und Betriebsrisiken, einschließlich Ausfallzeiten. Indem Sie Ihre Amazon MWAA-Umgebung auf einer Legacy-Version betreiben, bestätigen Sie, dass Sie diese Risiken verstehen und wissentlich übernehmen, und erklären sich damit einverstanden, Ihr Upgrade auf die neueste Version so schnell wie möglich abzuschließen. Der fortgesetzte Betrieb Ihrer Umgebung auf einer Legacy-Version unterliegt der Vereinbarung, die Ihre Nutzung der AWS Services regelt.

Legacy-Versionen gelten als nicht allgemein verfügbar und bieten AWS keine Unterstützung mehr für die Legacy-Version. Daher AWS kann den Zugriff auf oder die Verwendung von Legacy-Versionen jederzeit einschränken, wenn AWS feststellt, dass die Legacy-Version ein Sicherheits- oder Haftungsrisiko oder ein Risiko von Gesundheitsrisiken für die Services, , AWS seine Bols oder andere Dritte darstellt. Ihre Entscheidung, Ihre Workloads weiterhin auf einer Legacy-Version auszuführen, kann dazu führen, dass Ihre Inhalte nicht verfügbar, beschädigt oder nicht wiederherstellbar sind. Umgebungen, die auf einer Legacy-Version ausgeführt werden, unterliegen Ausnahmen vom Service Level Agreement (SLA).

Umgebungen und zugehörige Software, die auf einer Legacy-Version ausgeführt wird, können Fehler, Fehler, Fehler und schädliche Komponenten enthalten. Dementsprechend AWS stellt die Legacy-Version so bereit, wie es ist, und berücksichtigt nicht die Informationen, die in der Vereinbarung oder den Nutzungsbedingungen enthalten sind.

Weitere Informationen zum -Modell AWS der geteilten Verantwortung von finden Sie unter [Gemeinsame Verantwortung](#) im AWS Well-Architected Framework.

# Amazon Managed Workflows für Apache Airflow Service-Endpunkte und Kontingente

Amazon Managed Workflows for Apache Airflow hat die folgenden Servicequoten und Endpunkte. Servicekontingente, auch Limits genannt, sind die maximale Anzahl von Serviceressourcen oder Vorgängen für Ihr AWS Konto.

Inhalt

- [Service-Endpunkte](#)
- [Servicekontingente](#)
- [Steigende Quoten](#)

## Service-Endpunkte

Eine Liste der Endpunkte für Amazon MWAA finden Sie unter [Amazon Managed Workflows for Apache Airflow](#) Endpoints and Quotas.

## Servicekontingente

| Kontingentname         | Beschreibung                                                         | Standardkontingent | Einstellbar |
|------------------------|----------------------------------------------------------------------|--------------------|-------------|
| Umgebungen             | Die maximale Anzahl von Amazon MWAA-Umgebungen pro Konto und Region. | 10                 | Ja          |
| Worker pro Umgebung    | Die maximale Anzahl von Mitarbeitern pro Amazon MWAA-Umgebung.       | 25                 | Ja          |
| Webserver pro Umgebung | Die maximale Anzahl von Webservern pro Amazon MWAA-Umgebung.         | 5                  | Ja          |

## Steigende Quoten

Sie können eine Erhöhung eines einstellbaren Kontingents beantragen, indem Sie einen [Antrag auf Erhöhung des Kontingents stellen](#).

# Häufig gestellte Fragen zu Amazon MWAA

Auf dieser Seite werden häufig gestellte Fragen beschrieben, die bei der Verwendung von Amazon Managed Workflows for Apache Airflow auftreten können.

## Inhalt

- [Unterstützte Versionen](#)
  - [Was unterstützt Amazon MWAA für Apache Airflow v2?](#)
  - [Warum werden ältere Versionen von Apache Airflow nicht unterstützt?](#)
  - [Welche Python-Version sollte ich verwenden?](#)
  - [Welche Version von verwendet pip Amazon MWAA?](#)
- [Anwendungsfälle](#)
  - [Wann sollte ich im AWS Step Functions Vergleich zu verwenden? Amazon MWAA?](#)
- [Umgebungsspezifikationen](#)
  - [Wie viel Aufgabenspeicher ist für jede Umgebung verfügbar?](#)
  - [Was ist das Standardbetriebssystem, das für Amazon MWAA-Umgebungen verwendet wird?](#)
  - [Kann ich ein benutzerdefiniertes Image für meine Amazon MWAA-Umgebung verwenden?](#)
  - [Ist Amazon MWAA HIPAA-konform?](#)
  - [Unterstützt Amazon MWAA Spot-Instances?](#)
  - [Unterstützt Amazon MWAA eine benutzerdefinierte Domain?](#)
  - [Kann ich SSH in meine Umgebung einbinden?](#)
  - [Warum ist eine selbstreferenzierende Regel für die VPC-Sicherheitsgruppe erforderlich?](#)
  - [Kann ich Umgebungen aus verschiedenen Gruppen in IAM ausblenden?](#)
  - [Kann ich temporäre Daten auf dem Apache Airflow Worker speichern?](#)
  - [Kann ich mehr als 25 Apache Airflow-Worker angeben?](#)
  - [Unterstützt Amazon MWAA gemeinsam genutzte Amazon VPCs oder gemeinsam genutzte Subnetze?](#)
- [Metriken](#)
  - [Welche Metriken werden verwendet, um zu bestimmen, ob Worker skaliert werden sollen?](#)
  - [Kann ich benutzerdefinierte Metriken in erstellen CloudWatch?](#)
- [DAGs , Operatoren, Verbindungen und andere Fragen](#)

- [Kann ich die verwenden PythonVirtualenvOperator?](#)
- [Wie lange braucht Amazon MWAA, um eine neue DAG-Datei zu erkennen?](#)
- [Warum wird meine DAG-Datei nicht von Apache Airflow aufgenommen?](#)
- [Kann ich ein plugins.zip oder requirements.txt aus einer Umgebung entfernen?](#)
- [Warum sehe ich meine Plugins nicht im Menü Admin-Plugins von Apache Airflow v2.0.2?](#)
- [Kann ich AWS Database Migration Service \(DMS\) Operatoren verwenden?](#)

## Unterstützte Versionen

### Was unterstützt Amazon MWAA für Apache Airflow v2?

Informationen dazu, was Amazon MWAA unterstützt, finden Sie unter [Apache Airflow-Versionen auf Amazon Managed Workflows für Apache Airflow](#).

### Warum werden ältere Versionen von Apache Airflow nicht unterstützt?

Wir unterstützen nur die neueste (ab dem Start) Apache Airflow Version Apache Airflow v1.10.12 aufgrund von Sicherheitsbedenken bei älteren Versionen.

### Welche Python-Version sollte ich verwenden?

Die folgenden Apache Airflow-Versionen werden auf Amazon Managed Workflows for Apache Airflow unterstützt.

#### Note

- Ab Apache Airflow v2.2.2 unterstützt Amazon MWAA die Installation von Python-Anforderungen, Anbieterpaketen und benutzerdefinierten Plugins direkt auf dem Apache Airflow-Webserver.
- Ab Apache Airflow v2.7.2 muss Ihre Anforderungsdatei eine `---constraint`Anweisung enthalten. Wenn Sie keine Einschränkung angeben, gibt Amazon MWAA eine für Sie an, um sicherzustellen, dass die in Ihren Anforderungen aufgeführten Pakete mit der von Ihnen verwendeten Version von Apache Airflow kompatibel sind.

Weitere Informationen zum Einrichten von Einschränkungen in Ihrer Anforderungsdatei finden Sie unter [Installieren von Python-Abhängigkeiten](#).

| Apache Airflow-Version | Leitfaden für Apache Airflow                               | Apache Airflow-Einschränkungen                            | Python-Version              |
|------------------------|------------------------------------------------------------|-----------------------------------------------------------|-----------------------------|
| <a href="#">v2.8.1</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.8.1</a> | <a href="#">Apache Airflow v2.8.1-Einschränkungsdatei</a> | <a href="#">Python 3.11</a> |
| <a href="#">v2.7.2</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.7.2</a> | <a href="#">Apache Airflow v2.7.2-Einschränkungsdatei</a> | <a href="#">Python 3.11</a> |
| <a href="#">v2.6.3</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.6.3</a> | <a href="#">Apache Airflow v2.6.3-Einschränkungsdatei</a> | <a href="#">Python 3.10</a> |
| <a href="#">v2.5.1</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.5.1</a> | <a href="#">Apache Airflow v2.5.1-Einschränkungsdatei</a> | <a href="#">Python 3.10</a> |
| <a href="#">v2.4.3</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.4.3</a> | <a href="#">Apache Airflow v2.4.3-Einschränkungsdatei</a> | <a href="#">Python 3.10</a> |
| <a href="#">v2.2.2</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.2.2</a> | <a href="#">Apache Airflow v2.2.2-Einschränkungsdatei</a> | <a href="#">Python 3.7</a>  |
| <a href="#">v2.0.2</a> | <a href="#">Referenzhandbuch für Apache Airflow v2.0.2</a> | <a href="#">Apache Airflow v2.0.2-Einschränkungsdatei</a> | <a href="#">Python 3.7</a>  |

Weitere Informationen zur Migration Ihrer selbstverwalteten Apache-Airflow-Bereitstellungen oder zur Migration einer vorhandenen Amazon-MWAA-Umgebung, einschließlich Anweisungen zum Sichern Ihrer Metadatenbank, finden Sie im [Amazon-MWAA-Migrationshandbuch](#).

## Welche Version von verwendet **pip** Amazon MWAA?

Für Umgebungen mit Apache Airflow v1.10.12 installiert Amazon MWAA pip Version 21.1.2.

**Note**

Amazon MWAA wird pip für Apache Airflow v1.10.12-Umgebungen nicht aktualisiert.

Für Umgebungen mit Apache Airflow v2 und höher installiert Amazon MWAA pip Version 21.3.1.

## Anwendungsfälle

### Wann sollte ich im AWS Step Functions Vergleich zu verwenden? Amazon MWAA?

1. Sie können Step Functions verwenden, um einzelne Kundenaufträge zu verarbeiten, da Step Functions skalieren kann, um die Nachfrage nach einer Bestellung oder einer Million Bestellungen zu decken.
2. Wenn Sie einen Workflow über Nacht ausführen, der die Bestellungen des Vortags verarbeitet, können Sie Step Functions oder Amazon MWAA verwenden. Mit Amazon MWAA können Sie eine Open-Source-Option verwenden, um den Workflow von den von Ihnen verwendeten AWS Ressourcen zu abstrahieren.

## Umgebungsspezifikationen

### Wie viel Aufgabenspeicher ist für jede Umgebung verfügbar?

Der Aufgabenspeicher ist auf 10 GB begrenzt und wird von [Amazon ECS Fargate 1.3](#) angegeben. Die Menge an RAM wird durch die von Ihnen angegebene Umgebungsklasse bestimmt. Weitere Informationen zu Umgebungsklassen finden Sie unter [Konfiguration der Amazon MWAA-Umgebungsklasse](#).

### Was ist das Standardbetriebssystem, das für Amazon MWAA-Umgebungen verwendet wird?

Amazon MWAA-Umgebungen werden auf Instances erstellt, auf denen Amazon Linux AMI ausgeführt wird.

## Kann ich ein benutzerdefiniertes Image für meine Amazon MWAA-Umgebung verwenden?

Benutzerdefinierte Images werden nicht unterstützt. Amazon MWAA verwendet Images, die auf Amazon Linux AMI basieren. Amazon MWAA installiert die zusätzlichen Anforderungen, indem `pip3 -r install` für die Anforderungen ausgeführt wird, die in der Datei `requirements.txt` angegeben sind, die Sie dem Amazon S3-Bucket für die Umgebung hinzufügen.

## Ist Amazon MWAA HIPAA-konform?

Amazon MWAA ist für den [Health Insurance Portability and Accountability Act \(HIPAA\)](#) berechtigt. Wenn Sie ein HIPAA Business Associate Addendum (BAA) mit eingerichtet haben AWS, können Sie Amazon MWAA für Workflows verwenden, die geschützte Gesundheitsinformationen (PHI) in Umgebungen verarbeiten, die am oder nach dem 14. November 2022 erstellt wurden.

## Unterstützt Amazon MWAA Spot-Instances?

Amazon MWAA unterstützt derzeit keine On-Demand-Amazon EC2-Spot-Instance-Typen für Apache Airflow. Eine Amazon MWAA-Umgebung kann jedoch Spot-Instances auf auslösen, z. B. Amazon EMR und Amazon EC2.

## Unterstützt Amazon MWAA eine benutzerdefinierte Domain?

Führen Sie einen der folgenden Schritte aus, um eine benutzerdefinierte Domain für Ihren Amazon MWAA-Hostnamen verwenden zu können:

- Für Amazon MWAA-Bereitstellungen mit öffentlichem Webserverzugriff können Sie Amazon CloudFront mit `Lambda@Edge` verwenden, um den Datenverkehr an Ihre Umgebung weiterzuleiten, und einen benutzerdefinierten Domänennamen zuordnen CloudFront. Weitere Informationen und ein Beispiel für die Einrichtung einer benutzerdefinierten Domäne für eine öffentliche Umgebung finden Sie im Beispiel für eine [benutzerdefinierte Amazon MWAA-Domäne für einen öffentlichen Webserver](#) im Amazon MWAA-Beispiel GitHub -Repository.
- Für Amazon MWAA-Bereitstellungen mit privatem Webserverzugriff können Sie einen Application Load Balancer (ALB) verwenden, um den Datenverkehr an Amazon MWAA weiterzuleiten und dem ALB einen benutzerdefinierten Domänennamen zuzuweisen. Weitere Informationen finden Sie unter [the section called "Verwenden eines Load Balancer \(fortgeschritten\)"](#).



## Kann ich SSH in meine Umgebung einbinden?

Obwohl SSH in einer Amazon MWAA-Umgebung nicht unterstützt wird, ist es möglich, eine DAG zu verwenden, um Bash-Befehle mit der `auszuführenBashOperator` auszuführen. Beispielsweise:

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago
with DAG(dag_id="any_bash_command_dag", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
 cli_command = BashOperator(
 task_id="bash_command",
 bash_command="{{ dag_run.conf['command'] }}"
)
```

Um die DAG in der Apache Airflow-Benutzeroberfläche auszulösen, verwenden Sie:

