Leitfaden zur Implementierung

Verteilte Lasttests auf AWS

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Verteilte Lasttests auf AWS: Leitfaden zur Implementierung

Copyright © 2026 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Handelsmarken und die 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 Marken, die nicht im Besitz von Amazon sind, gehören den jeweiligen Besitzern, die möglicherweise mit Amazon verbunden sind oder von Amazon gesponsert werden.

Table of Contents

Übersicht über die Lösung 1
Features 2
Vorteile 4
Anwendungsfälle 5
Konzepte und Definitionen 6
Übersicht über die Architektur 8
Architekturdiagramm 8
Überlegungen zum Well-Architected AWS-Design 11
Operative Exzellenz 11
Sicherheit 11
Zuverlässigkeit 12
Leistungseffizienz 13
Kostenoptimierung 13
Nachhaltigkeit 13
Einzelheiten zur Architektur 15
Frontend 15
Test-API laden 15
Web-Konsole 16
MCP-Server (optional) 17
Backend 17
Pipeline für Container-Images 17
Testen der Framework-Bereitstellung 18
Infrastruktur wird getestet 18
Test-Engine auslasten 19
MCP-Server 19
Amazon Bedrock Gateway AgentCore 20
DLT MCP Server Lambda 20
Integration der Authentifizierung 20
AWS-Services in dieser Lösung 21
So funktioniert Distributed Load Testing auf AWS 23
MCP-Server-Workflow (optional) 25
Designüberlegungen 27
Unterstützte Anwendungen 27
Testtypen 28

Modi der Verkehrsform 33
Planung von Tests 34
Gleichzeitige Tests 35
Benutzerverwaltung 35
Identitätsanbieterverbund 36
Regionale Bereitstellung 39
Planen Sie Ihren Einsatz 40
Cost (Kosten) 40
Kosten für die Integration von DevOps AWS-Agenten 42
CloudWatch Dashboard-Kosten pro Testszenario 42
Zusätzliche Kosten für MCP-Server (optional) 43
Zusätzliche Kosten für die von ALB + ECS Fargate gehostete Webkonsole (optional) 43
Sicherheit 44
IAM-Rollen 45
Amazon CloudFront 45
Amazon API Gateway 45
AWS Fargate-Sicherheitsgruppe 46
Amazon VPC 46
Netzwerk-Stresstest 48
Beschränken Sie den Zugriff auf die öffentliche Benutzerschnittstelle 48
MCP-Serversicherheit (optional) 49
Sicherheit für gehostete Webkonsolen mit ALB + ECS Fargate (optional) 49
Kopflose Sicherheit (bringen Sie Ihren eigenen Webserver mit) (optional) 50
Third-party Test-Frameworks 50
Unterstützte AWS Regionen 52
Von MCP Server unterstützte AWS-Regionen (optional) 53
Kontingente 55
Kontingente für AWS-Services in dieser Lösung 55
CloudFormation AWS-Kontingente 55
Kontingente für Lasttests 55
Gleichzeitige Tests 56
Amazon EC2-Testrichtlinie 56
Amazon-Richtlinie für CloudFront Auslastungstests 57
Überwachung der Lösung nach der Bereitstellung 57
CloudWatch Alarme einrichten 57
Beauftragen Sie einen Experten 58

Kurzfristige AWS Countdown Premium-Engagements für verteilte Lasttests auf AWS 58
Bereitstellen der Lösung 60
Überblick über den Bereitstellungsprozess 61
Bereitstellung mit dem AWS Launch Wizard 62
Mit AWS bereitstellen CloudFormation 62
CloudFormation AWS-Vorlage 62
Starten Sie den Stack (Standard, CloudFront + S3-gehostete Webkonsole) 63
Starten Sie den Stack (von ALB + von ECS Fargate gehostete Webkonsole) 68
Starten Sie den Stack (Headless) 75
In AWS-Regionen GovCloud (USA) bereitstellen 77
Multi-Region Bereitstellung 78
Benutze die Lösung 82
Erstellen Sie ein Testszenario 82
Schritt 1: Allgemeine Einstellungen 82
Schritt 2: Szenariokonfiguration 83
Schritt 3: Verkehrsform 86
Schritt 4: Überprüfen und Erstellen 92
Führen Sie ein Testszenario aus 93
Ansicht der Szenariodetails 93
Arbeitsablauf bei der Testausführung 93
Status der Testläufe 94
Überwachung mit Live-Daten 94
Einen Test stornieren 96
Erkunden Sie die Testergebnisse 97
Übersichtsmetriken für den Testlauf 97
Tabelle mit Testläufen 98
Vergleich der Ausgangswerte 98
Detaillierte Testergebnisse 98
Registerkarte „Fehler“ 100
Registerkarte „Artefakte“ 100
Struktur der S3-Ergebnisse 100
Überwachen Sie mit Logs Insights CloudWatch 101
Verwenden von gespeicherten Abfragen 101
DLT — Zeitplan für den Test 101
DLT — Testfehler 102
DLT — Aufgabenfehler 102