```
{ "command" : "your bash command" }
```

## Warum ist eine selbstreferenzierende Regel für die VPC-Sicherheitsgruppe erforderlich?

Indem Sie eine selbstreferenzierende Regel erstellen, beschränken Sie die Quelle auf dieselbe Sicherheitsgruppe in der VPC und sie ist nicht für alle Netzwerke geöffnet. Weitere Informationen hierzu finden Sie unter [the section called "Sicherheit in Ihrer VPC"](#).

## Kann ich Umgebungen aus verschiedenen Gruppen in IAM ausblenden?

Sie können den Zugriff einschränken, indem Sie einen Umgebungsnamen in angeben AWS Identity and Access Management. Die Sichtbarkeitsfilterung ist jedoch nicht in der AWS Konsole verfügbar. Wenn ein Benutzer eine Umgebung sehen kann, kann er alle Umgebungen sehen.

## Kann ich temporäre Daten auf dem Apache Airflow Worker speichern?

Ihre Apache Airflow Operators können temporäre Daten in der Workers speichern. Apache Airflow Workers können auf temporäre Dateien in der `/tmp` auf den Fargate-Containern für Ihre Umgebung zugreifen.

**Note**

Der gesamte Aufgabenspeicher ist gemäß [Amazon ECS Fargate 1.3](#) auf 10 GB begrenzt. Es gibt keine Garantie dafür, dass nachfolgende Aufgaben auf derselben Fargate-Container-Instance ausgeführt werden, die möglicherweise einen anderen /tmp Ordner verwendet.

## Kann ich mehr als 25 Apache Airflow-Worker angeben?

Ja. Sie können zwar bis zu 25 Apache Airflow-Worker in der Amazon MWAA-Konsole angeben, Sie können jedoch bis zu 50 in einer Umgebung konfigurieren, indem Sie eine Kontingenterhöhung beantragen. Weitere Informationen finden Sie unter [Anfordern einer Kontingenterhöhung](#).

## Unterstützt Amazon MWAA gemeinsam genutzte Amazon VPCs oder gemeinsam genutzte Subnetze?

Amazon MWAA unterstützt keine freigegebenen Amazon VPCs oder freigegebenen Subnetze. Die Amazon VPC, die Sie beim Erstellen einer Umgebung auswählen, sollte dem Konto gehören, das versucht, die Umgebung zu erstellen. Sie können jedoch Datenverkehr von einer Amazon VPC im Amazon MWAA-Konto an eine gemeinsam genutzte VPC weiterleiten. Weitere Informationen und ein Beispiel für das Routing von Datenverkehr an eine gemeinsam genutzte Amazon VPC finden Sie unter [Zentralisiertes ausgehendes Routing ins Internet](#) im Amazon-VPC-Transit-Gateways-Handbuch.

## Metriken

### Welche Metriken werden verwendet, um zu bestimmen, ob Worker skaliert werden sollen?

Amazon MWAA überwacht die QueuedTasks und RunningTasks in , CloudWatch um festzustellen, ob Apache Airflow Workers in Ihrer Umgebung skaliert werden soll. Weitere Informationen hierzu finden Sie unter [Überwachung und Metriken](#).

### Kann ich benutzerdefinierte Metriken in erstellen CloudWatch?

Nicht in der - CloudWatch Konsole. Sie können jedoch eine DAG erstellen, die benutzerdefinierte Metriken in schreibt CloudWatch. Weitere Informationen finden Sie unter [the section called "Verwenden einer DAG zum Schreiben benutzerdefinierter Metriken"](#).

## DAGs , Operatoren, Verbindungen und andere Fragen

### Kann ich die verwenden `PythonVirtualenvOperator`?

Die `PythonVirtualenvOperator` wird auf Amazon MWAA nicht explizit unterstützt, aber Sie können ein benutzerdefiniertes Plugin erstellen, das die verwendet `PythonVirtualenvOperator`. Einen Beispiel-Code finden Sie unter [the section called “Benutzerdefiniertes Plugin zum Patchen PythonVirtualenvOperator”](#).

### Wie lange braucht Amazon MWAA, um eine neue DAG-Datei zu erkennen?

DAGs werden regelmäßig vom Amazon S3-Bucket mit Ihrer Umgebung synchronisiert. Wenn Sie eine neue DAG-Datei hinzufügen, dauert es etwa 300 Sekunden, bis Amazon MWAA mit der Verwendung der neuen Datei beginnt. Wenn Sie eine vorhandene DAG aktualisieren, dauert es etwa 30 Sekunden, bis Amazon MWAA Ihre Updates erkennt.

Diese Werte, 300 Sekunden für neue DAGs und 30 Sekunden für Aktualisierungen vorhandener DAGs, entsprechen [min\\_file\\_process\\_interval](#) den Apache-Airflow-Konfigurationsoptionen [dag\\_dir\\_list\\_interval](#) bzw. .

### Warum wird meine DAG-Datei nicht von Apache Airflow aufgenommen?

Die folgenden Lösungen sind für dieses Problem möglich:

1. Überprüfen Sie, ob Ihre Ausführungsrolle über ausreichende Berechtigungen für Ihren Amazon S3-Bucket verfügt. Weitere Informationen hierzu finden Sie unter [Amazon MWAA-Ausführungsrolle](#).
2. Überprüfen Sie, ob für den Amazon S3-Bucket Block Public Access konfiguriert und Versioning aktiviert ist. Weitere Informationen hierzu finden Sie unter [Erstellen eines Amazon S3 cket cket erstellen](#).
3. Überprüfen Sie die DAG-Datei selbst. Stellen Sie beispielsweise sicher, dass jede DAG über eine eindeutige DAG-ID verfügt.

### Kann ich ein `plugins.zip` oder `requirements.txt` aus einer Umgebung entfernen?

Derzeit gibt es keine Möglichkeit, ein `plugins.zip` oder `requirements.txt` aus einer Umgebung zu entfernen, sobald sie hinzugefügt wurden, aber wir arbeiten an dem Problem. Zwischenzeitlich

besteht eine Problemumgehung darin, auf eine leere Text- bzw. ZIP-Datei zu verweisen. Weitere Informationen hierzu finden Sie unter [Löschen von Dateien auf Amazon S3](#).

## Warum sehe ich meine Plugins nicht im Menü Admin-Plugins von Apache Airflow v2.0.2?

Aus Sicherheitsgründen hat der Apache-Airflow-Webserver auf Amazon MWAA begrenzten Netzwerkausgang und installiert weder Plugins noch Python-Abhängigkeiten direkt auf dem Apache-Airflow-Webserver für Umgebungen der Version 2.0.2. Das angezeigte Plugin ermöglicht es Amazon MWAA, Ihre Apache Airflow-Benutzer in AWS Identity and Access Management (IAM) zu authentifizieren.

Um Plugins und Python-Abhängigkeiten direkt auf dem Webserver installieren zu können, empfehlen wir, eine neue Umgebung mit Apache Airflow v2.2 und höher zu erstellen. Amazon MWAA installiert Python-Abhängigkeiten und benutzerdefinierte Plugins direkt auf dem Webserver für Apache Airflow v2.2 und höher.

## Kann ich AWS Database Migration Service (DMS) Operatoren verwenden?

Amazon MWAA unterstützt [DMS-Operatoren](#). Dieser Operator kann jedoch nicht verwendet werden, um Aktionen in der Amazon-Aurora-PostgreSQL-Metadatendatenbank auszuführen, die einer Amazon-MWAA-Umgebung zugeordnet ist.

# Amazon Managed Workflows for Apache Airflow

In diesem Thema werden häufig auftretende Probleme und Fehler beschrieben, die bei der Verwendung von Apache Airflow in Amazon Managed Workflows for Apache Airflow auftreten können, sowie empfohlene Schritte zur Behebung dieser Fehler.

## Inhalt

- [Fehlerbehebung: DAGs, Operatoren, Verbindungen und andere Probleme in Apache Airflow v2](#)
  - [Verbindungen](#)
    - [Ich kann keine Verbindung zu Secrets Manager herstellen](#)
    - [Wie konfiguriere ich die Bedingungen für densecretsmanager:ResourceTag/<tag-key> Secrets Manager oder eine Ressourcenbeschränkung in meiner Ausführungsrollenrichtlinie?](#)
    - [Ich kann keine Verbindung zu Snowflake herstellen](#)
    - [Ich kann meine Verbindung in der Airflow-Benutzeroberfläche nicht sehen](#)
  - [Webserver](#)
    - [Ich sehe einen 5xx-Fehler beim Zugriff auf den Webserver](#)
    - [Ich sehe den Fehler „Der Scheduler scheint nicht zu laufen“](#)
  - [Aufgaben](#)
    - [Ich sehe, dass meine Aufgaben hängen bleiben oder nicht abgeschlossen werden](#)
  - [CLI](#)
    - [Ich sehe einen '503'-Fehler, wenn ich eine DAG in der CLI auslöse](#)
    - [Warum schlägt der dags backfill Apache Airflow CLI-Befehl fehl? Gibt es eine Problemumgehung?](#)
  - [Operatoren](#)
    - [Ich habe einen PermissionError: \[Errno 13\] Permission denied Fehler bei der Verwendung des S3Transform-Operators erhalten](#)
- [Fehlerbehebung: DAGs, Operatoren, Verbindungen und andere Probleme in Apache Airflow v1](#)
  - [requirements.txt wird aktualisiert](#)
    - [Das Hinzufügen apache-airflow-providers-amazon führt dazu, dass meine Umgebung ausfällt](#)
  - [Defekte DAG](#)
    - [Ich habe bei der Verwendung von Amazon DynamoDB DynamoDB-Operatoren die Fehlermeldung „Broken DAG“ erhalten](#)

- [Ich habe den Fehler „Broken DAG: Kein Modul namens psycpg2“ erhalten](#)
- [Ich habe bei der Verwendung der Slack-Operatoren die Fehlermeldung „Broken DAG“ erhalten](#)
- [Ich habe verschiedene Fehler bei der Installation von Google/GCP/ erhaltenBigQuery](#)
- [Ich habe den Fehler „Broken DAG: Kein Modul namens Cython“ erhalten](#)
- [Operatoren](#)
  - [Ich habe einen Fehler bei der Verwendung desBigQuery Operators erhalten](#)
- [Verbindungen](#)
  - [Ich kann keine Verbindung zu Snowflake herstellen](#)
  - [Ich kann keine Verbindung zu Secrets Manager herstellen](#)
  - [Ich kann auf '<DB-identifizier-name>.cluster-id keine Verbindung zu meinem MySQL-Server herstellen. <region>.rds.amazonaws.com'](#)
- [Webserver](#)
  - [Ich benutze dasBigQueryOperator und es führt zum Absturz meines Webservers](#)
  - [Ich sehe einen 5xx-Fehler beim Zugriff auf den Webserver](#)
  - [Ich sehe den Fehler „Der Scheduler scheint nicht zu laufen“](#)
- [Aufgaben](#)
  - [Ich sehe, dass meine Aufgaben hängen bleiben oder nicht abgeschlossen werden](#)
- [CLI](#)
  - [Ich sehe einen '503'-Fehler, wenn ich eine DAG in der CLI auslöse](#)
- [Problembehandlung: Erstellen und Aktualisieren einer Amazon MWAA-Umgebung](#)
  - [Aktualisieren von requirements.txt](#)
    - [Ich habe eine neue Version von meinem angegebenrequirements.txt und es dauert mehr als 20 Minuten, meine Umgebung zu aktualisieren](#)
- [Plug-ins](#)
  - [Unterstützt Amazon MWAA die Implementierung einer benutzerdefinierten Benutzeroberfläche?](#)
  - [Ich kann benutzerdefinierte UI-Änderungen auf dem Amazon MWAA Local Runner über Plugins implementieren, aber wenn ich versuche, dasselbe auf Amazon MWAA zu tun, sehe ich weder meine Änderungen noch irgendwelche Fehler. Warum passiert das?](#)

- [Ich kann die Option für S3 Block Public Access-Einstellungen nicht auswählen](#)
- [Erstellen der -Umgebung](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, und sie steckt im Status „Creating“ fest](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, aber der Status wird als „Erstellen fehlgeschlagen“ angezeigt](#)
  - [Ich habe versucht, eine VPC auszuwählen, und habe den Fehler „Netzwerkfehler“ erhalten](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, und habe den Fehler „Muss übergeben werden“ für den Dienst, die Partition oder die Ressource erhalten](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, und der Status wird als „Verfügbar“ angezeigt, aber wenn ich versuche, auf die Airflow-Benutzeroberfläche zuzugreifen, wird der Fehler „Empty Reply from Server“ oder „502 Bad Gateway“ angezeigt](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, und mein Benutzername besteht aus einer Reihe zufälliger Charakternamen](#)
- [Umgebung aktualisieren](#)
  - [Ich habe versucht, die Umgebungsklasse zu ändern, aber das Update ist fehlgeschlagen](#)
- [Zugangsumgebung](#)
  - [Ich kann nicht auf die Apache Airflow-Benutzeroberfläche zugreifen](#)
- [Problembehandlung: CloudWatch Protokolle und CloudTrail Fehler](#)
- [Logs \(Protokolle\)](#)
  - [Ich kann meine Aufgabenprotokolle nicht sehen, oder ich habe die Fehlermeldung „Remote-Protokoll aus Cloudwatch log\\_group lesen“ erhalten](#)
  - [Aufgaben schlagen fehl, wenn keine Protokolle vorhanden sind](#)
  - [Ich sehe einen Fehler " in ResourceAlreadyExistsException CloudTrail](#)
  - [Ich sehe den Fehler „Ungültige Anfrage“ in CloudTrail](#)
  - [In den Apache Airflow-Protokollen wird die Meldung „Eine 64-Bit-Oracle-Client-Bibliothek kann nicht gefunden werden: „libclntsh.so: cannot open shared object file: No such file or directory“ angezeigt](#)
  - [Ich sehe in meinen Scheduler-Protokollen die Meldung psychopg2, dass der Server die Verbindung unerwartet geschlossen hat](#)
  - [Ich sehe in meinen DAG-Verarbeitungsprotokollen die Meldung „Der Executor meldet die Aufgabeninstanz %s als abgeschlossen \(%s\), obwohl für die Aufgabe angegeben ist, dass sie %s ist“](#)

- [Ich erhalte die Meldung „Konnten keine Remote-Protokolle von log\\_group lesen: airflow-`{\*environmentName}` -Task log\\_stream: `\* {\*DAG\_ID} /\* {\*TASK\_ID} /\* {\*time} /\* {\*n}` .log.“ in meinen Aufgabenprotokollen](#)

## Fehlerbehebung: DAGs, Operatoren, Verbindungen und andere Probleme in Apache Airflow v2

Die Themen auf dieser Seite beschreiben Lösungen für Apache Airflow v2 Python-Abhängigkeiten, benutzerdefinierte Plugins, DAGs, Operatoren, Verbindungen, Aufgaben und Webserverprobleme, die in einer Amazon Managed Workflows for Apache Airflow-Umgebung auftreten können.

### Inhalt

- [Verbindungen](#)
  - [Ich kann keine Verbindung zu Secrets Manager herstellen](#)
  - [Wie konfiguriere ich die Bedingungen für `densecretsmanager:ResourceTag/<tag-key>` Secrets Manager oder eine Ressourcenbeschränkung in meiner Ausführungsrollenrichtlinie?](#)
  - [Ich kann keine Verbindung zu Snowflake herstellen](#)
  - [Ich kann meine Verbindung in der Airflow-Benutzeroberfläche nicht sehen](#)
- [Webserver](#)
  - [Ich sehe einen 5xx-Fehler beim Zugriff auf den Webserver](#)
  - [Ich sehe den Fehler „Der Scheduler scheint nicht zu laufen“](#)
- [Aufgaben](#)
  - [Ich sehe, dass meine Aufgaben hängen bleiben oder nicht abgeschlossen werden](#)
- [CLI](#)
  - [Ich sehe einen '503'-Fehler, wenn ich eine DAG in der CLI auslöse](#)
  - [Warum schlägt der `backfill` Apache Airflow CLI-Befehl fehl? Gibt es eine Problemumgehung?](#)
- [Operatoren](#)
  - [Ich habe einen `PermissionError: \[Errno 13\] Permission denied` Fehler bei der Verwendung des `S3Transform-Operators` erhalten](#)



## Verbindungen

Das folgende Thema beschreibt die Fehler, die auftreten können, wenn Sie eine Apache Airflow-Verbindung oder eine andere AWS Datenbank verwenden.

### Ich kann keine Verbindung zu Secrets Manager herstellen

Wir empfehlen die folgende Schritte:

1. Erfahren Sie, wie Sie geheime Schlüssel für Ihre Apache Airflow-Verbindung und Variablen in erstellen [the section called “Secrets Manager konfigurieren”](#).
2. Erfahren Sie, wie Sie den geheimen Schlüssel für eine Apache Airflow-Variable (test-variable) in verwenden [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Variable](#).
3. Erfahren Sie, wie Sie den geheimen Schlüssel für eine Apache Airflow-Verbindung (myconn) in verwenden [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Verbindung](#).

Wie konfiguriere ich die Bedingungen für den **secretsmanager:ResourceTag/<tag-key>** Secrets Manager oder eine Ressourcenbeschränkung in meiner Ausführungsrollenrichtlinie?

#### Note

Gilt für Apache Airflow Version 2.0 und früher.

Aufgrund eines bekannten Problems in Apache Airflow können Sie den Zugriff auf Secrets Manager-Geheimnisse derzeit nicht einschränken, indem Sie Bedingungsschlüssel oder andere Ressourceneinschränkungen in der Ausführungsrolle Ihrer Umgebung verwenden.

### Ich kann keine Verbindung zu Snowflake herstellen

Wir empfehlen die folgende Schritte:

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.

2. Fügen Sie der Datei requirements.txt für Ihre Umgebung die folgenden Einträge hinzu.

```
apache-airflow-providers-snowflake==1.3.0
```

3. Fügen Sie Ihrer DAG die folgenden Importe hinzu:

```
from airflow.providers.snowflake.operators.snowflake import SnowflakeOperator
```

Stellen Sie sicher, dass das Apache Airflow-Verbindungsobjekt die folgenden Schlüssel-Wert-Paare enthält:

1. Conn-ID: snowflake\_conn
2. Verbindungstyp: Schneeflocke
3. Gastgeber: <my account>. <my region if not us-west-2>.snowflakecomputing.com
4. Schema: <my schema>
5. Einloggen: <my user name>
6. Passwort: \*\*\*\*\*
7. Hafen: <port, if any>
8. Zusätzlich:

```
{
 "account": "<my account>",
 "warehouse": "<my warehouse>",
 "database": "<my database>",
 "region": "<my region if not using us-west-2 otherwise omit this line>"
}
```

Beispiel:

```
>>> import json
>>> from airflow.models.connection import Connection
>>> myconn = Connection(
... conn_id='snowflake_conn',
... conn_type='Snowflake',
... host='YOUR_ACCOUNT.YOUR_REGION.snowflakecomputing.com',
... schema='YOUR_SCHEMA',
... login='YOUR_USERNAME',
```

```
... password='YOUR_PASSWORD',
... port='YOUR_PORT'
... extra=json.dumps(dict(account='YOUR_ACCOUNT', warehouse='YOUR_WAREHOUSE',
... database='YOUR_DB_OPTION', region='YOUR_REGION')),
...)
```

## Ich kann meine Verbindung in der Airflow-Benutzeroberfläche nicht sehen

Apache Airflow stellt Verbindungsvorlagen in der Apache Airflow-Benutzeroberfläche bereit. Es verwendet dies, um die Verbindungs-URI-Zeichenfolge zu generieren, unabhängig vom Verbindungstyp. Wenn eine Verbindungsvorlage in der Apache Airflow-Benutzeroberfläche nicht verfügbar ist, kann eine alternative Verbindungsvorlage verwendet werden, um eine Verbindungs-URI-Zeichenfolge zu generieren, z. B. mithilfe der HTTP-Verbindungsvorlage.

Wir empfehlen die folgende Schritte:

1. Sehen Sie sich die von Amazon MWAA bereitgestellten Verbindungstypen in der Apache Airflow-Benutzeroberfläche unter [an In Amazon MWAA-Umgebungen installierte Apache Airflow-Anbieterpakete](#).
2. Sehen Sie sich die Befehle zum Erstellen einer Apache Airflow-Verbindung in der CLI unter [an Befehlsreferenz für Apache Airflow CLI](#).
3. Erfahren Sie, wie Sie Verbindungsvorlagen in der Apache Airflow-Benutzeroberfläche synonym für Verbindungstypen verwenden, die in der Apache Airflow-Benutzeroberfläche auf Amazon MWAA nicht verfügbar sind, unter [Übersicht über Verbindungsarten](#).

## Webserver

Das folgende Thema beschreibt die Fehler, die Sie für Ihren Apache Airflow-Webserver auf Amazon MWAA erhalten können.

### Ich sehe einen 5xx-Fehler beim Zugriff auf den Webserver

Wir empfehlen die folgende Schritte:

1. Überprüfen Sie die Konfigurationsoptionen von Apache Airflow. Stellen Sie sicher, dass die Schlüssel-Wert-Paare, die Sie als Apache Airflow-Konfigurationsoption angegeben haben AWS Secrets Manager, korrekt konfiguriert wurden. Weitere Informationen hierzu finden Sie unter [the section called “Ich kann keine Verbindung zu Secrets Manager herstellen”](#).

2. Überprüfen Sie `requirements.txt`. Stellen Sie sicher, dass das Airflow-Paket „Extras“ und andere in Ihrem Paket aufgelistete Bibliotheken mit Ihrer Apache Airflow-Version kompatibel sind.
3. Weitere Möglichkeiten zum Angeben von Python-Abhängigkeiten in einer `requirements.txt` Datei finden Sie unter [Verwaltung von Python-Abhängigkeiten in requirements.txt](#).

## Ich sehe den Fehler „Der Scheduler scheint nicht zu laufen“

Wenn der Scheduler nicht zu laufen scheint oder der letzte „Herzschlag“ vor mehreren Stunden empfangen wurde, werden Ihre DAGs möglicherweise nicht in Apache Airflow angezeigt und neue Aufgaben werden nicht geplant.

Wir empfehlen die folgende Schritte:

1. Vergewissern Sie sich, dass Ihre VPC-Sicherheitsgruppe eingehenden Zugriff auf den Port zulässt 5432. Dieser Port wird benötigt, um eine Verbindung zur Amazon Aurora PostgreSQL-Metadatenbank für Ihre Umgebung herzustellen. Nachdem diese Regel hinzugefügt wurde, geben Sie Amazon MWAA ein paar Minuten, und der Fehler sollte verschwinden. Weitere Informationen hierzu finden Sie unter [the section called “Sicherheit in Ihrer VPC”](#).

### Note

- Die Aurora PostgreSQL-Metadatenbank ist Teil der [Amazon MWAA-Servicearchitektur](#) und in Ihrer nicht sichtbar AWS-Konto.
- Datenbankfehler sind in der Regel ein Symptom für einen Schedulerausfall und nicht die Hauptursache.

2. Wenn der Scheduler nicht läuft, kann dies an einer Reihe von Faktoren liegen, z. B. an [Fehlern bei der Installation von Abhängigkeiten](#) oder einem [überlasteten Scheduler](#). Vergewissern Sie sich, dass Ihre DAGs, Plugins und Anforderungen ordnungsgemäß funktionieren, indem Sie sich die entsprechenden Protokollgruppen unter CloudWatch Logs ansehen. Weitere Informationen hierzu finden Sie unter [Überwachung und Metriken](#).

## Aufgaben

Das folgende Thema beschreibt die Fehler, die Sie bei Apache Airflow-Aufgaben in einer Umgebung erhalten können.

## Ich sehe, dass meine Aufgaben hängen bleiben oder nicht abgeschlossen werden

Wenn Ihre Apache Airflow-Aufgaben „hängen“ oder nicht abgeschlossen werden, empfehlen wir die folgenden Schritte:

1. Es kann eine große Anzahl von DAGs definiert sein. Reduzieren Sie die Anzahl der DAGs und führen Sie ein Update der Umgebung durch (z. B. das Ändern eines Log-Levels), um einen Reset zu erzwingen.
  - a. Airflow analysiert DAGs, unabhängig davon, ob sie aktiviert sind oder nicht. Wenn Sie mehr als 50% der Kapazität Ihrer Umgebung nutzen, können Sie den Apache Airflow Scheduler überfordern. Dies führt zu einer hohen Gesamtanalyse inCloudWatch Metriken oder zu langen DAG-Verarbeitungszeiten inCloudWatch Protokollen. Es gibt andere Möglichkeiten zur Optimierung von Apache Airflow-Konfigurationen, die in dieser Anleitung jedoch nicht behandelt werden.
  - b. Weitere Informationen zu den bewährten Methoden, die wir empfehlen, um die Leistung Ihrer Umgebung zu optimieren, finden Sie unter [the section called “Leistungsoptimierung für Apache Airflow”](#).
2. Möglicherweise befindet sich eine große Anzahl von Aufgaben in der Warteschlange. Dies wird häufig als eine große — und wachsende — Anzahl von Aufgaben im Status „Keine“ oder als große Anzahl unter Aufgaben in der Warteschlange und/oder Aufgaben, die noch ausstehen, angezeigtCloudWatch. Dies kann aus folgenden Gründen geschehen:
  - a. Wenn mehr Aufgaben ausgeführt werden müssen, als die Umgebung ausführen kann, und/oder wenn eine große Anzahl von Aufgaben, die vor der automatischen Skalierung in die Warteschlange gestellt wurden, Zeit hat, die Aufgaben zu erkennen und zusätzliche Worker bereitzustellen.
  - b. Wenn mehr Aufgaben ausgeführt werden müssen, als eine Umgebung ausführen kann, empfehlen wir, die Anzahl der Aufgaben, die Ihre DAGs gleichzeitig ausführen, zu reduzieren und/oder die Mindestanzahl der Apache Airflow Worker zu erhöhen.
  - c. Wenn es eine große Anzahl von Aufgaben gibt, die in der Warteschlange standen, bevor Autoscaling Zeit hatte, zusätzliche Worker zu erkennen und bereitzustellen, empfehlen wir, die Taskbereitstellung zu staffeln und/oder die Mindestanzahl der Apache Airflow Worker zu erhöhen.
  - d. Sie können den Befehl [update-environment](#) in derAWS Command Line Interface (AWS CLI) verwenden, um die minimale oder maximale Anzahl von Workern zu ändern, die in Ihrer Umgebung ausgeführt werden.

```
aws mwaas update-environment --name MyEnvironmentName --min-workers 2 --max-workers 10
```

- e. Weitere Informationen zu den bewährten Methoden, die wir empfehlen, um die Leistung Ihrer Umgebung zu optimieren, finden Sie unter [the section called “Leistungsoptimierung für Apache Airflow”](#).
3. Möglicherweise werden während der Ausführung Aufgaben gelöscht, die als Aufgabenprotokolle angezeigt werden, die ohne weitere Angabe in Apache Airflow beendet werden. Dies kann aus folgenden Gründen geschehen:
    - a. Wenn es einen kurzen Moment gibt, in dem 1) die aktuellen Aufgaben die aktuelle Umgebungskapazität überschreiten, gefolgt von 2) einigen Minuten, in denen keine Aufgaben ausgeführt werden oder in die Warteschlange gestellt werden, dann 3) neue Aufgaben in die Warteschlange gestellt werden.
    - b. Amazon MWAA Autoscaling reagiert auf das erste Szenario, indem zusätzliche Mitarbeiter hinzugefügt werden. Im zweiten Szenario werden die zusätzlichen Arbeiter entfernt. Einige der Aufgaben, die sich in der Warteschlange befinden, können dazu führen, dass die Worker gerade entfernt werden und enden, wenn der Container gelöscht wird.
    - c. Wir empfehlen, die Mindestanzahl an Mitarbeitern in Ihrer Umgebung zu erhöhen. Eine weitere Möglichkeit besteht darin, das Timing Ihrer DAGs und Aufgaben anzupassen, um sicherzustellen, dass diese Szenarien nicht eintreten.
    - d. Sie können auch festlegen, dass die Mindestanzahl an Mitarbeitern der maximalen Anzahl an Mitarbeitern in Ihrer Umgebung entspricht, wodurch die automatische Skalierung effektiv deaktiviert wird. Verwenden Sie den Befehl [update-environment](#) in der AWS Command Line Interface (AWS CLI), um die automatische Skalierung zu deaktivieren, indem Sie die Mindest- und Höchstanzahl von Workern auf die gleiche setzen.

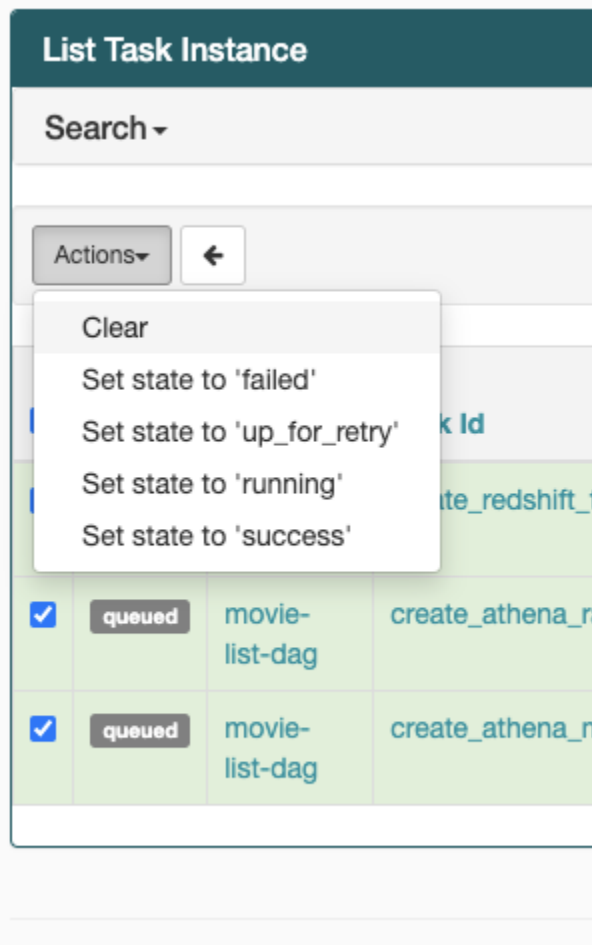
```
aws mwaas update-environment --name MyEnvironmentName --min-workers 5 --max-workers 5
```

- e. Weitere Informationen zu den bewährten Methoden, die wir empfehlen, um die Leistung Ihrer Umgebung zu optimieren, finden Sie unter [the section called “Leistungsoptimierung für Apache Airflow”](#).
4. Wenn Ihre Aufgaben im Status „Wird ausgeführt“ stecken, können Sie die Aufgaben auch löschen oder sie als erfolgreich oder fehlgeschlagen markieren. Dadurch kann die Autoscaling-

Komponente für Ihre Umgebung die Anzahl der Worker, die in Ihrer Umgebung ausgeführt werden, reduzieren. Die folgende Abbildung zeigt ein Beispiel für eine gestrandete Aufgabe.



- Wählen Sie den Kreis für die gestrandete Aufgabe und klicken Sie dann auf Löschen (wie abgebildet). Dadurch kann Amazon MWAA Mitarbeiter verkleinern. Andernfalls kann Amazon MWAA nicht feststellen, welche DAGs aktiviert oder deaktiviert sind, und kann nicht herunterskalieren, wenn es noch Aufgaben in der Warteschlange gibt.



5. Weitere Informationen zum Apache Airflow-Aufgabenlebenszyklus finden Sie unter [Konzepte](#) im Apache Airflow-Referenzleitfaden.

## CLI

Im folgenden Thema werden die Fehler beschrieben, die beim Ausführen von Airflow-CLI-Befehlen in der auftreten könnenAWS Command Line Interface.

## Ich sehe einen '503'-Fehler, wenn ich eine DAG in der CLI auslöse

Die Airflow-CLI wird auf dem Apache Airflow-Webserver ausgeführt, der über eine eingeschränkte Parallelität verfügt. In der Regel können maximal 4 CLI-Befehle gleichzeitig ausgeführt werden.

## Warum schlägt der `dags backfill` Apache Airflow CLI-Befehl fehl? Gibt es eine Problemumgehung?

### Note

Das Folgende gilt nur für Apache Airflow v2.0.2-Umgebungen.

Der `backfill` Befehl analysiert, wie andere Apache Airflow CLI-Befehle, alle DAGs lokal, bevor irgendwelche DAGs verarbeitet werden, unabhängig davon, für welche DAG die CLI-Operation gilt. In Amazon MWAA-Umgebungen, in denen Apache Airflow v2.0.2 verwendet wird, schlägt der Analysevorgang fehl und der Vorgang wird nicht aufgerufen, da Plugins und Anforderungen zum Zeitpunkt der Ausführung des CLI-Befehls noch nicht auf dem `backfill` Webserver installiert sind. Wenn Sie in Ihrer Umgebung keine Anforderungen oder Plugins hätten, wäre der `backfill` Vorgang erfolgreich.

Um den `backfill` CLI-Befehl ausführen zu können, empfehlen wir, ihn in einem Bash-Operator aufzurufen. Wird in einem Bash-Operator vom Worker initiiert, sodass die DAGs erfolgreich parsen können, da alle notwendigen Anforderungen und Plugins verfügbar und installiert sind. `backfill` Das folgende Beispiel zeigt, wie Sie eine DAG mit einer `BashOperator` zu ausführenden erstellen können `backfill`.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

with DAG(dag_id="backfill_dag", schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
 cli_command = BashOperator(
 task_id="bash_command",
 bash_command="airflow dags backfill my_dag_id"
)
```



## Operatoren

Im folgenden Thema werden die Fehler beschrieben, die bei der Verwendung von Operatoren auftreten können.

Ich habe einen **PermissionError: [Errno 13] Permission denied** Fehler bei der Verwendung des S3Transform-Operators erhalten

Wir empfehlen die folgenden Schritte, wenn Sie versuchen, ein Shell-Skript mit dem S3Transform-Operator auszuführen und eine **PermissionError: [Errno 13] Permission denied** Fehlermeldung erhalten. Die folgenden Schritte setzen voraus, dass Sie über eine vorhandene Datei `plugins.zip` verfügen. Informationen zum Erstellen einer neuen Datei `plugins.zip` finden Sie unter [Installation benutzerdefinierter Plugins](#).

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.
2. Erstellen Sie Ihr „Transform“-Skript.

```
#!/bin/bash
cp $1 $2
```

3. (optional) macOS- und Linux-Benutzer müssen möglicherweise den folgenden Befehl ausführen, um sicherzustellen, dass das Skript ausführbar ist.

```
chmod 777 transform_test.sh
```

4. Fügen Sie das Skript zu Ihrer `plugins.zip` hinzu.

```
zip plugins.zip transform_test.sh
```

5. Folgen Sie den Schritten [unter plugins.zip auf Amazon S3 hochladen](#).
6. Folgen Sie den Schritten [unter Angabe der Version plugins.zip auf der Amazon MWAA-Konsole](#).
7. Erstellen Sie die folgende DAG.

```
from airflow import DAG
from airflow.providers.amazon.aws.operators.s3_file_transform import
 S3FileTransformOperator
from airflow.utils.dates import days_ago
import os
```

```
DAG_ID = os.path.basename(__file__).replace(".py", "")

with DAG (dag_id=DAG_ID, schedule_interval=None, catchup=False,
 start_date=days_ago(1)) as dag:
 file_transform = S3FileTransformOperator(
 task_id='file_transform',
 transform_script='/usr/local/airflow/plugins/transform_test.sh',
 source_s3_key='s3://YOUR_S3_BUCKET/files/input.txt',
 dest_s3_key='s3://YOUR_S3_BUCKET/files/output.txt'
)
```

8. Folgen Sie den Schritten [unter DAG-Code auf Amazon S3 hochladen](#).

## Fehlerbehebung: DAGs, Operatoren, Verbindungen und andere Probleme in Apache Airflow v1

Die Themen auf dieser Seite enthalten Lösungen für Apache Airflow v1.10.12 Python-Abhängigkeiten, benutzerdefinierte Plugins, DAGs, Operatoren, Verbindungen, Aufgaben und Webserverprobleme, die in einer Amazon Managed Workflows for Apache Airflow-Umgebung auftreten können.

### Inhalt

- [requirements.txt wird aktualisiert](#)
  - [Das Hinzufügen von apache-airflow-providers-amazon führt dazu, dass meine Umgebung ausfällt](#)
- [Defekte DAG](#)
  - [Ich habe bei der Verwendung von Amazon DynamoDB DynamoDB-Operatoren die Fehlermeldung „Broken DAG“ erhalten](#)
  - [Ich habe den Fehler „Broken DAG: Kein Modul namens pycpg2“ erhalten](#)
  - [Ich habe bei der Verwendung der Slack-Operatoren die Fehlermeldung „Broken DAG“ erhalten](#)
  - [Ich habe verschiedene Fehler bei der Installation von Google/GCP/ erhalten BigQuery](#)
  - [Ich habe den Fehler „Broken DAG: Kein Modul namens Cython“ erhalten](#)
- [Operatoren](#)
  - [Ich habe einen Fehler bei der Verwendung des BigQuery Operators erhalten](#)
- [Verbindungen](#)
  - [Ich kann keine Verbindung zu Snowflake herstellen](#)

- [Ich kann keine Verbindung zu Secrets Manager herstellen](#)
- [Ich kann auf '<DB-identifizier-name>.cluster-id keine Verbindung zu meinem MySQL-Server herstellen. <region>.rds.amazonaws.com'](#)
- [Webserver](#)
  - [Ich benutze dasBigQueryOperator und es führt zum Absturz meines Webservers](#)
  - [Ich sehe einen 5xx-Fehler beim Zugriff auf den Webserver](#)
  - [Ich sehe den Fehler „Der Scheduler scheint nicht zu laufen“](#)
- [Aufgaben](#)
  - [Ich sehe, dass meine Aufgaben hängen bleiben oder nicht abgeschlossen werden](#)
- [CLI](#)
  - [Ich sehe einen '503'-Fehler, wenn ich eine DAG in der CLI auslöse](#)

## requirements.txt wird aktualisiert

Im folgenden Thema werden die Fehler beschrieben, die bei der Aktualisierung Ihres auftreten können `requirements.txt`.

Das Hinzufügen **apache-airflow-providers-amazon** führt dazu, dass meine Umgebung ausfällt

`apache-airflow-providers-xyz` ist nur mit Apache Airflow v2 kompatibel. `apache-airflow-backport-providers-xyz` ist kompatibel mit Apache Airflow 1.10.12.

## Defekte DAG

Im folgenden Thema werden die Fehler beschrieben, die beim Ausführen von DAGs auftreten können.

Ich habe bei der Verwendung von Amazon DynamoDB DynamoDB-Operatoren die Fehlermeldung „Broken DAG“ erhalten

Wir empfehlen die folgende Schritte:

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.
2. Fügen Sie das folgende Paket zu Ihrem `requirements.txt`.

```
boto
```

3. Weitere Möglichkeiten zum Angeben von Python-Abhängigkeiten in einer `requirements.txt` Datei finden Sie unter [Verwaltung von Python-Abhängigkeiten in requirements.txt](#).

Ich habe den Fehler „Broken DAG: Kein Modul namens `psycpg2`“ erhalten

Wir empfehlen die folgende Schritte:

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.
2. Fügen Sie Folgendes zu Ihrer `requirements.txt` Apache Airflow-Version hinzu. Beispiel:

```
apache-airflow[postgres]==1.10.12
```

3. Weitere Möglichkeiten zum Angeben von Python-Abhängigkeiten in einer `requirements.txt` Datei finden Sie unter [Verwaltung von Python-Abhängigkeiten in requirements.txt](#).

Ich habe bei der Verwendung der Slack-Operatoren die Fehlermeldung „Broken DAG“ erhalten

Wir empfehlen die folgende Schritte:

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.
2. Fügen Sie das folgende Paket zu Ihrer Apache Airflow-Version hinzu `requirements.txt` und geben Sie Ihre Apache Airflow-Version an. Beispiel:

```
apache-airflow[slack]==1.10.12
```

3. Weitere Möglichkeiten zum Angeben von Python-Abhängigkeiten in einer `requirements.txt` Datei finden Sie unter [Verwaltung von Python-Abhängigkeiten in requirements.txt](#).

Ich habe verschiedene Fehler bei der Installation von Google/GCP/ erhalten BigQuery

Amazon MWAA verwendet Amazon Linux, für das eine bestimmte Version der Cython- und Kryptografiebibliotheken erforderlich ist. Wir empfehlen die folgende Schritte:

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.
2. Fügen Sie das folgende Paket zu Ihrem `hinzurequirements.txt`.

```
grpcio==1.27.2
cython==0.29.21
pandas-gbq==0.13.3
cryptography==3.3.2
apache-airflow-backport-providers-amazon[google]
```

3. Wenn Sie keine Backport-Anbieter verwenden, können Sie Folgendes verwenden:

```
grpcio==1.27.2
cython==0.29.21
pandas-gbq==0.13.3
cryptography==3.3.2
apache-airflow[gcp]==1.10.12
```

4. Weitere Möglichkeiten zum Angeben von Python-Abhängigkeiten in einer `requirements.txt` Datei finden Sie unter [Verwaltung von Python-Abhängigkeiten in requirements.txt](#).

## Ich habe den Fehler „Broken DAG: Kein Modul namens Cython“ erhalten

Amazon MWAA verwendet Amazon Linux, für das eine bestimmte Version von Cython erforderlich ist. Wir empfehlen die folgende Schritte:

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.
2. Fügen Sie das folgende Paket zu Ihrem `hinzurequirements.txt`.

```
cython==0.29.21
```

3. Cython-Bibliotheken haben verschiedene erforderliche Pip-Abhängigkeitsversionen. Verwenden Sie beispielsweise `awswrangler==2.4.0 requires pyarrow<3.1.0, >=2.0.0`, sodass pip3 versucht, es zu installieren, `pyarrow==3.0.0` was zu einem defekten DAG-Fehler führt. Wir empfehlen, die älteste akzeptable Version explizit anzugeben. Wenn Sie beispielsweise den Mindestwert `pyarrow==2.0.0` vorher angeben, `awswrangler==2.4.0` verschwindet der Fehler und die `requirements.txt` Installation wird korrekt durchgeführt. Die endgültigen Anforderungen sollten wie folgt aussehen:

```
cython==0.29.21
pyarrow==2.0.0
awsxwangler==2.4.0
```

4. Weitere Möglichkeiten zum Angeben von Python-Abhängigkeiten in einer `requirements.txt` Datei finden Sie unter [Verwaltung von Python-Abhängigkeiten in requirements.txt](#).

## Operatoren

Im folgenden Thema werden die Fehler beschrieben, die bei der Verwendung von Operatoren auftreten können.

### Ich habe einen Fehler bei der Verwendung des BigQuery Operators erhalten

Amazon MWAA unterstützt keine Operatoren mit Benutzeroberflächenerweiterungen. Wir empfehlen die folgende Schritte:

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.
2. Eine Problemlösung besteht darin, die Erweiterung zu überschreiben, indem Sie der DAG eine Zeile hinzufügen, die `<operator name>.operator_extra_links = None` nach dem Import der problematischen Operatoren festgelegt werden soll. Beispiel:

```
from airflow.contrib.operators.bigquery_operator import BigQueryOperator
BigQueryOperator.operator_extra_links = None
```

3. Sie können diesen Ansatz für alle DAGs verwenden, indem Sie den oben genannten Ansatz zu einem Plugin hinzufügen. Ein Beispiel finden Sie unter [the section called "Benutzerdefiniertes Plugin zum Patchen PythonVirtualenvOperator"](#).

## Verbindungen

Das folgende Thema beschreibt die Fehler, die auftreten können, wenn Sie eine Apache Airflow-Verbindung oder eine andere AWS Datenbank verwenden.

### Ich kann keine Verbindung zu Snowflake herstellen

Wir empfehlen die folgende Schritte:

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.
2. Fügen Sie der Datei requirements.txt für Ihre Umgebung die folgenden Einträge hinzu.

```
asn1crypto == 0.24.0
snowflake-connector-python == 1.7.2
```

3. Fügen Sie Ihrer DAG die folgenden Importe hinzu:

```
from airflow.contrib.hooks.snowflake_hook import SnowflakeHook
from airflow.contrib.operators.snowflake_operator import SnowflakeOperator
```

Stellen Sie sicher, dass das Apache Airflow-Verbindungsobjekt die folgenden Schlüssel-Wert-Paare enthält:

1. Conn-ID: snowflake\_conn
2. Verbindungstyp: Schneeflocke
3. Gastgeber: <my account>. <my region if not us-west-2>.snowflakecomputing.com
4. Schema: <my schema>
5. Einloggen: <my user name>
6. Passwort: \*\*\*\*\*
7. Hafen: <port, if any>
8. Zusätzlich:

```
{
 "account": "<my account>",
 "warehouse": "<my warehouse>",
 "database": "<my database>",
 "region": "<my region if not using us-west-2 otherwise omit this line>"
}
```

Beispiel:

```
>>> import json
>>> from airflow.models.connection import Connection
>>> myconn = Connection(
```

```
... conn_id='snowflake_conn',
... conn_type='Snowflake',
... host='YOUR_ACCOUNT.YOUR_REGION.snowflakecomputing.com',
... schema='YOUR_SCHEMA'
... login='YOUR_USERNAME',
... password='YOUR_PASSWORD',
... port='YOUR_PORT'
... extra=json.dumps(dict(account='YOUR_ACCOUNT', warehouse='YOUR_WAREHOUSE',
database='YOUR_DB_OPTION', region='YOUR_REGION')),
...)
```

## Ich kann keine Verbindung zu Secrets Manager herstellen

Wir empfehlen die folgende Schritte:

1. Erfahren Sie, wie Sie geheime Schlüssel für Ihre Apache Airflow-Verbindung und Variablen in erstellen [the section called “Secrets Manager konfigurieren”](#).
2. Erfahren Sie, wie Sie den geheimen Schlüssel für eine Apache Airflow-Variable (test-variable) in verwenden [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Variable](#).
3. Erfahren Sie, wie Sie den geheimen Schlüssel für eine Apache Airflow-Verbindung (myconn) in verwenden [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Verbindung](#).