DLT — Orphan Cleanup 103
CloudWatch Alarme 103
Abonnieren Sie Alarmbenachrichtigungen 104
OrphanCleanupFailure 104
MetricFilterCount 105
MCP-Serverintegration 106
Schritt 1: Holen Sie sich den MCP-Endpunkt und das Zugriffstoken 106
Schritt 2: Testen Sie mit MCP Inspector 107
Schritt 3: Konfigurieren Sie KI-Entwicklungsclients 109
Rotation des geheimen MCP-Server-Clients 112
Beispiele für Prompts 117
Integration mit DevOps AWS-Agenten 119
Voraussetzungen 119
Registrieren Sie einen Agentenbereich 120
Untersuchen Sie einen Testlauf 120
Überprüfen der Erkenntnisse 121
Vorgeschlagener Workflow 122
Fehlerbehebung 124
Lösung eines bekannten Problems 124
Probleme bei der Bereitstellung von ALB + ECS Fargate 127
Probleme bei der Headless-Bereitstellung (bringen Sie Ihren eigenen Webserver mit) 128
Kontaktieren Sie AWS Support. 129
Fall erstellen 129
Wie können wir helfen? 129
Zusätzliche Informationen 130
Helfen Sie uns, Ihren Fall schneller zu lösen 130
Löse es jetzt oder kontaktiere uns 130
Aktualisieren Sie die Lösung 131
Aktualisierung mit dem AWS Launch Wizard 131
Update mit AWS CloudFormation 131
Fehlerbehebung bei Updates von Versionen vor v3.3.0 133
Der Pfad der Zip-Datei ändert sich von Version 3 auf Version 4 134
Aktualisierung der regionalen Stacks 135
AWS Systems Manager Manager-Anwendungsmanager 135
Deinstallieren Sie die Lösung 136
Verwendung der AWS-Managementkonsole 136

AWS CloudFormation 136
AWS-STARTASSISTENT 136
Verwenden der AWS-Befehlszeilenschnittstelle 137
Manuelles Löschen einbehaltener Ressourcen 137
Löschen der Amazon S3 S3-Buckets 137
Löschen der Amazon DynamoDB-Tabellen 138
Löschen der Protokollgruppen CloudWatch 138
Löschen der Dashboards CloudWatch 139
Säuberung der Ressourcen von ALB + ECS Fargate nach der Bereitstellung 140
Wiederherstellung von DLT-Daten nach dem Löschen des Stacks 140
Hintergrund 140
Wichtige Einschränkungen 142
Wiederherstellungsprozess 142
Vermeidung von Stack-Drift 148
Was kann nicht wiederhergestellt werden 148
Präventive Maßnahmen 149
Entwicklerhandbuch 150
Quellcode 150
Wartung 150
Versionen 150
Anpassung des Container-Images 151
Tester-Image laden 151
Bild der Webkonsole (nur ALB + ECS Fargate-Vorlage) 152
Tester-Image-URI laden 154
API für verteilte Lasttests 157
ERHALTE /stack-info 159
GET /scenarios 159
POST /szenarien 160
OPTIONEN/Szenarien 162
GET /scenarios/ {testId} 163
POST /scenarios/ {TestID} 165
LÖSCHEN SIE /scenarios/ {testId} 166
OPTIONEN /scenarios/ {testId} 166
GET /scenarios/ {testId} /testruns 168
GET /scenarios/ {testId} /testruns/ {test} RunId 170
LÖSCHEN Sie /scenarios/ {testId} /testruns/ {test} RunId 172

GET /scenarios/ {testId} /baseline 173
PUT /scenarios/ {TestId} /baseline 175
LÖSCHEN Sie /scenarios/ {testId} /baseline 176
GET /tasks 177
OPTIONEN /Aufgaben 178
GET /regions 178
OPTIONEN /Regionen 179
Erhöhen Sie die Container-Ressourcen 180
Erstellen Sie eine neue Version der Aufgabendefinition 180
Aktualisieren Sie die DynamoDB-Tabelle 181
Spezifikation der MCP-Tools 181
Tools lesen 182
Tools schreiben 189
Workflow-Anleitungen 199
DLT-CLI 202
Wichtigste Funktionen 202
Installation und ausführliche Referenz 203
Schnellstart 204
Referenz zum Cron-Ausdruck 205
Akzeptierte Werte 205
Nicht unterstützte Muster 206
Voreingestellte Muster 207
Einschränkungen bei der Terminplanung 208
Referenz 209
Datenerfassung 209
Mitwirkende 209
Glossar 211
Technische Protokolle und Formate 211
Begriffe zum Testen und zur Datenbank 213
AWS- und Systembedingungen 213
Bedingungen für den Belastungstest 214
Überarbeitungen 216
Hinweise 217
..... ccxviii