Ich kann auf '<DB-identifizier-name>.cluster-id keine Verbindung zu meinem MySQL-Server herstellen. <region>.rds.amazonaws.com'

Die Sicherheitsgruppe von Amazon MWAA und die RDS-Sicherheitsgruppe benötigen eine Eingangsregel, um Datenverkehr zu und von einander zuzulassen. Wir empfehlen die folgende Schritte:

1. Ändern Sie die RDS-Sicherheitsgruppe, um den gesamten Datenverkehr von der VPC-Sicherheitsgruppe von Amazon MWAA zuzulassen.
2. Ändern Sie die VPC-Sicherheitsgruppe von Amazon MWAA, um den gesamten Datenverkehr von der RDS-Sicherheitsgruppe zuzulassen.
3. Führen Sie Ihre Aufgaben erneut aus und überprüfen Sie, ob die SQL-Abfrage erfolgreich war, indem Sie die Apache CloudWatch Airflow-Protokolle in den Protokollen überprüfen.



## Webserver

Das folgende Thema beschreibt die Fehler, die Sie für Ihren Apache Airflow-Webserver auf Amazon MWAA erhalten können.

### Ich benutze dasBigQueryOperator und es führt zum Absturz meines Webservers

Wir empfehlen die folgende Schritte:

1. Apache Airflow-Operatoren wie dieBigQueryOperator undQuboleOperator die enthaltenoperator\_extra\_links können zum Absturz Ihres Apache Airflow-Webservers führen. Diese Operatoren versuchen, Code auf Ihren Webserver zu laden, was aus Sicherheitsgründen nicht zulässig ist. Wir empfehlen, die Operatoren in Ihrer DAG zu patchen, indem Sie nach Ihren Importanweisungen den folgenden Code hinzufügen:

```
BigQueryOperator.operator_extra_links = None
```

2. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#)onGitHub.

### Ich sehe einen 5xx-Fehler beim Zugriff auf den Webserver

Wir empfehlen die folgende Schritte:

1. Überprüfen Sie die Konfigurationsoptionen von Apache Airflow. Stellen Sie sicher, dass die Schlüssel-Wert-Paare, die Sie als Apache Airflow-Konfigurationsoption angegeben haben, wie z. B.AWS Secrets Manager, korrekt konfiguriert wurden. Weitere Informationen hierzu finden Sie unter [the section called “Ich kann keine Verbindung zu Secrets Manager herstellen”](#).
2. Überprüfe dierequirements.txt. Stellen Sie sicher, dass das Airflow-Paket „Extras“ und andere in Ihrem Paket aufgelistete Bibliotheken mit Ihrer Apache Airflow-Version kompatibelrequirements.txt sind.
3. Weitere Möglichkeiten zum Angeben von Python-Abhängigkeiten in einerrequirements.txt Datei finden Sie unter[Verwaltung von Python-Abhängigkeiten in requirements.txt](#).

## Ich sehe den Fehler „Der Scheduler scheint nicht zu laufen“

Wenn der Scheduler nicht zu laufen scheint oder der letzte „Herzschlag“ vor mehreren Stunden empfangen wurde, werden Ihre DAGs möglicherweise nicht in Apache Airflow angezeigt und neue Aufgaben werden nicht geplant.

Wir empfehlen die folgende Schritte:

1. Vergewissern Sie sich, dass Ihre VPC-Sicherheitsgruppe eingehenden Zugriff auf den Port zulässt 5432. Dieser Port wird benötigt, um eine Verbindung zur Amazon Aurora PostgreSQL-Metadatenbank für Ihre Umgebung herzustellen. Nachdem diese Regel hinzugefügt wurde, geben Sie Amazon MWAA ein paar Minuten, und der Fehler sollte verschwinden. Weitere Informationen hierzu finden Sie unter [the section called “Sicherheit in Ihrer VPC”](#).

### Note

- Die Aurora PostgreSQL-Metadatenbank ist Teil der [Amazon MWAA-Servicearchitektur](#) und in Ihrer nicht sichtbar AWS-Konto.
- Datenbankfehler sind in der Regel ein Symptom für einen Schedulerausfall und nicht die Hauptursache.

2. Wenn der Scheduler nicht läuft, kann dies an einer Reihe von Faktoren liegen, z. B. an [Fehlern bei der Installation von Abhängigkeiten](#) oder einem [überlasteten Scheduler](#). Vergewissern Sie sich, dass Ihre DAGs, Plugins und Anforderungen ordnungsgemäß funktionieren, indem Sie sich die entsprechenden Protokollgruppen unter CloudWatch Logs ansehen. Weitere Informationen hierzu finden Sie unter [Überwachung und Metriken](#).

## Aufgaben

Das folgende Thema beschreibt die Fehler, die Sie bei Apache Airflow-Aufgaben in einer Umgebung erhalten können.

### Ich sehe, dass meine Aufgaben hängen bleiben oder nicht abgeschlossen werden

Wenn Ihre Apache Airflow-Aufgaben „hängen“ oder nicht abgeschlossen werden, empfehlen wir die folgenden Schritte:

1. Es kann eine große Anzahl von DAGs definiert sein. Reduzieren Sie die Anzahl der DAGs und führen Sie ein Update der Umgebung durch (z. B. das Ändern eines Log-Levels), um einen Reset zu erzwingen.
  - a. Airflow analysiert DAGs, unabhängig davon, ob sie aktiviert sind oder nicht. Wenn Sie mehr als 50% der Kapazität Ihrer Umgebung nutzen, können Sie den Apache Airflow Scheduler überfordern. Dies führt zu einer hohen Gesamtanalysezeit in CloudWatch Metriken oder zu langen DAG-Verarbeitungszeiten in CloudWatch Protokollen. Es gibt jedoch nicht behandelt. Sie werden in dieser Anleitung jedoch nicht behandelt.
  - b. Weitere Informationen zu den bewährten Methoden, die wir empfehlen, um die Leistung Ihrer Umgebung zu optimieren, finden Sie unter [the section called “Leistungsoptimierung für Apache Airflow”](#).
2. Möglicherweise befindet sich eine große Anzahl von Aufgaben in der Warteschlange. Dies wird häufig als eine große — und wachsende — Anzahl von Aufgaben im Status „Keine“ oder als große Anzahl unter Aufgaben in der Warteschlange und/oder Aufgaben, die noch ausstehen, angezeigt in CloudWatch. Dies kann aus folgenden Gründen geschehen:
  - a. Wenn mehr Aufgaben ausgeführt werden müssen, als die Umgebung ausführen kann, und/oder wenn eine große Anzahl von Aufgaben, die vor der automatischen Skalierung in die Warteschlange gestellt wurden, Zeit hat, die Aufgaben zu erkennen und zusätzliche Worker bereitzustellen.
  - b. Wenn mehr Aufgaben ausgeführt werden müssen, als eine Umgebung ausführen kann, empfehlen wir, die Anzahl der Aufgaben, die Ihre DAGs gleichzeitig ausführen, zu reduzieren und/oder die Mindestanzahl der Apache Airflow Worker zu erhöhen.
  - c. Wenn es eine große Anzahl von Aufgaben gibt, die in der Warteschlange standen, bevor Autoscaling Zeit hatte, zusätzliche Worker zu erkennen und bereitzustellen, empfehlen wir, die Taskbereitstellung zu staffeln und/oder die Mindestanzahl der Apache Airflow Worker zu erhöhen.
  - d. Sie können den Befehl [update-environment](#) in der AWS Command Line Interface (AWS CLI) verwenden, um die minimale oder maximale Anzahl von Workern zu ändern, die in Ihrer Umgebung ausgeführt werden.

```
aws mwa update-environment --name MyEnvironmentName --min-workers 2 --max-workers 10
```

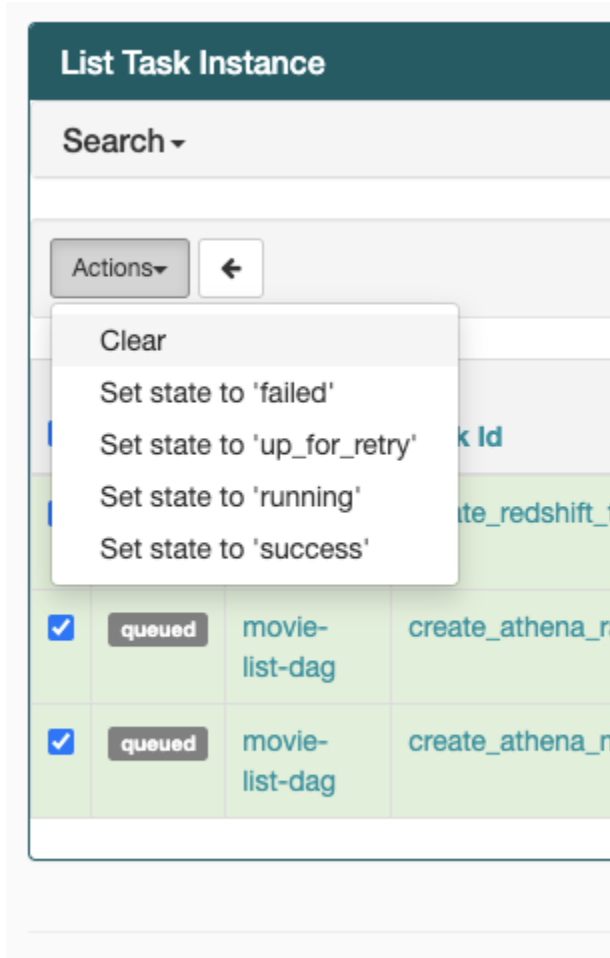
- e. Weitere Informationen zu den bewährten Methoden, die wir empfehlen, um die Leistung Ihrer Umgebung zu optimieren, finden Sie unter [the section called “Leistungsoptimierung für Apache Airflow”](#).
3. Möglicherweise werden während der Ausführung Aufgaben gelöscht, die als Aufgabenprotokolle angezeigt werden, die ohne weitere Angabe in Apache Airflow beendet werden. Dies kann aus folgenden Gründen geschehen:
- a. Wenn es einen kurzen Moment gibt, in dem 1) die aktuellen Aufgaben die aktuelle Umgebungskapazität überschreiten, gefolgt von 2) einigen Minuten, in denen keine Aufgaben ausgeführt werden oder in die Warteschlange gestellt werden, dann 3) neue Aufgaben in die Warteschlange gestellt werden.
  - b. Amazon MWAA Autoscaling reagiert auf das erste Szenario, indem zusätzliche Mitarbeiter hinzugefügt werden. Im zweiten Szenario werden die zusätzlichen Arbeiter entfernt. Einige der Aufgaben, die sich in der Warteschlange befinden, können dazu führen, dass die Worker gerade entfernt werden und enden, wenn der Container gelöscht wird.
  - c. Wir empfehlen, die Mindestanzahl an Mitarbeitern in Ihrer Umgebung zu erhöhen. Eine weitere Möglichkeit besteht darin, das Timing Ihrer DAGs und Aufgaben anzupassen, um sicherzustellen, dass diese Szenarien nicht eintreten.
  - d. Sie können auch festlegen, dass die Mindestanzahl an Mitarbeitern der maximalen Anzahl an Mitarbeitern in Ihrer Umgebung entspricht, wodurch die automatische Skalierung effektiv deaktiviert wird. Verwenden Sie den Befehl [update-environment](#) in der AWS Command Line Interface (AWS CLI), um die automatische Skalierung zu deaktivieren, indem Sie die Mindest- und Höchstanzahl von Workern auf die gleiche setzen.

```
aws mwaa update-environment --name MyEnvironmentName --min-workers 5 --max-workers 5
```

- e. Weitere Informationen zu den bewährten Methoden, die wir empfehlen, um die Leistung Ihrer Umgebung zu optimieren, finden Sie unter [the section called “Leistungsoptimierung für Apache Airflow”](#).
4. Wenn Ihre Aufgaben im Status „Wird ausgeführt“ stecken, können Sie die Aufgaben auch löschen oder sie als erfolgreich oder fehlgeschlagen markieren. Dadurch kann die Autoscaling-Komponente für Ihre Umgebung die Anzahl der Worker, die in Ihrer Umgebung ausgeführt werden, reduzieren. Die folgende Abbildung zeigt ein Beispiel für eine nicht behandelte.



- Wählen Sie den Kreis für die gestrandete Aufgabe und klicken Sie dann auf Löschen (wie abgebildet). Dadurch kann Amazon MWAA Mitarbeiter verkleinern. Andernfalls kann Amazon MWAA nicht feststellen, welche DAGs aktiviert oder deaktiviert sind, und kann nicht herunterskalieren, wenn es noch Aufgaben in der Warteschlange gibt.



- Weitere Informationen zum Apache Airflow-Aufgabenlebenszyklus finden Sie unter [Konzepte](#) im Apache Airflow-Referenzleitfaden.

## CLI

Im folgenden Thema werden die Fehler beschrieben, die beim Ausführen von Airflow-CLI-Befehlen in der auftreten könnenAWS Command Line Interface.

Ich sehe einen '503'-Fehler, wenn ich eine DAG in der CLI auslöse

Die Airflow-CLI wird auf dem Apache Airflow-Webserver ausgeführt, der über eine eingeschränkte Parallelität verfügt. In der Regel können maximal 4 CLI-Befehle gleichzeitig ausgeführt werden.

# Problembehandlung: Erstellen und Aktualisieren einer Amazon MWAA-Umgebung

Die Themen auf dieser Seite enthalten Fehler, die bei der Erstellung und Aktualisierung einer Amazon Managed Workflows for Apache Airflow-Umgebung auftreten können, sowie Informationen zur Behebung dieser Fehler.

## Inhalt

- [Aktualisieren von requirements.txt](#)
  - [Ich habe eine neue Version von meinem angegebenrequirements.txt und es dauert mehr als 20 Minuten, meine Umgebung zu aktualisieren](#)
- [Plug-ins](#)
  - [Unterstützt Amazon MWAA die Implementierung einer benutzerdefinierten Benutzeroberfläche?](#)
  - [Ich kann benutzerdefinierte UI-Änderungen auf dem Amazon MWAA Local Runner über Plugins implementieren, aber wenn ich versuche, dasselbe auf Amazon MWAA zu tun, sehe ich weder meine Änderungen noch irgendwelche Fehler. Warum passiert das?](#)
- [Erstellen Buckets](#)
  - [Ich kann die Option für S3 Block Public Access-Einstellungen nicht auswählen](#)
- [Erstellen der -Umgebung](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, und sie steckt im Status „Creating“ fest](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, aber der Status wird als „Erstellen fehlgeschlagen“ angezeigt](#)
  - [Ich habe versucht, eine VPC auszuwählen, und habe den Fehler „Netzwerkfehler“ erhalten](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, und habe den Fehler „Muss übergeben werden“ für den Dienst, die Partition oder die Ressource erhalten](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, und der Status wird als „Verfügbar“ angezeigt, aber wenn ich versuche, auf die Airflow-Benutzeroberfläche zuzugreifen, wird der Fehler „Empty Reply from Server“ oder „502 Bad Gateway“ angezeigt](#)
  - [Ich habe versucht, eine Umgebung zu erstellen, und mein Benutzername besteht aus einer Reihe zufälliger Charakternamen](#)
- [Umgebung aktualisieren](#)
  - [Ich habe versucht, die Umgebungsklasse zu ändern, aber das Update ist fehlgeschlagen](#)
- [Zugangsumgebung](#)

- [Ich kann nicht auf die Apache Airflow-Benutzeroberfläche zugreifen](#)

## Aktualisieren von `requirements.txt`

Im folgenden Thema werden die Fehler beschrieben, die bei der Aktualisierung Ihres auftreten können `requirements.txt`.

Ich habe eine neue Version von meinem angegebenen `requirements.txt` und es dauert mehr als 20 Minuten, meine Umgebung zu aktualisieren

Wenn es länger als zwanzig Minuten dauert, bis Ihre Umgebung eine neue Version einer `requirements.txt` Datei installiert hat, ist das Umgebungsupdate fehlgeschlagen und Amazon MWAA führt ein Rollback zur letzten stabilen Version des Container-Images durch.

1. Überprüfen Sie die Paketversionen. Wir empfehlen, immer entweder eine bestimmte Version (`=`) oder eine maximale Version (`>=`) für die Python-Abhängigkeiten in Ihrem anzugeben `requirements.txt`.
2. Überprüfen Sie die Apache Airflow-Protokolle. Wenn Sie Apache Airflow-Logs aktiviert haben, überprüfen Sie auf der [Seite Protokollgruppen in der CloudWatch Konsole, ob Ihre Protokollgruppen](#) erfolgreich erstellt wurden. Wenn Sie leere Protokolle sehen, liegt der häufigste Grund in fehlenden Berechtigungen in Ihrer Ausführungsrolle für Amazon S3 CloudWatch oder Amazon S3, wo Protokolle geschrieben werden. Weitere Informationen hierzu finden Sie unter [Ausführungsrolle](#).
3. Überprüfen Sie hier: Apache Airflow-Konfigurationsoptionen. Wenn Sie Secrets Manager verwenden, stellen Sie sicher, dass die Schlüssel-Wert-Paare, die Sie als Apache Airflow-Konfigurationsoption angegeben haben, korrekt konfiguriert wurden. Weitere Informationen hierzu finden Sie unter [the section called "Secrets Manager konfigurieren"](#).
4. Überprüfen Sie die VPC-Netzwerkconfiguration. Weitere Informationen hierzu finden Sie unter [the section called "Die Umgebung steckt fest"](#).
5. Überprüfen der Berechtigungen für Ausführungsrolle. Eine Ausführungsrolle ist eine AWS Identity and Access Management (IAM-) Rolle mit einer Berechtigungsrichtlinie, die Amazon MWAA die Berechtigung erteilt, die Ressourcen anderer AWS Dienste (wie Amazon S3 CloudWatch, Amazon SQS, Amazon ECR) in Ihrem Namen aufzurufen. Ihr [vom Kunden verwalteter Schlüssel](#) oder [Ihr AWS eigener Schlüssel](#) muss ebenfalls Zugriff erhalten. Weitere Informationen hierzu finden Sie unter [Ausführungsrolle](#).

6. Informationen zum Ausführen eines Skripts zur Fehlerbehebung, das die Amazon VPC-Netzwerkeinrichtung und -konfiguration für Ihre Amazon MWAA-Umgebung überprüft, finden Sie im Skript „[Umgebung verifizieren](#)“ in den AWS Support Tools unter GitHub.

## Plug-ins

Das folgende Thema beschreibt Probleme, die bei der Konfiguration oder Aktualisierung von Apache Airflow-Plug-ins auftreten können.

### Unterstützt Amazon MWAA die Implementierung einer benutzerdefinierten Benutzeroberfläche?

Ab Apache Airflow v2.2.2 unterstützt Amazon MWAA die Installation von Plug-ins auf dem Apache Airflow-Webserver und die Implementierung einer benutzerdefinierten Benutzeroberfläche. Wenn in Ihrer Amazon MWAA-Umgebung Apache Airflow v2.0.2 oder älter ausgeführt wird, können Sie keine benutzerdefinierte Benutzeroberfläche implementieren.

Weitere Informationen zur Versionsverwaltung und zum Upgrade Ihrer vorhandenen Umgebungen finden Sie unter [Versionen](#).

Ich kann benutzerdefinierte UI-Änderungen auf dem [Amazon MWAA Local Runner](#) über Plug-ins implementieren, aber wenn ich versuche, dasselbe auf Amazon MWAA zu tun, sehe ich weder meine Änderungen noch irgendwelche Fehler. Warum passiert das?

Der Amazon MWAA Local Runner hat alle Apache Airflow-Komponenten in einem Image gebündelt, sodass Sie benutzerdefinierte UI-Plug-in-Änderungen anwenden können.

## Erstellen Buckets

Im folgenden Thema werden die Fehler beschrieben, die beim Erstellen eines Amazon S3 S3-Buckets auftreten können.

Ich kann die Option für S3 Block Public Access-Einstellungen nicht auswählen

Die [Ausführungsrolle](#) für Ihre Amazon MWAA-Umgebung benötigt die Genehmigung für die `GetBucketPublicAccessBlock` Aktion im Amazon S3 S3-Bucket, um zu überprüfen, ob der Bucket den öffentlichen Zugriff blockiert hat. Wir empfehlen die folgenden Schritte:



1. Folgen Sie den Schritten, um [Ihrer Ausführungsrolle eine JSON-Richtlinie zuzuordnen](#).
2. Fügen Sie die folgende JSON-Richtlinie an:

```
{
 "Effect": "Allow",
 "Action": [
 "s3:GetObject*",
 "s3:GetBucket*",
 "s3:List*"
],
 "Resource": [
 "arn:aws:s3:::YOUR_S3_BUCKET_NAME",
 "arn:aws:s3:::YOUR_S3_BUCKET_NAME/*"
]
}
```

Ersetzen Sie die Beispiel-Platzhalter in *YOUR\_S3\_BUCKET\_NAME* durch Ihren Amazon S3 S3-Bucket-Namen, z. B. *my-mwaa-unique-s3-bucket-name*.

3. Informationen zum Ausführen eines Skripts zur Fehlerbehebung, das die Amazon VPC-Netzwerkeinrichtung und -konfiguration für Ihre Amazon MWAA-Umgebung überprüft, finden Sie im Skript „[Umgebung verifizieren](#)“ in den AWS Support Tools unter GitHub.

## Erstellen der -Umgebung

Im folgenden Thema werden die Fehler beschrieben, die beim Erstellen einer Umgebung auftreten können.

Ich habe versucht, eine Umgebung zu erstellen, und sie steckt im Status „Creating“ fest

Wir empfehlen die folgende Schritte:

1. Überprüfen Sie das VPC-Netzwerk mit öffentlichem Routing. Wenn Sie eine Amazon VPC mit Internetzugang verwenden, überprüfen Sie Folgendes:
  - Dass Ihre Amazon VPC so konfiguriert ist, dass sie Netzwerkverkehr zwischen den verschiedenen AWS Ressourcen zulässt, die von Ihrer Amazon MWAA-Umgebung verwendet werden, wie in definiert [the section called “Informationen zu Netzwerken”](#). Beispielsweise muss Ihre VPC-Sicherheitsgruppe entweder den gesamten Datenverkehr

in einer Regel zur Selbstreferenzierung zulassen oder optional den Portbereich für den HTTPS-Portbereich 443 und einen TCP-Portbereich 5432 angeben.

- Überprüfen Sie das VPC-Netzwerk mit privatem Routing. Wenn Sie eine Amazon VPC ohne Internetzugang verwenden, überprüfen Sie Folgendes:
  - Dass Ihre Amazon VPC so konfiguriert ist, dass sie Netzwerkverkehr zwischen den verschiedenen AWS Ressourcen für Ihre Amazon MWAA-Umgebung zulässt, wie in [definiert](#) [the section called “Informationen zu Netzwerken”](#). Beispielsweise dürfen Ihre beiden privaten Subnetze weder über eine Routentabelle zu einem NAT-Gateway (oder einer NAT-Instance) noch über ein Internet-Gateway verfügen.
- Informationen zum Ausführen eines Skripts zur Fehlerbehebung, das die Amazon VPC-Netzwerkeinrichtung und -konfiguration für Ihre Amazon MWAA-Umgebung überprüft, finden Sie im Skript „[Umgebung verifizieren](#)“ in den AWS Support Tools unter GitHub.

Ich habe versucht, eine Umgebung zu erstellen, aber der Status wird als „Erstellen fehlgeschlagen“ angezeigt

Wir empfehlen die folgende Schritte:

- Überprüfen Sie die VPC-Netzwerkkonfiguration. Weitere Informationen hierzu finden Sie unter [the section called “Die Umgebung steckt fest”](#).
- Überprüfen Sie die Benutzerberechtigungen. Amazon MWAA führt einen Probelauf mit den Anmeldeinformationen eines Benutzers durch, bevor eine Umgebung erstellt wird. Ihr AWS Konto verfügt möglicherweise nicht über die Berechtigung in AWS Identity and Access Management (IAM), einige Ressourcen für eine Umgebung zu erstellen. Wenn Sie beispielsweise den Apache Airflow-Zugriffsmodus für privates Netzwerk gewählt haben, muss Ihrem AWS Konto von Ihrem Administrator Zugriff auf die [Full Console Access Amazon MWAA-Zugriffskontrollrichtlinie](#) für Ihre Umgebung gewährt worden sein, die es Ihrem Konto ermöglicht, VPC-Endpunkte zu erstellen.
- Überprüfen der Berechtigungen für Ausführungsrolle. Eine Ausführungsrolle ist eine AWS Identity and Access Management (IAM-) Rolle mit einer Berechtigungsrichtlinie, die Amazon MWAA die Berechtigung erteilt, die Ressourcen anderer AWS Dienste (wie Amazon S3 CloudWatch, Amazon SQS, Amazon ECR) in Ihrem Namen aufzurufen. Ihr [vom Kunden verwalteter Schlüssel](#) oder [Ihr AWS eigener Schlüssel](#) muss ebenfalls Zugriff erhalten. Weitere Informationen hierzu finden Sie unter [Ausführungsrolle](#).
- Überprüfen Sie die Apache Airflow-Protokolle. Wenn Sie Apache Airflow-Logs aktiviert haben, überprüfen Sie auf der [Seite Protokollgruppen in der CloudWatch Konsole, ob Ihre](#)

[Protokollgruppen](#) erfolgreich erstellt wurden. Wenn Sie leere Protokolle sehen, liegt der häufigste Grund in fehlenden Berechtigungen in Ihrer Ausführungsrolle für Amazon S3CloudWatch oder Amazon S3, wo Protokolle geschrieben werden. Weitere Informationen hierzu finden Sie unter [Ausführungsrolle](#).

- Informationen zum Ausführen eines Skripts zur Fehlerbehebung, das die Amazon VPC-Netzwerkeinrichtung und -konfiguration für Ihre Amazon MWAA-Umgebung überprüft, finden Sie im Skript „[Umgebung verifizieren](#)“ in den AWS Support Tools unter GitHub.
- Wenn Sie eine Amazon VPC ohne Internetzugang verwenden, stellen Sie sicher, dass Sie einen Amazon S3-Gateway-Endpoint erstellt und Amazon ECR die erforderlichen Mindestberechtigungen für den Zugriff auf Amazon S3 erteilt haben. Weitere Informationen zur Erstellung eines Amazon S3 S3-Gateway-Endpoints finden nachfolgend:
  - [Erstellen eines Amazon VPC-Netzwerks ohne Internetzugang](#)
  - [Erstellen Sie den Amazon S3 S3-Gateway-Endpoint](#) im Amazon Elastic Container Registry-Benutzerhandbuch

Ich habe versucht, eine VPC auszuwählen, und habe den Fehler „Netzwerkfehler“ erhalten

Wir empfehlen die folgende Schritte:

- Wenn beim Erstellen Ihrer Umgebung der Fehler „Netzwerkfehler“ angezeigt wird, wenn Sie versuchen, eine Amazon VPC auszuwählen, schalten Sie alle laufenden In-Browser-Proxys aus, und versuchen Sie es erneut.

Ich habe versucht, eine Umgebung zu erstellen, und habe den Fehler „Muss übergeben werden“ für den Dienst, die Partition oder die Ressource erhalten

Wir empfehlen die folgende Schritte:

- Möglicherweise erhalten Sie diesen Fehler, weil die URI, die Sie für Ihren Amazon S3 S3-Bucket angegeben haben, ein '/' am Ende der URI enthält. Wir empfehlen, das '/' im Pfad zu entfernen. Der Wert sollte das folgende Format aufweisen:

```
s3://your-bucket-name
```

Ich habe versucht, eine Umgebung zu erstellen, und der Status wird als „Verfügbar“ angezeigt, aber wenn ich versuche, auf die Airflow-Benutzeroberfläche zuzugreifen, wird der Fehler „Empty Reply from Server“ oder „502 Bad Gateway“ angezeigt

Wir empfehlen die folgende Schritte:

1. Überprüfen Sie die Konfiguration der VPC-Sicherheitsgruppen. Weitere Informationen hierzu finden Sie unter [the section called “Die Umgebung steckt fest”](#).
2. Vergewissern Sie sich, dass alle Apache Airflow-Pakete, die Sie in der Liste aufgeführt haben, der Apache Airflow-`Versionrequirements.txt` entsprechen, die Sie auf Amazon MWAA ausführen. Weitere Informationen hierzu finden Sie unter [Python-Abhängigkeiten installieren](#).
3. Informationen zum Ausführen eines Skripts zur Fehlerbehebung, das die Amazon VPC-Netzwerkeinrichtung und -konfiguration für Ihre Amazon MWAA-Umgebung überprüft, finden Sie im Skript „[Umgebung verifizieren](#)“ in den AWS Support Tools unter GitHub.

Ich habe versucht, eine Umgebung zu erstellen, und mein Benutzername besteht aus einer Reihe zufälliger Charakternamen

- Apache Airflow hat maximal 64 Zeichen für Benutzernamen. Wenn Ihre AWS Identity and Access Management (IAM) -Rolle diese Länge überschreitet, wird ein Hash-Algorithmus verwendet, um sie zu reduzieren und gleichzeitig einzigartig zu bleiben.

## Umgebung aktualisieren

Im folgenden Thema werden die Fehler beschrieben, die beim Aktualisieren einer Umgebung auftreten können.

Ich habe versucht, die Umgebungsklasse zu ändern, aber das Update ist fehlgeschlagen

Wenn Sie Ihre Umgebung auf eine andere Umgebungsklasse aktualisieren (z. B. indem Sie einem `mw1.medium` zu einer ändern `mw1.small`) und die Anforderung zur Aktualisierung Ihrer Umgebung fehlgeschlagen ist, wechselt der Umgebungsstatus in einen `UPDATE_FAILED` Zustand und die Umgebung wird auf die vorherige stabile Version einer Umgebung zurückgesetzt und entsprechend abgerechnet.

Wir empfehlen die folgende Schritte:

1. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.
2. Informationen zum Ausführen eines Skripts zur Fehlerbehebung, das die Amazon VPC-Netzwerkeinrichtung und -konfiguration für Ihre Amazon MWAA-Umgebung überprüft, finden Sie im Skript „[Umgebung verifizieren](#)“ in den AWS Support Tools unter GitHub.

## Zugangsumgebung

Im folgenden Thema werden die Fehler beschrieben, die beim Zugriff auf eine Umgebung auftreten können.

### Ich kann nicht auf die Apache Airflow-Benutzeroberfläche zugreifen

Wir empfehlen die folgende Schritte:

1. Überprüfen Sie die Benutzerberechtigungen. Möglicherweise wurde Ihnen kein Zugriff auf eine Berechtigungsrichtlinie gewährt, mit der Sie die Apache Airflow-Benutzeroberfläche anzeigen können. Weitere Informationen hierzu finden Sie unter [the section called “Zugreifen auf eine Amazon MWAA-Umgebung”](#).
2. Überprüfen des Netzwerkzugriffs. Dies kann daran liegen, dass Sie den Zugriffsmodus Privates Netzwerk ausgewählt haben. Wenn die URL Ihrer Apache Airflow-Benutzeroberfläche das folgende Format hat `387fbcn-8dh4-9hfj-0dnd-834jhdfb-vpce.c10.us-west-2.airflow.amazonaws.com`, bedeutet dies, dass Sie privates Routing für Ihren Apache Airflow-Webserver verwenden. Sie können entweder den Apache Airflow-Zugriffsmodus auf den öffentlichen Netzwerkzugriffsmodus aktualisieren oder einen Mechanismus für den Zugriff auf den VPC-Endpunkt für Ihren Apache Airflow-Webserver erstellen. Weitere Informationen hierzu finden Sie unter [the section called “Verwaltung des Zugriffs auf VPC-Endpunkte”](#).

## Problembehandlung: CloudWatch Protokolle und CloudTrail Fehler

Die Themen auf dieser Seite enthalten Lösungen für Amazon CloudWatch Logs und AWS CloudTrail Fehler, die in einer Umgebung mit Amazon Managed Workflows für Apache Airflow auftreten können.

### Inhalt

- [Logs \(Protokolle\)](#)
  - [Ich kann meine Aufgabenprotokolle nicht sehen, oder ich habe die Fehlermeldung „Remote-Protokoll aus Cloudwatch log\\_group lesen“ erhalten](#)

- [Aufgaben schlagen fehl, wenn keine Protokolle vorhanden sind](#)
- [Ich sehe einen Fehler " in ResourceAlreadyExistsException CloudTrail](#)
- [Ich sehe den Fehler „Ungültige Anfrage“ in CloudTrail](#)
- [In den Apache Airflow-Protokollen wird die Meldung „Eine 64-Bit-Oracle-Client-Bibliothek kann nicht gefunden werden: „libcintsh.so: cannot open shared object file: No such file or directory“ angezeigt](#)
- [Ich sehe in meinen Scheduler-Protokollen die Meldung psychopg2, dass der Server die Verbindung unerwartet geschlossen hat](#)
- [Ich sehe in meinen DAG-Verarbeitungsprotokollen die Meldung „Der Executor meldet die Aufgabeninstanz %s als abgeschlossen \(%s\), obwohl für die Aufgabe angegeben ist, dass sie %s ist“](#)
- [Ich erhalte die Meldung „Konnten keine Remote-Protokolle von log\\_group lesen: airflow-`{\*environmentName}` -Task log\\_stream: `\* {\*DAG\_ID} / \* {\*TASK\_ID} / \* {\*time} / \* {\*n} .log.`“ in meinen Aufgabenprotokollen](#)

## Logs (Protokolle)

Im folgenden Thema werden die Fehler beschrieben, die beim Anzeigen von Apache Airflow-Protokollen auftreten können.

Ich kann meine Aufgabenprotokolle nicht sehen, oder ich habe die Fehlermeldung „Remote-Protokoll aus Cloudwatch log\_group lesen“ erhalten

Amazon MWAA hat Apache Airflow so konfiguriert, dass Protokolle direkt von und nach Amazon Logs gelesen und geschrieben werden. CloudWatch Wenn ein Worker eine Aufgabe nicht startet oder keine Protokolle schreibt, wird der folgende Fehler angezeigt:

```
*** Reading remote log from Cloudwatch log_group: airflow-environmentName-Task
log_stream: DAG_ID/TASK_ID/timestamp/n.log.Could not read remote logs from log_group:
airflow-environmentName-Task log_stream: DAG_ID/TASK_ID/time/n.log.
```

- Wir empfehlen die folgende Schritte:
  - a. Stellen Sie sicher, dass Sie Aufgabenprotokolle auf der INFO Ebene für Ihre Umgebung aktiviert haben. Weitere Informationen finden Sie unter [Airflow-Protokolle in Amazon anzeigen CloudWatch](#).

- b. Stellen Sie sicher, dass die [Umgebungsausführungsrolle](#) über die richtigen Berechtigungsrichtlinien verfügt.
- c. Stellen Sie sicher, dass Ihr Operator oder Ihre Aufgabe ordnungsgemäß funktioniert, über ausreichende Ressourcen zum Parsen der DAG verfügt und über die entsprechenden Python-Bibliotheken zum Laden verfügt. Um zu überprüfen, ob Sie über die richtigen Abhängigkeiten verfügen, versuchen Sie, Importe zu entfernen, bis Sie den gefunden haben, der das Problem verursacht. Wir empfehlen, Ihre Python-Abhängigkeiten mit dem [Amazon MWAA Local-Runner-Tool](#) zu testen.

## Aufgaben schlagen fehl, wenn keine Protokolle vorhanden sind

Wenn Aufgaben in einem Workflow fehlschlagen und Sie keine Protokolle für die fehlgeschlagenen Aufgaben finden können, überprüfen Sie, ob Sie den `queue` Parameter in Ihren Standardargumenten festlegen, wie im Folgenden gezeigt.

```
from airflow import DAG
from airflow.operators.bash_operator import BashOperator
from airflow.utils.dates import days_ago

Setting queue argument to default.
default_args = {
 "start_date": days_ago(1),
 "queue": "default"
}

with DAG(dag_id="any_command_dag", schedule_interval=None, catchup=False,
 default_args=default_args) as dag:
 cli_command = BashOperator(
 task_id="bash_command",
 bash_command="{{ dag_run.conf['command'] }}"
)
```

Um das Problem zu beheben, entfernen Sie es `queue` aus Ihrem Code und rufen Sie die DAG erneut auf.

## Ich sehe einen Fehler " in ResourceAlreadyExistsException CloudTrail

```
"errorCode": "ResourceAlreadyExistsException",
 "errorMessage": "The specified log stream already exists",
```

```
"requestParameters": {
 "logGroupName": "airflow-MyAirflowEnvironment-DAGProcessing",
 "logStreamName": "scheduler_cross-account-eks.py.log"
}
```

Bestimmte Python-Anforderungen, z. B. ein `apache-airflow-backport-providers-amazon` Rollback der `watchtower` Bibliothek, mit der Amazon MWAAs kommuniziert, CloudWatch auf eine ältere Version. Wir empfehlen die folgende Schritte:

- Fügen Sie die folgende Bibliothek zu Ihrer `requirements.txt` hinzu

```
watchtower==1.0.6
```

## Ich sehe den Fehler „Ungültige Anfrage“ in CloudTrail

```
Invalid request provided: Provided role does not have sufficient permissions for s3
location airflow-xxx-xxx/dags
```

Wenn Sie eine Amazon MWAAs-Umgebung und einen Amazon S3 S3-Bucket mit derselben AWS CloudFormation Vorlage erstellen, müssen Sie Ihrer AWS CloudFormation Vorlage einen `DependsOn` Abschnitt hinzufügen. Die beiden Ressourcen (MWAAs-Umgebung und MWAAs-Ausführungsrichtlinie) haben eine Abhängigkeit von. AWS CloudFormation Wir empfehlen die folgende Schritte:

- Fügen Sie Ihrer Vorlage die folgende **DependsOn** Anweisung hinzu. AWS CloudFormation

```
...
 MaxWorkers: 5
 NetworkConfiguration:
 SecurityGroupIds:
 - !GetAtt SecurityGroup.GroupId
 SubnetIds: !Ref subnetIds
 WebserverAccessMode: PUBLIC_ONLY
 DependsOn: MwaaExecutionPolicy

 MwaaExecutionPolicy:
 Type: AWS::IAM::ManagedPolicy
 Properties:
 Roles:
 - !Ref MwaaExecutionRole
```



```
PolicyDocument:
 Version: 2012-10-17
 Statement:
 - Effect: Allow
 Action: airflow:PublishMetrics
 Resource:
 ...
```

Ein Beispiel finden Sie unter [Schnellstartanleitung für Amazon Managed Workflows für Apache Airflow](#).

In den Apache Airflow-Protokollen wird die Meldung „Eine 64-Bit-Oracle-Client-Bibliothek kann nicht gefunden werden: „libcIntsh.so: cannot open shared object file: No such file or directory“ angezeigt

- Wir empfehlen die folgende Schritte:
  - Wenn Sie Apache Airflow v2 verwenden, fügen Sie es `core.lazy_load_plugins : False` als Apache Airflow-Konfigurationsoption hinzu. Weitere Informationen finden Sie unter [Verwenden von Konfigurationsoptionen zum Laden von Plugins in 2](#).

Ich sehe in meinen Scheduler-Protokollen die Meldung `psycopg2`, dass der Server die Verbindung unerwartet geschlossen hat

Wenn Sie einen Fehler wie den folgenden sehen, sind Ihrem Apache Airflow Scheduler möglicherweise die Ressourcen ausgegangen.

```
2021-06-14T10:20:24.581-05:00 sqlalchemy.exc.OperationalError:
(psycopg2.OperationalError) server closed the connection unexpectedly
2021-06-14T10:20:24.633-05:00 This probably means the server terminated abnormally
2021-06-14T10:20:24.686-05:00 before or while processing the request.
```

Wir empfehlen die folgende Schritte:

- Erwägen Sie ein Upgrade auf Apache Airflow v2.0.2, mit dem Sie bis zu 5 Scheduler angeben können.

Ich sehe in meinen DAG-Verarbeitungsprotokollen die Meldung „Der Executor meldet die Aufgabeninstanz %s als abgeschlossen (%s), obwohl für die Aufgabe angegeben ist, dass sie %s ist“

Wenn Sie einen Fehler wie den folgenden sehen, haben Ihre lang andauernden Aufgaben möglicherweise das Zeitlimit für Aufgaben auf Amazon MWAA erreicht. Amazon MWAA hat für jede einzelne Airflow-Aufgabe ein Limit von 12 Stunden festgelegt, um zu verhindern, dass Aufgaben in der Warteschlange hängen bleiben und Aktivitäten wie Autoscaling blockiert werden.

```
Executor reports task instance %s finished (%s) although the task says its %s. (Info: %s) Was the task killed externally
```

Wir empfehlen die folgende Schritte:

- Erwägen Sie, die Aufgabe in mehrere, kürzer laufende Aufgaben aufzuteilen. Airflow verwendet in der Regel ein Modell, bei dem die Bediener asynchron arbeiten. Es ruft Aktivitäten auf externen Systemen auf, und Apache Airflow Sensors fragt ab, wann der Vorgang abgeschlossen ist. Wenn ein Sensor ausfällt, kann er sicher erneut versucht werden, ohne dass die Funktionalität des Bedieners beeinträchtigt wird.