Automatisieren Sie das Testen Ihrer Softwareanwendungen in großem Maßstab

Distributed Load Testing auf AWS hilft Ihnen dabei, Leistungstests Ihrer Softwareanwendungen im großen Maßstab zu automatisieren, um Engpässe zu identifizieren, bevor Sie Ihre Anwendung veröffentlichen. Diese Lösung simuliert Tausende von verbundenen Benutzern, die HTTP-Anfragen mit kontinuierlicher Geschwindigkeit generieren, ohne dass Server bereitgestellt werden müssen.

Diese Lösung nutzt Amazon Elastic Container Service (Amazon ECS) auf AWS Fargate, um Container bereitzustellen, auf denen Ihre Lasttestsimulationen ausgeführt werden, und bietet die folgenden Funktionen:

- Stellen Sie Amazon ECS auf AWS Fargate-Containern bereit, die unabhängig voneinander ausgeführt werden, um die Ladekapazität Ihrer Anwendung zu testen.
- Simulieren Sie Zehntausende von gleichzeitigen Benutzern in mehreren AWS-Regionen, die kontinuierlich Anfragen generieren.
- Passen Sie Ihre Anwendungstests mithilfe von JMeter -, k6 -, Locust-Testskripten oder einer einfachen HTTP-Endpunktkonfiguration an. Sicherheitsüberlegungen zu den mitgelieferten Frameworks finden Sie unter Test-Frameworks. Third-party
- Planen Sie Lasttests so, dass sie sofort, zu einem future Zeitpunkt oder nach einem wiederkehrenden Zeitplan ausgeführt werden.
- Führen Sie mehrere Lasttests gleichzeitig in verschiedenen Szenarien und Regionen durch.

Dieser Implementierungsleitfaden bietet einen Überblick über die Lösung Distributed Load Testing on AWS, ihre Referenzarchitektur und Komponenten, Überlegungen zur Planung der Bereitstellung und Konfigurationsschritte für die Bereitstellung der Lösung in der Amazon Web Services (AWS) -Cloud. Es enthält Links zu einer CloudFormationAWS-Vorlage, mit der die AWS-Services gestartet und konfiguriert werden, die für die Bereitstellung dieser Lösung erforderlich sind, wobei die bewährten AWS-Methoden für Sicherheit und Verfügbarkeit verwendet werden.

Zu den Zielgruppen für die Nutzung der Funktionen und Fähigkeiten dieser Lösung in ihrer Umgebung gehören IT-Infrastrukturarchitekten, Administratoren und DevOps Fachleute, die über praktische Erfahrung in der Architektur in der AWS-Cloud verfügen.

Verwenden Sie diese Navigationstabelle, um schnell Antworten auf diese Fragen zu finden:

Wenn du willst.	Lesen.
Informieren Sie sich über die Kosten für den Betrieb dieser Lösung. Die geschätzten Kosten für den Betrieb dieser Lösung in der Region USA Ost (Nord-Virginia) belaufen sich auf USD 30,90 pro Monat für AWS-Ressourcen.	Kosten
Machen Sie sich mit den Sicherheitsüberlegungen für diese Lösung vertraut.	Sicherheit
Erfahren Sie, wie Sie Kontingente für diese Lösung einplanen.	Kontingente
Erfahren Sie, welche AWS-Regionen diese Lösung unterstützen.	Unterstützte AWS-Regionen
Erfahren Sie mehr über den optionalen MCP-Server für die Analyse von AI-assisted Lasttests.	MCP-Serverintegration
Erfahren Sie mehr über die verfügbaren Hosting-Optionen für Webkonsolen (CloudFront + S3, ALB + ECS Fargate oder Headless).	Stellen Sie die Lösung bereit
Sehen Sie sich die in dieser Lösung enthaltene CloudFormation AWS-Vorlage an oder laden Sie sie herunter, um die Infrastrukturressourcen (den „Stack“) für diese Lösung automatisch bereitzustellen.	CloudFormation AWS-Vorlage
Greifen Sie auf den Quellcode zu und verwenden Sie optional das AWS Cloud Development Kit (AWS CDK), um die Lösung bereitzustellen.	GitHub Repository

Features

Die Lösung bietet die folgenden Funktionen:

Support mehrerer Test-Frameworks

Unterstützt JMeter-, k6- und Locust-Testskripte sowie einfache HTTP-Endpunkttests, ohne dass benutzerdefinierte Skripts erforderlich sind. Weitere Informationen finden Sie unter Testtypen im Abschnitt Architekturdetails. Sicherheitsüberlegungen zu den mitgelieferten Frameworks finden Sie unter Third-party Test-Frameworks.