Ich erhalte die Meldung „Konnten keine Remote-Protokolle von log\_group lesen: airflow-`{*environmentName}`-Task log\_stream: `{*DAG_ID}` / `{*TASK_ID}` / `{*time}` / `{*n}` .log.“ in meinen Aufgabenprotokollen

Wenn Sie einen Fehler wie den folgenden sehen, enthält die Ausführungsrolle für Ihre Umgebung möglicherweise keine Berechtigungsrichtlinie zum Erstellen von Protokollstreams für Aufgabenprotokolle.

```
Could not read remote logs from log_group: airflow-{*environmentName}-Task log_stream:{*DAG_ID}/{*TASK_ID}/{*time}/{*n}.log.
```

Wir empfehlen die folgende Schritte:

- Ändern Sie die Ausführungsrolle für Ihre Umgebung mithilfe einer der Beispielrichtlinien unter [the section called “Ausführungsrolle”](#).

Möglicherweise haben Sie in Ihrer `requirements.txt` Datei auch ein Provider-Paket angegeben, das mit Ihrer Apache Airflow-Version nicht kompatibel ist. Wenn Sie beispielsweise Apache Airflow v2.0.2 verwenden, haben Sie möglicherweise ein Paket angegeben, z. B. das [apache-airflow-providers-databricks](#) Paket, das nur mit Airflow 2.1+ kompatibel ist.

Wir empfehlen die folgende Schritte:

1. Wenn Sie Apache Airflow v2.0.2 verwenden, ändern Sie die Datei und fügen Sie sie hinzu.  
`requirements.txt` `apache-airflow[databricks]` Dadurch wird die richtige Version des Databricks-Pakets installiert, die mit Apache Airflow v2.0.2 kompatibel ist.
2. Testen Sie Ihre DAGs, benutzerdefinierten Plugins und Python-Abhängigkeiten lokal mit dem [aws-mwaa-local-runner](#) on GitHub.

# Amazon MWAA-Dokumentenverlauf

In der folgenden Tabelle werden wichtige Ergänzungen der Amazon MWAA-Service-Dokumentation beschrieben, die im November 2020 beginnen. Abonnieren Sie den RSS-Feed, um Benachrichtigungen über Aktualisierungen dieser Dokumentation zu erhalten.

| Änderung                                                                                                            | Beschreibung                                                                                                                                                                                                                                                                                                                                                                                                | Datum        |
|---------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------|
| <a href="#">Amazon MWAA unterstützt die automatische Skalierung von Webservern und die Apache Airflow REST API.</a> | <p>Amazon MWAA unterstützt jetzt die automatische Skalierung von Webservern sowie die Möglichkeit, auf die Apache Airflow REST API zuzugreifen und diese zu verwenden.</p> <ul style="list-style-type: none"><li>• <a href="#">the section called “Automatische Skalierung von Webservern konfigurieren”</a></li><li>• <a href="#">the section called “Verwenden der Apache Airflow REST API”</a></li></ul> | 16. Mai 2024 |
| <a href="#">Die Beschreibung des Verhaltens der automatischen Skalierung wurde verbessert</a>                       | <p>Das folgende Thema wurde aktualisiert, um das neue automatische Skalierungsverhalten von Amazon MWAA widerzuspiegeln, wenn Mitarbeiter neue Aufgaben übernehmen, während Fargate-Mitarbeiter nach unten skalieren.</p> <ul style="list-style-type: none"><li>• <a href="#">the section called “Konfiguration von Worker Auto Scaling”</a></li></ul>                                                      | 10. Mai 2024 |

## [Support für größere Instance-Größen](#)

Amazon MWAA unterstützt jetzt zwei Optionen mit größerer Instance-Größe für größere Workloads: und `mw1.xlarge` `mw1.2xlarge`

16. April 2024

- [the section called “Funktionen der Umgebung”](#)

## [Neue Apache Airflow-Version](#)

Amazon MWAA unterstützt jetzt Apache Airflow v2.8.1. Dieses Update enthält Informationen zu aktualisierten Anbieterpaketen und Details zur Verwendung von Apache Airflow v2.8.1 auf Amazon MWAA.

22. Februar 2024

- [Versionen](#)
- [the section called “Anbieterpakete für Apache Airflow v2.8.1-Verbindungen”](#)

## [Support für gemeinsam genutzte Amazon VPC](#)

Amazon MWAA unterstützt die Erstellung einer kontoübergreifenden Umgebung für Organisationen, die Amazon OpenSearch Service zur Verwaltung von Amazon MWAA-Ressourcen mithilfe einer zentralen gemeinsamen Amazon VPC in einem Eigentümerkonto verwenden. Im Rahmen dieser Markteinführung können Sie mit Amazon MWAA wählen, ob Sie Ihre eigenen Amazon VPC-Endpoints erstellen und verwalten möchten.

15. November 2023

- [the section called “Verwalten Ihrer eigenen Amazon-VPC-Endpoints”](#)

## [Neue Apache Airflow-Version](#)

Amazon MWAA unterstützt jetzt Apache Airflow v2.7.2. Dieses Update enthält Informationen zu aktualisierten Anbieterpaketen und Details zur Verwendung von Apache Airflow v2.7.2 auf Amazon MWAA.

6. November 2023

- [Versionen](#)
- [the section called “Anbieterpakete für Apache Airflow v2.7.2-Verbindungen”](#)

## [Neue Apache Airflow-Version](#)

Amazon MWAA unterstützt jetzt Apache Airflow v2.6.3. Dieses Update enthält Informationen zu aktualisierten Anbieterpaketen und Details zur Verwendung von Apache Airflow v2.6.3 auf Amazon MWAA.

9. August 2023

- [Versionen](#)
- [the section called “Anbieterpakete für Apache Airflow v2.6.3-Verbindungen”](#)

## [Informationen zur veralteten Version](#)

Das Thema zur veralteten Version wurde aktualisiert und enthält nun Hinweise zu veralteten Versionen und Zeitpläne für Apache Airflow v2.0.2 und Apache Airflow v2.2.2.

31. Juli 2023

- [the section called “Veraltete Versionen von Apache Airflow”](#)

## Neue Themen und Anwendungsfälle

Amazon MWAA unterstützt kleinere Versions-Upgrades. Dieses Update umfasst das folgende neue Thema, in dem beschrieben wird, wie Sie die Umgebung aktualisieren und sicherstellen, dass Ihre Workflow-Ressourcen mit der Version von Apache Airflow kompatibel sind, auf die Sie aktualisieren:

5. Juni 2023

- [the section called “Aktualisierung der Version”](#)

## Das Thema wurde aktualisiert

Aktualisierte, vom Kunden verwaltete IAM-Richtlinien, die einem Benutzer vollen Konsolen- und API-Zugriff auf Amazon MWAA gewähren. Das Update beschreibt, warum Sie die Erlaubnis für `iam:PassRole` erteilen müssen, damit ein Benutzer Rollen an Amazon MWAA weitergeben kann. Amazon MWAA verwendet diese Berechtigungen, um Aktionen im Namen eines Benutzers durchzuführen.

12. April 2023

- [the section called “Zugreifen auf eine Amazon MWAA-Umgebung”](#)



## Neue Leitlinien

Das Thema zur Konfiguration AWS Secrets Manager als Backend für Amazon MWAA wurde aktualisiert und enthält nun Anleitungen zur Verwendung von Suchmustern. Mithilfe von Suchmustern können Sie die Geheimnisse, nach denen Apache Airflow sucht, einschränken und die Anzahl der API-Aufrufe reduzieren, die Amazon MWAA an Secrets Manager durchführt, um Verbindungen und Variablen abzurufen. Dies reduziert die Kosten, die mit der Verwendung von Secrets Manager als Backend verbunden sind.

12. April 2023

- [Erstellen Sie das Secrets Manager Backend als Apache Airflow-Konfigurationsoption](#)

## [Neue Apache Airflow-Version](#)

Amazon MWAA unterstützt jetzt Apache Airflow v2.5.1. Dieses Update enthält Informationen zu aktualisierten Anbieterpaketen und Details zur Verwendung von Apache Airflow v2.5.1 auf Amazon MWAA.

11. April 2023

- [Versionen](#)
- [the section called “Anbieterpakete für Apache Airflow v2.5.1-Verbindungen”](#)

## [Neue Themen und Anwendungsfälle](#)

Es wurde ein neues Thema zur Verwendung eines Startskripts mit einer Amazon MWAA-Umgebung hinzugefügt. In diesem Thema wird beschrieben, wie Sie ein Startskript für eine bestehende Umgebung konfigurieren, es zur Installation von Linux-Laufzeiten verwenden und Umgebungsvariablen festlegen.

03. April 2023

- [the section called “Verwenden eines Startskripts”](#)

[Der Abschnitt über den Zugriff auf private Webserver wurde aktualisiert](#)

Das folgende Thema zum Zugriff auf private Webserver wurde aktualisiert. Das Update stellt klar, dass Sie in Umgebungen mit privatem Webserverzugriff ein Python-Wheel-Archiv (.whl) verwenden müssen, um Abhängigkeiten zu packen und zu installieren.

24. Februar 2023

- [Zugriffsmodus für private Webserver](#)

## [Es wurden Informationen zu veralteten Apache Airflow-Versionen hinzugefügt](#)

Das Thema [Versionen](#) wurde mit neuen Informationen darüber aktualisiert, wie Amazon MWAA mit veralteten Apache Airflow-Versionen umgegangen ist. Ein Abschnitt über das Upgrade auf eine neuere Version von Apache Airflow und ein Abschnitt, in dem die Änderungen zwischen Apache Airflow v1 und Apache Airflow v2 beschrieben wurden, wurden entfernt. Weitere Informationen zur Migration auf eine neue Version von Apache Airflow finden Sie im [Amazon MWAA-Migrationshandbuch](#).

17. Februar 2023

- [the section called “Veraltete Versionen von Apache Airflow”](#)
- [the section called “Unterstützung der Apache Airflow-Version und häufig gestellte Fragen”](#)

## [Korrekturen bei den Amazon MWAA-Container-Metriken](#)

20. Januar 2023

Das Thema Container-Metriken wurde aktualisiert und eine Reihe fehlerhafter Metriken entfernt, die in der Dimension nicht vorhanden waren. `Cluster` Es wurde ein zusätzlicher Abschnitt hinzugefügt, in dem beschrieben wird, wie Sie die Anzahl der zusätzlichen Worker, die eine Umgebung zu einem bestimmten Zeitpunkt nutzt, auswerten können, indem Sie die `CPUUtilization` oder die `MemoryUtilization` Metrik für die `AdditionalWorker` Komponente grafisch darstellen und den Statistiktyp auf `Sample Count` festlegen.

- [the section called “Bewertung der Anzahl zusätzlicher Worker- und Webserver-Container”](#)

## Neue Apache Airflow-Version

Amazon MWAA unterstützt jetzt Apache Airflow v2.4.3. Dieses Update enthält Informationen zu aktualisierten Anbieterpaketen, Details zur Verwendung von Apache Airflow v2.4.3 auf Amazon MWAA und konsolidierte Informationen darüber, welche Funktionen in jeder Apache Airflow-Version auf Amazon MWAA unterstützt werden.

5. Januar 2023

- [Versionen](#)
- [the section called “Anbieterpakete für Apache Airflow v2.4.3-Verbindungen”](#)

[Das Thema zur serviceve  
rknüpften Rolle wurde aktualisi  
ert](#)

Aktualisierte Informationen über die serviceve rknüpfte Rolle, die Amazon MWAA verwendet, um AWS Ressourcen in Ihrem Namen zu erstellen und zu verwalten, einschließlich Informationen darüber, wie Sie die serviceve rknüpfte Rolle löschen können, wenn Sie sie nicht mehr benötigen. Dazu gehört eine aktualisierte Berechtigungsrichtlinie für dienstbezogene Rollen, die es Amazon MWAA ermöglicht, zusätzliche CloudWatch Metriken unter dem Namespace zu veröffentlichen. AWS/MWAA

18. November 2022

- [the section called “Servicegebundene Rolle”](#)

## Neues Thema zu Servicemetriken

Es wurde ein neues Thema hinzugefügt, das Servicemetriken beschreibt, die von Amazon MWAA unter dem AWS/MWAA Namespace ausgegeben werden. Dazu gehören Planer, Worker und Webserver für Amazon ECS-Cluster-Metriken, Amazon SQS-Metriken für die Warteschlangen, die es Amazon MWAA ermöglichen, Scheduler und Worker zu entkoppeln, sowie Amazon RDS-Metriken für die Metadatendatenbank.

18. November 2022

- [the section called “Container-, Warteschlangen- und Datenbank-Metriken”](#)

## Neues Thema

Es wurde eine neue Anleitung zum Ändern einer Einschränkungsdatei hinzugefügt, um neue Versionen von Anbieterpaketen zur Verwendung mit Ihrer Amazon MWAA-Umgebung anzugeben.

18. November 2022

- [the section called “Angabe neuerer Anbieterpakete”](#)



## [Der FAQ-Eintrag wurde aktualisiert](#)

Aktualisierte Informationen zur 15. November 2022  
HIPAA-Eignung von Amazon  
MWAA.

- [the section called “Compliance mit HIPAA”](#)

## [Neues Thema](#)

Es wurde ein neues Thema 21. Oktober 2022  
zur Verwendung von  
[aws:SourceArn](#) Kontextsc  
hlüsseln für [aws:SourceAccount](#) globale Bedingun  
gen in einer Vertrauensrichtlin  
ie für Amazon MWAA-Ausf  
ührungsrollen hinzugefügt,  
um zu verhindern, dass der  
stellvertretende Mitarbeiter  
dienstübergreifend verwirrt  
wird.

- [the section called “Serviceübergreifende Confused-Deputy-Prävention”](#)

## [Neuer Beispielcode](#)

Es wurden aktualisierte 13. September 2022  
Anweisungen und ein DAG-  
Codebeispiel hinzugefügt,  
in das benutzerdefinierte  
Metriken auf Betriebss  
systemebene geschrieben  
werden CloudWatch.

- [the section called “Verwenden einer DAG zum Schreiben benutzerdefinierter Metriken”](#)

### [Neuer Beispielcode](#)

Es wurden aktualisierte Anweisungen und ein neues AWS Lambda Python-Codebeispiel hinzugefügt, das ein Apache Airflow CLI-Token abrufen und dann eine DAG in einer angegebenen Amazon MWAA-Umgebung aufrufen.

12. September 2022

- [the section called “Aufrufen von DAGs mit Lambda”](#)

### [Neue Architekturdiagramme](#)

Es wurden neue Architekturdiagramme hinzugefügt, die eine Amazon MWAA-Umgebung mit einem öffentlichen und einem privaten Webserver demonstrieren.

12. September 2022

- [the section called “Apache Airflow-Zugriffsmodi”](#)

### [Neuer Beispielcode](#)

Es wurden aktualisierte Anweisungen und ein neues DAG-Codebeispiel hinzugefügt, das ein Apache Airflow CLI-Token abrufen und dann eine andere DAG in einer anderen Amazon MWAA-Umgebung aufrufen.

16. August 2022

- [the section called “DAGs in verschiedenen Umgebungen aufrufen”](#)

[Neuer Beispielcode](#)

Es wurden aktualisierte Anweisungen und eine neue DAG hinzugefügt, die Aurora PostgreSQL einer Umgebung nach Metadateninformationen abfragt, das Ergebnis in CSV-Dateien schreibt und die Dateien in Amazon S3 speichert.

12. August 2022

- [the section called “Exportieren von Umgebungs metadaten nach Amazon S3”](#)

[Neuer Beispielcode](#)

Es wurden aktualisierte Anweisungen und eine neue DAG hinzugefügt, die ein AWS CodeArtifact Token zur Laufzeit aktualisiert und das Ergebnis in Amazon S3 speichert.

03. August 2022

- [the section called “Erfrischen und AWS CodeArtifactToken zur Laufzeit”](#)

[Neuer Beispielcode](#)

Aktualisierte Anweisungen und ein DAG-Codebeispiel für die Verwendung von `ECSOperator` in Amazon MWAA hinzugefügt.

26. Juli 2022

- [the section called “Verwendung der `ECSOperator`”](#)

[Neuer Beispielcode](#)

Aktualisierte Anweisungen und ein DAG-Codebeispiel für die Verwendung von `SSHOperator` in Amazon MWAA hinzugefügt.

15. Juli 2022

- [the section called “Verwenden des `SSHOperator`”](#)

[Neuer Beispielcode](#)

Neue Anweisungen und ein DAG-Codebeispiel für die Verwendung von dbt Postgres mit Amazon MWAA hinzugefügt.

17. Juni 2022

- [the section called “Verwenden von dbt mit Amazon MWAA”](#)

[Neue Themen und Anwendungsfälle](#)

Neue Anweisungen und ein DAG-Codebeispiel für die Installation von Abhängigkeiten mithilfe von Python-Wheel-Dateien für Amazon MWAA-Umgebungen mit öffentlichem und privatem Zugriff hinzugefügt.

13. Mai 2022

- [Verwaltung von Abhängigkeiten mithilfe von Python-Rädern](#)

## [Neue Themen und Anwendungsfälle](#)

Neue Hinweise zur Auswahl der Apache Airflow-Metriken hinzugefügt, an die Amazon MWAA sendet. CloudWatch

19. April 2022

- [Auswahl der Apache Airflow-Metriken, die gemeldet werden](#)

## [Neue Anleitungen](#)

Amazon MWAA bietet einen Migrationsleitfaden für die Migration von Apache Airflow-Workflows aus selbstverwalteten Bereitstellungen sowie bestehenden Amazon MWAA-Umgebungen.

7. März 2022

- [Leitfaden zur Amazon MWAA-Migration](#)

## [Neue Themen und Anwendungsfälle](#)

Es wurden neue bewährte Sicherheitsmethoden für die Arbeit mit Apache Airflow hinzugefügt, einschließlich einer Lösung zur Erkennung von Änderungen an den Apache Airflow-Benutzerrechten.

18. Februar 2022

- [the section called “Bewährte Sicherheitsmethoden in Apache Airflow”](#)

## Neuer Beispielcode

Es wurde ein neues Codebeispiel für die Erstellung zeitzoneabhängiger DAGs mit [Pendulum](#) hinzugefügt und es wurde klargestellt, wie ein benutzerdefiniertes Plugin verwendet wird, um die Zeitzone zu ändern, in der Apache Airflow-Protokolle erstellt werden.

11. Februar 2022

- [the section called “Zeitzone einer DAG ändern”](#)

## [Start von Apache Airflow v2.2.2](#)

Amazon Managed Workflows für Apache Airflow unterstützt jetzt Apache Airflow v2.2.2. Ab Version 2.2 installiert Amazon MWAA Python-Pakete und benutzerdefinierte Plugins direkt auf dem Apache Airflow-Webserver, sodass Sie Ihre Umgebungen flexibler verwalten können. Weitere Informationen finden Sie unter den folgenden Topics.

27. Januar 2022

- [Apache Airflow-Versionen auf Amazon Managed Workflows für Apache Airflow](#).
- [the section called “Anbieterpakete für Apache Airflow v2.2.2-Verbindungen”](#).
- [Apache Airflow v2.2.2 Changelog](#) auf der Apache Airflow-Dokumentationswebsite.

## Neue Tutorials

Es wurde ein neues Tutorial hinzugefügt, das zeigt, wie eine neue benutzerdefinierte Apache Airflow-Rolle erstellt und die Rolle einem Apache Airflow-Benutzer zugewiesen wird, der über IAM zugewiesen wurde, um den Zugriff des Benutzers auf eine Teilmenge der angegebenen DAGs zu beschränken.

8. Dezember 2021

- [the section called “Tutorial : Benutzer auf eine Teilmenge von DAGs beschränken”](#)



## Behobene Probleme

22. November 2021

Es wurde eine Empfehlung mit bewährten Methoden zur Einstellung des Werts von `nscheduler.min_file_process_interval` hinzugefügt, um die CPU-Auslastung zu optimieren.

Es wurde ein Beispiel für eine IAM-Richtlinie hinzugefügt, das Zugriff auf Secrets Manager Manager-Ressourcen in der Ausführungsrolle gewährt. Es wurde ein Thema zur Fehlerbehebung zur Verwendung von Secrets Manager Manager-Bedingungsschlüsseln hinzugefügt.

- [Leistungsoptimierung der Art und Weise, wie der Scheduler DAGs analysiert](#)
- [Erteilen Sie Amazon MWAA die Erlaubnis, auf geheime Schlüssel von Secrets Manager zuzugreifen](#)
- [Konfiguration von Bedingungsschlüsseln in der Amazon MWAA-Ausführungsrolle für Secrets Manager](#)

## Neuer Beispielcode

Das folgende neue Codebeispiel zum Ändern der Zeitzone, in der DAGs mithilfe eines benutzerdefinierten Plugins verarbeitet werden, sowie ein neues Thema zur Fehlerbehebung für das Aufrufen des `dags backfill` Apache Airflow CLI-Befehls innerhalb eines Bash-Operators hinzugefügt.

1. November 2021

- [the section called “Zeitzone einer DAG ändern”](#)
- [CLI-Befehl zum Zurückfüllen mit einem Bash-Operator](#)

## Behobene Probleme

Es wurden Probleme im Amazon ECS-Operator-Codebeispiel behoben und die zusätzlichen Berechtigungen geklärt, die in der Amazon MWAA-Ausführungsrolle erforderlich sind, damit die Umgebung auf die Amazon ECS-Aufgabenprotokollgruppe in CloudWatch Logs zugreifen kann.

26. Oktober 2021

- [Amazon ECS-Berechtigungen](#).

### Neuer Beispielcode

Es wurde ein neues Codebeispiel hinzugefügt, das die Aurora PostgreSQL-Datenbank nach Informationen abfragt, die für DAG-Läufe relevant sind, und die Ergebnisse in eine auf Amazon CSV S3 gespeicherte Datei schreibt.

1. Oktober 2021

- [the section called “Exportieren von Umgebungs metadaten nach Amazon S3”](#).

### Behobene Probleme

Es wurden Informationen darüber korrigiert, wie Amazon MWAA neue und geänderte Objekte aus Ihrem Amazon S3 S3-Ziel-Bucket automatisch mit Ihren Schemata und Workern synchronisiert.

1. Oktober 2021

- [So funktioniert der DAG-Ordner](#).

### Wird jetzt unterstützt

Amazon MWAA unterstützt jetzt zusätzliche Anbieterpakete für Apache Airflow 2.0+. Weitere Informationen zu unterstützten Paketen finden Sie im Folgenden:

24. September 2021

- [the section called “Anbieterpakete für Apache Airflow v2.0.2-Verbindungen”](#).

## Neue Befehle und Verfahren

Zusätzliche Anleitungen und AWS CLI Befehlsbeispiele für die Erstellung eines Amazon S3 S3-Gateway-Endpunkts bei Verwendung einer Amazon VPC ohne Internetzugang hinzugefügt:

24. September 2021

- [Erstellen eines Amazon VPC-Netzwerks ohne Internetzugang](#).

## Neue Themen und Anwendungsfälle

Die folgenden Änderungen wurden hinzugefügt:

19. September 2021

- Es wurde ein neues Codebeispiel hinzugefügt, das einen Amazon Elastic Container Service-Operator in verwendet [the section called “Verwendung der ECSOperator”](#).
- Es wurden neue Themen zur Fehlerbehebung für Probleme bei der Konfiguration von Apache Airflow-Plugins in [the section called “Plug-ins”](#) hinzugefügt.

## Neue unterstützte Region

Amazon MWAA ist jetzt in den folgenden Regionen verfügbar:

31. August 2021

- Asien-Pazifik (Mumbai) – ap-south-1
- Asien-Pazifik (Seoul) – ap-northeast-2
- Europa (London) – eu-west-2
- Europa (Paris) – eu-west-3
- Kanada (Zentral) – ca-central-1
- Südamerika (São Paulo) – sa-east-1

Weitere Informationen zur regionalen Verfügbarkeit und zu den Service-Endpunkten finden Sie im Folgenden:

- [Amazon MWAA-Endpunkte und Kontingente](#) in der. Allgemeine AWS-Referenz

## Neue Themen und Anwendungsfälle

Die folgenden Änderungen wurden hinzugefügt:

27. August 2021

- Die Beispielrichtlinien wurden aktualisiert, sodass Amazon MWAA Amazon S3 S3-Einstellungen auf Kontoebene () abrufen kann. `s3:GetAccountPublicAccessBlock` [Amazon MWAA-Ausführungsrolle](#)

## Behobene Probleme

Die folgenden Änderungen wurden hinzugefügt:

27. August 2021

- Die AWS CloudFormation Vorlage wurde korrigiert, sodass eine selbstreferenzierende Regel für eingehende Nachrichten für die Sicherheitsgruppe in verwendet wird. [Erstellen Sie das VPC-Netzwerk](#)
- Die AWS CloudFormation Vorlage wurde dahingehend korrigiert, dass sie eine selbstreferenzierende Regel für eingehende Nachrichten für die Sicherheitsgruppe in verwendet. [Schnellstartanleitung für Amazon Managed Workflows für Apache Airflow](#)

## Neue Themen und Anwendungsfälle

Die folgenden Änderungen wurden hinzugefügt:

20. August 2021

- Der DAG-Decorator wurde der Liste der von Apache Airflow v2.0.2 unterstützten Funktionen hinzugefügt. [Apache Airflow-Versionen auf Amazon Managed Workflows für Apache Airflow](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

13. August 2021

- `celery.sync_parallelism` Anwendungsfall zu hinzugefügt [Leistungs-optimierung für Apache Airflow auf Amazon MWA](#).
- Dienstendpunkte wurden zur Kontingentsseite hinzugefügt und der Name in [Amazon Managed Workflows für Apache Airflow Service-Endpunkte und Kontingente](#) geändert.
- Die Netzwerkvoraussetzungen wurden auf der Grundlage von Benutzerfeedback unter [Beginnen Sie mit Amazon Managed Workflows for Apache Airflow](#) geklärt.
- In Airflow CLI-Befehle verschoben `dags list-runs` und `dags next-execution` zu nicht unterstützten Airflow CLI-Befehlen hinzugefügt. [Befehlsreferenz für Apache Airflow CLI](#)

## Neuer Beispielcode

Die folgenden Änderungen wurden hinzugefügt:

13. August 2021

- Bash-Beispiel zum Setzen, Abrufen oder Löschen einer Apache Airflow v2.0.2-Variablen hinzugefügt. [Befehlsreferenz für Apache Airflow CLI](#)
- Apache Airflow v2.0.2-Abhängigkeiten und ein Beispiel für eine Airflow-Verbindung wurden hinzugefügt. [Amazon MWAA mit Amazon RDS für Microsoft SQL Server verwenden](#)

## Behobene Probleme

Die folgenden Änderungen wurden hinzugefügt:

13. August 2021

- Das Python-Codebeispiel wurde auf der Grundlage von Benutzerfeedback unter behoben [Erstellen einer SSH-Verbindung mit dem SSHOperator](#) .



## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

6. August 2021

- Zu `variables set` den unterstützten Airflow CLI-Befehlen in [Befehlsreferenz für Apache Airflow CLI](#) verschoben.
- Die Zusammenfassung der Änderungen in Version 2.0.2 wurde von der Airflow-Versionsseite zu „[Python-Abhängigkeiten installieren](#)“ basierend auf Benutzerfeedback“ hinzugefügt.
- Die Zusammenfassung der Änderungen in v2.0.2 wurde von der Airflow-Versionsseite zu „Basierend auf Benutzerfeedback“ hinzugefügt. [Befehlsreferenz für Apache Airflow CLI](#)
- Die Zusammenfassung der Änderungen in v2.0.2 wurde von der Airflow-Versionsseite zu „Basierend auf Benutzerfeedback“ hinzugefügt. [Übersicht über Verbindungsarten](#)
- Die Zusammenfassung der Änderungen in v2.0.2 wurde von der Airflow-Versionsseite zu „Basierend auf Benutzerfeedback“ hinzugefügt. [Installation benutzerdefinierter Plugins](#)

- Die Zusammenfassung der Änderungen in v2.0.2 wurde von der Airflow-Versionsseite zu „Basierend auf Benutzerfeedback“ hinzugefügt. [Hinzufügen oder Aktualisieren von DAGs](#)

### Neuer Beispielcode

Die folgenden Änderungen wurden hinzugefügt:

6. August 2021

- Beispielcode für Apache Airflow v2.0.2 wurde hinzugefügt. [Verwenden einer DAG zum Importieren von Variablen in die CLI](#)
- Beispielcode für Apache Airflow v2.0.2 wurde hinzugefügt. [Aufrufen von DAGs mit einer Lambda-Funktion](#)

## Neue Themen und Anwendungsfälle

Die folgenden Änderungen wurden hinzugefügt:

29. Juli 2021

- Es wurde ein Thema zur Fehlerbehebung für „Ich kann meine Verbindung in der Airflow-Benutzeroberfläche nicht sehen“ unter hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#)
- Es wurde eine Liste von Amazon VPCs hinzugefügt, die Amazon MWAA unterstützt. [Informationen zu Netzwerken in Amazon MWAA](#)

## Behobene Probleme

Die folgenden Änderungen wurden hinzugefügt:

29. Juli 2021

- Das Python-Codebeispiel wurde auf der Grundlage von Benutzerfeedback behoben, unter dem das Web-Login-Token gedruckt [Erstellen Sie ein Apache Airflow-Webserver-Zugriffstoken](#) werden konnte.
- Beim Thema Snowflake-Verbindung auf der Grundlage von Benutzerfeedback wurde nun ein einfaches Anführungszeichen für den Warehouse-Parameter bei verwendet. [Amazon Managed Workflows for Apache Airflow](#)

## Themen wurden entfernt oder verschoben

Die folgenden Änderungen wurden hinzugefügt:

23. Juli 2021

- Die bestehende Seite wurde so umstrukturiert, dass sie alle Dokumentationsseiten für Überwachung und Metriken enthält. [Überwachung und Metriken für Amazon Managed Workflows for Apache Airflow](#)
- In [Apache Airflow v2-Umgebungsmetriken in CloudWatch](#) das Navigationsmenü für Überwachung und Metriken verschoben.

## Neue Anleitungen

Die folgenden Änderungen wurden hinzugefügt:

23. Juli 2021

- Erstellt [In Amazon MWAA-Umgebungen installierte Apache Airflow-Anbieterpakete](#).
- Erstellt [Überblick über die Überwachung auf Amazon MWAA](#).
- Erstellt [Audit-Logs anzeigen AWS CloudTrail](#).
- Erstellt [Airflow-Protokolle in Amazon anzeigen CloudWatch](#).

## Behobene Probleme

Die folgenden Änderungen wurden hinzugefügt:

23. Juli 2021

- Das Python-Codebeispiel, das auf Benutzerfeedback basiert, wurde korrigiert, sodass eine Airflow-V erbindungszeichenfolge in der richtigen Reihenfolge generiert wurde, und der Port-Parameter wurde hinzugefügt. [Konfiguration einer Apache Airflow-V erbindung mithilfe eines Geheimnisses AWS Secrets Manager](#)
- Es wurde ein Schritt zur lokalen Installation eines Entpackungspakets hinzugefügt, das auf Benutzerfeedback basiert. [Ein benutzerdefiniertes Plugin mit Oracle erstellen](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

16. Juli 2021

- Thema für AWS DMS-Operatoren unter [Häufig gestellte Fragen zu Amazon MWAA](#) hinzugefügt.
- Es wurde ein Thema zur Fehlerbehebung für einen Remote-Protokollfehler zu [Amazon Managed Workflows for Apache Airflow](#) hinzugefügt.
- Zu `variables set` den nicht unterstützten Airflow CLI-Befehlen in verschoben. [Befehlsreferenz für Apache Airflow CLI](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

09. Juli 2021

- Es wurden sequentielle Schritte hinzugefügt, um eine Datei requirements.txt auf der Grundlage von Benutzerfeedback unter zu [Python-Abhängigkeiten installieren](#) erstellen.
- Es wurden sequentielle Schritte hinzugefügt, um eine Datei plugins.zip auf der Grundlage von Benutzerfeedback unter zu erstellen. [Installation benutzerdefinierter Plugins](#)
- Im gesamten Benutzerhandbuch wurden Querverweise links zum API-Referenzhandbuch im [Amazon Managed Workflows for Apache Airflow API-Referenzhandbuch](#) hinzugefügt.
- Es wurde ein Thema hinzugefügt, das erklärt, warum Plugins im Menü Airflow 2.0 Admin > Plugins nicht angezeigt werden unter. [Häufig gestellte Fragen zu Amazon MWAA](#)



## [Neue Anleitungen](#)

Die folgenden Änderungen wurden hinzugefügt:

09. Juli 2021

- Erstellt [Löschen von Dateien auf Amazon S3](#).

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

2. Juli 2021

- Eine Liste unterstützter Werte wurde hinzugefügt unter [Verwendung von vom Kunden verwalteten Schlüsseln zur Verschlüsselung](#).
- Das Beispiel für eine private Repo-URL wurde auf der Grundlage von Benutzerfeedback in [Verwaltung von Python-Abhängigkeiten in requirements.txt](#) aktualisiert und klarer formuliert.

## [Neuer Beispielcode](#)

Die folgenden Änderungen wurden hinzugefügt:

2. Juli 2021

- Beispielcode für Apache Airflow v1.10.12 zur Verwendung eines privaten Schlüssels AWS Secrets Manager für eine SSH-Verbindung unter hinzugefügt. [Erstellen einer SSH-Verbindung mit dem SSHOperator](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

25. Juni 2021

- Hinzugefügt StartedTaskInstances und FinishedTaskInstances Metriken zu [Apache Airflow v2-Umgebungsmetriken in CloudWatch](#).

## [Neuer Beispielcode](#)

Die folgenden Änderungen wurden hinzugefügt:

25. Juni 2021

- Beispielcode für Apache Airflow v2.0.2 wurde hinzugefügt unter [Amazon MWAA mit Amazon EKS verwenden](#)

## [Neue Anleitungen](#)

Die folgenden Änderungen wurden hinzugefügt:

25. Juni 2021

- Erstellt [Leistungsoptimierung für Apache Airflow auf Amazon MWAA](#).

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

18. Juni 2021

- Hinzugefügt `connections add` und `connections delete` zu den unterstützten Apache Airflow v2.0.2 CLI-Befehlen unter. [Befehlsreferenz für Apache Airflow CLI](#)
- Es wurde hinzugefügt, dass die neueste verfügbare Version Apache Airflow v2.0.2 unter AWS CloudFormation ist. [Schnellstartanleitung für Amazon Managed Workflows für Apache Airflow](#)
- Es wurde eine Frage zum Speichern temporärer Daten auf Apache Airflow Workers hinzugefügt. [Häufig gestellte Fragen zu Amazon MWAA](#)
- Es wurde ein Thema für den Fehler „Executor reports Task Instance %s finished“ hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#)
- Es wurde ein Thema für das Protokoll „Der Server hat die Verbindung unerwartet geschlossen“ hinzugefügt. [Amazon Managed](#)

## [Workflows for Apache Airflow](#)

- Es wurde ein Beispiel hinzugefügt, um CLI-Befehle auf einem SSH-Tunnel zu einem Bastion-Host auszuführen. [Ein Apache Airflow CLI-Token erstellen](#)
- Ein Thema für zufällig generierte Benutzernamen wurde hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#)
- Es wurde ein Thema für einen 503-Fehler beim Ausführen einer DAG in der CLI zu hinzugefügt [Amazon Managed Workflows for Apache Airflow](#).
- Es wurde ein Thema für benutzerdefinierte Plugins in Apache Airflow v2.0.2 hinzugefügt, die eine Airflow-Konfigurationsoption von benötigen, um Plugins `core.lazy_load_plugins : False` zu Beginn jedes Airflow-Prozesses zu laden, um die Standardinstellung der Version außer Kraft zu setzen. [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#)

- Der Beispielcode für Airflow-Konfigurationsoptionen für Apache Airflow v2.0.2-Plugins wurde unter hinzugefügt. [Erstellen eines benutzerdefinierten Plugins mit Apache Hive und Hadoop](#)
- Der Beispielcode für Airflow-Konfigurationsoptionen für Apache Airflow v2.0.2-Plugins wurde unter hinzugefügt. [Erstellen eines benutzerdefinierten Plugins, das Laufzeitumgebungsvariablen generiert](#)
- Der Beispielcode für Airflow-Konfigurationsoptionen für Apache Airflow v2.0.2-Plugins wurde unter hinzugefügt. [Ein benutzerdefiniertes Plugin für Apache Airflow erstellenPythonVirtualenvOperator](#)
- Der Beispielcode für Airflow-Konfigurationsoptionen für Apache Airflow v2.0.2-Plugins wurde unter hinzugefügt. [Ein benutzerdefiniertes Plugin mit Oracle erstellen](#)

## Neuer Beispielcode

Die folgenden Änderungen wurden hinzugefügt:

18. Juni 2021

- Beispielcode für eine Apache Airflow Snowflake -Verbindung wurde hinzugefügt unter. [Mit einem geheimen Schlüssel inAWS Secrets Manager für eine Apache Airflow Snowflake-Verbindung](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

2. Juni 2021

- Hinweise zur serverseitigen Verschlüsselung wurden hinzugefügt. [Erstellen eines Amazon S3 Bucket erstellen](#)
- Das Secrets-Backend für Apache Airflow v2.0.2 wurde hinzugefügt. [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#)
- Es wurde eine Frage für Anfragen zur Erhöhung der Quote von Apache Airflow Workers hinzugefügt. [Häufig gestellte Fragen zu Amazon MWAA](#)
- Es wurde eine Frage hinzugefügt, anhand derer anhand von Metriken bestimmt wird, ob Apache Airflow Workers skaliert werden soll. [Häufig gestellte Fragen zu Amazon MWAA](#)
- Frage zur Erstellung benutzerdefinierter Metriken in CloudWatch to [Häufig gestellte Fragen zu Amazon MWAA](#) hinzugefügt.
- Es wurden Schritte hinzugefügt, um private IP-

Adressen für einen Amazon S3 S3-VPC-Schnittstellenendpunkt für eine VPC mit privatem Routing in zu aktivieren. [Erstellung der erforderlichen VPC-Serviceendpunkte in einer Amazon VPC mit privatem Routing](#)

- Es wurde eine Option hinzugefügt, um einen SSH-Tunnel mithilfe der lokalen Portweiterleitung in einzurichten. [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem Linux Bastion Host](#)

### [Neuer Beispielcode](#)

Die folgenden Änderungen wurden hinzugefügt:

2. Juni 2021

- Beispielcode für eine DAG hinzugefügt, die die Amazon Aurora PostgreSQL-Metadatenbank abfragt und benutzerdefinierte Metriken für Amazon CloudWatch veröffentlicht unter. [Verwenden einer DAG zum Schreiben benutzerdefinierter Metriken in CloudWatch](#)



## Neue Leitfäden

Die folgenden Änderungen wurden hinzugefügt:

2. Juni 2021

- Eine Anleitung zur austauschbaren Verwendung von Verbindungsvorlagen in der Apache Airflow-Benutzeroberfläche wurde unter erstellt. [Übersicht über Verbindungsarten](#)

## Behobene Probleme

Die folgenden Änderungen wurden hinzugefügt:

2. Juni 2021

- In Option drei: Erstellen eines VPC-Netzwerks ohne Internetzugang zu wurden der AWS CloudFormation Vorlage Apache Airflow VPC-Endpunkte hinzugefügt. [Erstellen Sie das VPC-Netzwerk](#)

## [Start von Apache Airflow v2.0.2](#)

Allgemeine Verfügbarkeit von Apache Airflow v2.0.2.

26. Mai 2021

- Erstellt. [Apache Airflow-Versionen auf Amazon Managed Workflows für Apache Airflow](#)
- Erstellt [Apache Airflow v2-Umgebungsmetriken in CloudWatch](#).
- Versionsspezifische Links für Apache Airflow v2.0.2 zu hinzugefügt. [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#)
- Versionsspezifische Anleitungen für Apache Airflow v2.0.2 wurden hinzugefügt. [Python-Abhängigkeiten installieren](#)
- Versionsspezifische Anleitungen für Apache Airflow v2.0.2 wurden hinzugefügt. [Verwaltung von Python-Abhängigkeiten in requirements.txt](#)
- Beispiel-Plugins für Apache Airflow v2.0.2 wurden hinzugefügt. [Installation benutzerdefinierter Plugins](#)
- Beispielcode für Apache Airflow v2.0.2 wurde hinzugefügt. [Aurora-PostgreSQL-Datenbank](#)

### [Bereinigung in einer Amazon-MWAA-Umgebung](#)

- Beispielcode für Apache Airflow v2.0.2 wurde hinzugefügt. [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Verbindung](#)
- Beispielcode für Apache Airflow v2.0.2 wurde hinzugefügt. [Ein benutzerdefiniertes Plugin für Apache Airflow erstellen](#)[Python VirtualenvOperator](#)
- Apache Airflow v2.0.2-Befehle wurden hinzugefügt. [Befehlsreferenz für Apache Airflow CLI](#)
- Apache Airflow v2.0.2-Skripte wurden hinzugefügt. [Ein Apache Airflow CLI-Token erstellen](#)
- Es wurde ein Hinweis hinzugefügt, dass Amazon MWAA standardmäßig die neueste Apache Airflow-Version verwendet. [Erstellen Sie eine Amazon MWAA-Umgebung](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

14. Mai 2021

- Es wurden Anleitungen zur Fehlerbehebung bei Airflow-Aufgaben hinzugefügt, bei denen Probleme auftreten oder nicht ausgeführt werden. [Amazon Managed Workflows for Apache Airflow](#)

## [Behobene Probleme](#)

Die folgenden Änderungen wurden hinzugefügt:

12. Mai 2021

- Wir haben den Code des Beispiel-Plugins aktualisiert, um die neueste Java-Version in zu verwenden [Erstellen eines benutzerdefinierten Plugins mit Apache Hive und Hadoop](#). Zuvor war es `soos.enviro`  
`n["JAVA_HOME"]="/usr/lib/jvm/jre-1.8.0-openjdk-1.8.0.272.b10-1.amzn2.0.1.x86_64"` .

## [Themen wurden entfernt oder verschoben](#)

Die folgenden Änderungen wurden hinzugefügt:

10. Mai 2021

- Themen wurden nach Kategorien [Amazon Managed Workflows for Apache Airflow](#) auf neue Seiten verschoben.

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

10. Mai 2021

- [Amazon S3 S3-Bucket-Übersicht](#) wurde hinzugefügt [Arbeiten mit DAGs auf Amazon MWAA](#).

## [Themen wurden entfernt oder verschoben](#)

Die folgenden Änderungen wurden hinzugefügt:

7. Mai 2021

- [Zugreifen auf Apache Airflow](#) Zur Navigation auf oberster Ebene verschoben und Seiten für [Erstellen Sie ein Apache Airflow-Webserver-Zugriffstoken](#) [Ein Apache Airflow CLI-Token erstellen](#), und [Befehlsreferenz für Apache Airflow CLI](#) hinzugefügt.

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

7. Mai 2021

- Es wurden versionsspezifische Links zum Apache Airflow-Referenzhandbuch für alle unterstützten und nicht unterstützten Airflow CLI-Befehle hinzugefügt. [Befehlsreferenz für Apache Airflow CLI](#)
- Es wurden versionsspezifische Links zum Apache Airflow-Referenzhandbuch für alle Konfigurationsoptionen hinzugefügt. [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#)
- Das Amazon MWAA-CLI-Hilfsprogramm wurde hinzugefügt. [Verwaltung von Python-Abhängigkeiten in requirements.txt](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

30. April 2021

- Es wurden flache und verschachtelte Beispiele für die Strukturierung einer plugins.zip hinzugefügt. [Installation benutzerdefinierter Plugins](#)
- Das Amazon MWAA CLI-Hilfsprogramm wurde zu den Seiten [Hinzufügen oder Aktualisieren von DAGs](#), [Installation benutzerdefinierter Plugins](#), und [Python-Abhängigkeiten installieren](#) hinzugefügt.
- Die Abschnitte „Übersicht“, „Upload auf Amazon S3“ und „Installation auf Amazon MWAA“ wurden auf Grundlage von Benutzerfeedback auf den Seiten neu strukturiert. [Installation benutzerdefinierter Plugins](#), [Python-Abhängigkeiten installieren](#)
- Es wurde ein Anwendungsbeispiel hinzugefügt, um erforderliche VPC-Endpunkte zu erstellen und an eine bestehende Amazon-VPC ohne Internetzugang anzuhängen. [Informationen zu Netzwerken in Amazon MWAA](#)

## Neuer Beispielcode

Die folgenden Änderungen wurden hinzugefügt:

30. April 2021

- Beispielcode hinzugefügt, der einen geheimen Schlüssel in Secrets Manager für eine Apache Airflow-Variable in [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Variable](#) verwendet.

## Neue Anleitungen

Die folgenden Änderungen wurden hinzugefügt:

30. April 2021

- Erstellt [Erstellung der erforderlichen VPC-Serviceendpunkte in einer Amazon VPC mit privatem Routing](#).

## Behobene Probleme

Die folgenden Änderungen wurden hinzugefügt:

30. April 2021

- Ups! Wir haben `core.default_ui_timezone` auf `webserver.default_ui_timezone` in [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#) aktualisiert.



## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

23. April 2021

- Windows (PuTTY) -Schritte für den SSH-Tunnel wurden hinzugefügt. [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem Linux Bastion Host](#)
- Es wurde ein Thema für hinzugefügt `apache-airflow-providers-amazon`, das nur mit Apache Airflow 2.0 kompatibel ist. [Amazon Managed Workflows for Apache Airflow](#)

## [Neuer Beispielcode](#)

Die folgenden Änderungen wurden hinzugefügt:

23. April 2021

- Beispielcode hinzugefügt, der einen geheimen Schlüssel in Secrets Manager für eine Apache Airflow-Verbindung in [Mit einem geheimen Schlüssel in AWS Secrets Manager für eine Apache Airflow-Verbindung](#) verwendet.

## [Neue Anleitungen](#)

Die folgenden Änderungen wurden hinzugefügt:

23. April 2021

- Erstellt [Informationen zu Netzwerken in Amazon MWAA](#).
- Erstellt [Sicherheit in Ihrer VPC auf Amazon MWAA](#).
- Erstellt [Verwaltung des Zugriffs auf servicespezifische Amazon VPC-Endpunkte auf Amazon MWAA](#).

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

16. April 2021

- Es wurde eine neue AWS CloudFormation Vorlage hinzugefügt, um ein Amazon VPC-Netzwerk ohne Internetzugang zu erstellen. [Erstellen Sie das VPC-Netzwerk](#)
- Es wurde ein neues Tutorial zum Erstellen eines Ins AWS Client VPN hinzugefügt. [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem AWS Client VPN](#)
- Der Name der Netzwerkzugriffsseite wurde aufgrund von Benutzerfeedback in Apache Airflow-Zugriffsmodi geändert und die Dokumentation wurde optimiert. [Apache Airflow-Zugriffsmodi](#)
- Die Dokumente wurden so optimiert, dass sie nur Informationen und Vorlagen für die ersten Schritte von Amazon VPC enthalten, die auf Benutzerfeedback basieren. [Erstellen Sie das VPC-Netzwerk](#)
- Eine Problemumgehung für BigQuery Operatoren wurde hinzugefügt. [Amazon](#)

## [Managed Workflows for Apache Airflow](#)

- Eine Best Practice-Datei mit Einschränkungen für Apache Airflow v1.10.12 wurde hinzugefügt. [Python-Abhängigkeiten installieren](#)

## [Neuer Beispielcode](#)

Die folgenden Änderungen wurden hinzugefügt:

16. April 2021

- Beispielcode hinzugefügt, um ein benutzerdefiniertes Plugin mit Oracle zu erstellen. [Ein benutzerdefiniertes Plugin mit Oracle erstellen.](#)
- Beispielcode hinzugefügt, um ein benutzerdefiniertes Plugin zu erstellen, das Laufzeitumgebungsvariablen in generiert. [Erstellen eines benutzerdefinierten Plugins, das Laufzeitumgebungsvariablen generiert.](#)
-

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

9. April 2021

- Es wurde ein Thema für die Anforderung einer selbstreferenzierenden Regel für eine VPC-Sicherheitsgruppe hinzugefügt. [Häufig gestellte Fragen zu Amazon MWAA](#)
- Verzeichnis und Größenbeschränkungen für benutzerdefinierte Plugins wurden hinzugefügt. [Installation benutzerdefinierter Plugins](#)
- Anforderungsverzeichnis und Größenbeschränkungen für hinzugefügt [Python-Abhängigkeiten installieren](#).
- Die Apache Airflow-Konfigurationsoptionen für `foo.user` und `foo.pass` in [Verwaltung von Python-Abhängigkeiten in `requirements.txt`](#) wurden klargestellt.
- Eine Übersicht über die Konfigurationsoptionen wurde hinzugefügt. [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#)

## Neuer Beispielcode

Die folgenden Änderungen wurden hinzugefügt:

9. April 2021

- Beispielcode hinzugefügt, um ein benutzerdefiniertes Plugin mit PythonVirtualenvOperator in zu erstellen. [Ein benutzerdefiniertes Plugin für Apache Airflow erstellen](#) PythonVirtualenvOperator.
- Beispielcode hinzugefügt, um ein benutzerdefiniertes Plugin mit Apache Hive und Hadoop zu erstellen. [Erstellen eines benutzerdefinierten Plugins mit Apache Hive und Hadoop](#)

## Behobene Probleme

Die folgenden Änderungen wurden hinzugefügt:

31. März 2021

- Ups! Wir haben das Format für eine Datei requirements.txt aktualisiert und ein Beispiel hinzugefügt, das mit Apache Airflow v1.10.12 kompatibel ist. [Python-Abhängigkeiten installieren](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

26. März 2021

- Es wurde ein Workaround für das Entfernen einer Datei requirements.txt oder plugins.zip zu [Häufig gestellte Fragen zu Amazon MWAA](#) hinzugefügt.
- Eine Bash-Problemumgehung für SSH in einer Umgebung wurde hinzugefügt. [Häufig gestellte Fragen zu Amazon MWAA](#)
- Ein Thema für CloudTrail ResourceAlreadyExistsException Fehler wurde hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

19. März 2021

- Es wurde eine Liste der AWS Dienste hinzugefügt, die verwendet wurden [Amazon MWAA-Ausführungsrolle](#).
- Es wurde eine Liste der AWS Dienste hinzugefügt, die dafür verwendet wurden [Servicebezogene Rolle für Amazon MWAA](#).
- Frage für Python 3.7-Version für Amazon MWAA hinzugefügt. [Häufig gestellte Fragen zu Amazon MWAA](#)
- Frage für PythonVirtualenvOperator zu hinzugefügt. [Häufig gestellte Fragen zu Amazon MWAA](#)
- Das Fehlerbehebungsskript wurde als nächste Schritte für alle Themen im Zusammenhang mit VPC und Umgebungskonfiguration unter [Amazon Managed Workflows for Apache Airflow](#) hinzugefügt.
- Es wurde klargestellt, dass sich eine Linux-Bastion in derselben Region befinden muss wie eine Umgebung in. [Tutorial: Konfiguration des privaten Netzwerks](#)



## [Zugriff mit einem Linux Bastion Host](#)

### [Neue Anleitungen](#)

Die folgenden Änderungen wurden hinzugefügt:

19. März 2021

- Der Leitfaden für Apache Airflow-Verbindungen wurde für AWS Secrets Manager at [Konfiguration einer Apache Airflow-Verbindung mithilfe eines Geheimnisses AWS Secrets Manager](#) erstellt.
- Ein Schnellstart-Tutorial mit einer AWS CloudFormation Vorlage zum Erstellen der Amazon VPC-Infrastruktur, des Amazon S3 S3-Buckets und der Amazon MWAA-Umgebung unter erstellt. [Schnellstartanleitung für Amazon Managed Workflows für Apache Airflow](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

12. März 2021

- Das Thema zur Fehlerbehebung beim Erstellen eines Amazon S3 S3-Buckets wurde hinzugefügt [Amazon Managed Workflows for Apache Airflow](#).
- Es wurden Schritte zum Erstellen und Anhängen einer JSON-Richtlinie hinzugefügt [Amazon MWAA-Ausführungsrolle](#).

## [Neuer Beispielcode](#)

Die folgenden Änderungen wurden hinzugefügt:

12. März 2021

- Beispielcode hinzugefügt, um eine Konfiguration hinzuzufügen, wenn eine DAG ausgelöst wird. [Zugreifen auf Apache Airflow](#)

## [Neue Anleitungen](#)

Die folgenden Änderungen wurden hinzugefügt:

12. März 2021

- Leitfaden für bewährte Verfahren unter [erstellt Verwaltung von Python-Abhängigkeiten in requirements.txt](#).

## Neue Themen und Anwendungsfälle

Die folgenden Änderungen wurden hinzugefügt:

5. März 2021

- Das Thema Google/GCP/BigQuery zur Fehlerbehebung wurde hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#)
- Das Thema Cython-Fehlerbehebung wurde hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#)
- Das Thema MySQL-Fehlerbehebung wurde hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#).
- Das Thema zur Behebung von 5xx-Webserverfehlern wurde hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#)

### Wird jetzt unterstützt

Die folgenden Änderungen wurden hinzugefügt:

4. März 2021

- Bisher `backend_kwarg` wurde für nicht unterstützte AWS Secrets Manager und Sie benötigten eine Problemumgehung, um den Secrets Manager Manager-Funktionsaufruf zu überschreiben. Wird jetzt `backend_kwarg` unterstützt. Das Thema zur AWS Secrets Manager Fehlerbehebung finden Sie unter [Amazon Managed Workflows for Apache Airflow](#).

### Behobene Probleme

Die folgenden Änderungen wurden hinzugefügt:

4. März 2021

- Ups! Wir haben die Größe jeder Umgebungsklasse aktualisiert, um die tatsächliche Größe in [Konfiguration der Amazon MWAA-Umgebungsklasse](#) GB widerzuspiegeln.

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

26. Februar 2021

- Privater Netzwerkzugriff mithilfe einer VPC-Endpunktrichtlinie zu [Apache Airflow-Zugriffsmodi](#) hinzugefügt.
- Es wurden zusätzliche Prüfungen für das Thema Problembehandlung beim Erstellen einer Umgebung hinzugefügt [Amazon Managed Workflows for Apache Airflow](#).
- Es wurden Schritte zum Anzeigen von Protokollen für `requirements.txt` hinzugefügt [Python-Abhängigkeiten installieren](#).

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

25. Februar 2021

- Der Apache Hive-Anwendungsfall wurde hinzugefügt. [Python-Abhängigkeiten installieren](#)
- In der Dokumentation wurde klargestellt, dass die erforderlichen Abhängigkeiten für ein Apache Airflow-Paket in der `requirements.txt` Datei unterhalten sein müssen. [Python-Abhängigkeiten installieren](#)
- Das Thema „requirements.txt zur Fehlerbehebung aktualisieren“ wurde hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#)

## [Neue Tutorials](#)

Die folgenden Änderungen wurden hinzugefügt:

22. Februar 2021

- Tutorial zu privaten Netzwerken hinzugefügt [Tutorial: Konfiguration des privaten Netzwerkzugriffs mit einem Linux Bastion Host.](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

22. Februar 2021

- Private und öffentliche Netzwerkkonfigurationen wurden hinzugefügt [Apache Airflow-Zugriffsmodi](#).
- Anwendungsfall und Benutzerszenarien für Entwicklungsgruppen wurden hinzugefügt [Amazon MWAA-Ausführungsrolle](#).

## [Neuer Beispielcode](#)

Die folgenden Änderungen wurden hinzugefügt:

22. Februar 2021

- Beispiel-Python-Skripte für Web-Login-Token und CLI-Token wurden hinzugefügt [Zugreifen auf Apache Airflow](#).
- Beispielcode hinzugefügt, um DAG in einer anderen Umgebung auszulösen [Codebeispiele für Amazon Managed Workflows für Apache Airflow](#).
- Beispielcode hinzugefügt, um DAG mithilfe einer Lambda-Funktion auszulösen. [Aufrufen von DAGs mit einer Lambda-Funktion](#)

## Neue Befehle und Verfahren

Die folgenden Änderungen wurden hinzugefügt:

22. Februar 2021

- Schrittweise Verfahren zu allen Skripten unter [hinzugefügtZugreifen auf Apache Airflow](#).

## Neuer Beispielcode

Die folgenden Änderungen wurden hinzugefügt:

17. Februar 2021

- Das Curl-Beispiel für das Web-Login-Token wurde unter [Zugreifen auf Apache Airflow](#) aktualisiert.
- Beispielcode für die Verbindung mit einem Amazon RDS Microsoft SQL Server hinzugefügt [Amazon MWAA mit Amazon RDS für Microsoft SQL Server verwenden](#).



## [Neue Befehle und Verfahren](#)

Die folgenden Änderungen wurden hinzugefügt:

17. Februar 2021

- AWS CLI Befehle zu [Arbeiten mit DAGs auf Amazon MWAA](#) Seiten hinzugefügt.
- Apache Airflow unterstützt keine serialisierten DAGs in CLI-Befehlen. Da die CLI auf dem Webserver läuft, für den es aus Sicherheitsgründen keine Plugins oder Anforderungen gibt, unterstützen MWAA-Umgebungen mit einer Datei `plugins.zip` oder `requirements.txt` diese Befehle nicht. Apache Airflow `list_dags` und `backfill` Befehle wurden zu nicht unterstützten Befehlen unter verschoben. [Zugreifen auf Apache Airflow](#)

## [GitHub starten](#)

Die Dokumente zum Benutzerhandbuch sind jetzt als Open Source verfügbar GitHub. Wählen Sie auf einer beliebigen Seite „Diese Seite bearbeiten auf GitHub“.

17. Februar 2021

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

12. Februar 2021

- Es wurde eine Frage für den Anwendungsfall Step Functions im Vergleich zu Amazon MWAA hinzugefügt. [Amazon Managed Workflows for Apache Airflow](#)
- CLI-Zugriffsrichtlinie zu hinzugefügt [Zugreifen auf eine Amazon MWAA-Umgebung](#).
- Es wurde klargestellt, dass alle unterstützten Apache Airflow-Konfigurationsoptionen unter [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#) angegeben werden können.
- In den Dokumenten wurde klargestellt, dass MWAA, wenn ein Fargate-Container in einer Availability Zone ausfällt, zu dem anderen Container in einer anderen Availability Zone unterwechselt. [Erstellen Sie das VPC-Netzwerk](#)

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

5. Februar 2021

- [Konfiguration der Amazon MWAA-Umgebungs-klasse](#) hinzugefügt.

## [Themen wurden entfernt oder verschoben](#)

Die folgenden Änderungen wurden hinzugefügt:

4. Februar 2021

- Die Anforderung, dass der Amazon S3 S3-Bucket-Name mit `airflow-` at beginnen muss, wurde entfernt [Beginnen Sie mit Amazon Managed Workflows for Apache Airflow.](#)
- Verschieben [Zugreifen auf eine Amazon MWAA-Umgebung](#) und [Amazon MWAA-Ausführungsrolle nach Verwaltung des Zugriffs auf eine Amazon MWAA-Umgebung.](#)

## [Amazon MWA CloudFormation](#)

Aktualisieren Sie die Parameter, um eine Umgebung bei [Amazon CloudFormation MWA](#) zu erstellen.

4. Februar 2021

- Entfernen. SubnetList
- Entfernen TagList.
- Hinzufügen NetworkConfiguration.
- Hinzufügen TagMap.
- Fügen Sie Beispiele für Anfragen zum Erstellen einer Umgebung hinzu.

## [Neue Themen und Anwendungsfälle](#)

Die folgenden Änderungen wurden hinzugefügt:

29. Januar 2021

- Eine Beispiel-E-Mail-Konfiguration wurde hinzugefügt [Verwenden der Apache Airflow-Konfigurationsoptionen auf Amazon MWAA](#).
- Es wurde ein Thema PostgresHook zur Fehlerbehebung hinzugefügt [Amazon Managed Workflows for Apache Airflow](#).
- Thema AWS Secrets Manager zur Fehlerbehebung hinzugefügt zu [Amazon Managed Workflows for Apache Airflow](#).
- Hochleistungs-Anwendungsfall zu hinzugefügt [Konfiguration der automatischen Skalierung von Amazon MWAA Worker](#).

## [Einführung von Amazon MWAA](#)

Allgemeine Verfügbarkeit von Amazon Managed Workflows für Apache Airflow.

24. November 2020

- Dokumentation des Benutzerhandbuchs
- AWS CloudFormation Dokumentation

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.