Zwei Traffic Shape-Modi

Steuert das Laden entweder von der Lösung oder von Ihrem Skript aus. Im Standardmodus legen Sie die virtuellen Benutzer, den Hochlauf und die Haltedauer fest, und die Lösung wendet sie auf das Framework an, das der Test verwendet. Im nativen Modus führt die Lösung Ihr Skript unter der eigenen Befehlszeile des Frameworks aus und berücksichtigt die vom Skript deklarierte Last. Weitere Informationen finden Sie unter Traffic Shape Modes im Abschnitt Architekturdetails.

Simulation mit hoher Benutzerauslastung

Simuliert Zehntausende gleichzeitiger virtueller Benutzer, um Ihre Anwendung unter realistischen Lastbedingungen einem Stresstest zu unterziehen.

Multi-Region Lastverteilung

Verteilt Belastungstests auf mehrere AWS-Regionen, um den geografisch verteilten Benutzerverkehr zu simulieren und die globale Leistung zu bewerten.

Flexible Testplanung

Plant Tests so, dass sie sofort, zu einem bestimmten future Datum und zu einer bestimmten Uhrzeit oder nach einem wiederkehrenden Zeitplan ausgeführt werden. Dabei werden Cron-Ausdrücke für automatisierte Regressionstests verwendet.

Real-Time Überwachung

Bietet optionales Live-Datenstreaming zur Überwachung des Testfortschritts mit Echtzeitmetriken wie Antwortzeiten, Anzahl virtueller Benutzer und Erfolgsquoten von Anfragen.

Umfassende Testergebnisse

Zeigt detaillierte Testergebnisse mit Leistungskennzahlen, Perzentilen (p50, p90, p95, p99), Fehleranalysen und herunterladbaren Artefakten für die Offline-Analyse an.

Vergleich der Ausgangswerte

Definiert Basis-Testläufe für den Leistungsvergleich, um Verbesserungen oder Regressionen im Laufe der Zeit nachzuverfolgen.

Flexibilität der Endgeräte

Testet jeden HTTP- oder HTTPS-Endpunkt in AWS-Regionen, lokalen Umgebungen oder anderen Cloud-Anbietern.

Flexibles Hosting von Webkonsolen

Wählen Sie aus drei Hosting-Optionen für Webkonsolen: Amazon CloudFront + S3 (Standard), ALB + ECS Fargate mit einer benutzerdefinierten Domain oder Headless (bringen Sie Ihren eigenen Webserver mit). Die Option ALB + ECS Fargate unterstützt Umgebungen mit VPC Block Public Access-Richtlinien oder Anforderungen, in denen das öffentliche Internet nicht gefährdet ist, wie es in regulierten Branchen üblich ist. Das Backend und die Authentifizierung bleiben bei allen Optionen gleich. Weitere Informationen finden Sie unter Bereitstellen der Lösung.

Intuitive Webkonsole

Bietet eine webbasierte Konsole zum Erstellen, Verwalten und Überwachen von Tests, ohne dass eine Befehlszeileninteraktion erforderlich ist.

AI-Assisted Analyse (optional)

Lässt sich über den Model Context Protocol (MCP) -Server in KI-Entwicklungstools integrieren und ermöglicht so eine intelligente Analyse von Lasttestdaten.

Support mehrerer Protokolle

Unterstützt verschiedene Protokolle wie HTTP, HTTPS, JDBC WebSocket, JMS, FTP und gRPC durch benutzerdefinierte Testskripte.

Vorteile

Die Lösung bietet die folgenden Vorteile:

Umfassende Leistungstests

Unterstützt Lasttests, Stresstests und Dauertests, um die Anwendungsleistung unter verschiedenen Bedingungen gründlich zu bewerten.

Früherkennung von Problemen

Identifiziert Leistungsengpässe, Speicherlecks und Skalierbarkeitsprobleme vor der Produktionsbereitstellung und reduziert so das Risiko von Ausfällen.

Real-World Simulation der Nutzung

Simuliert präzise Benutzerverhalten und Verkehrsmuster in der realen Welt, um die Anwendungsleistung unter realistischen Bedingungen zu validieren.

Umsetzbare Performance Insights

Bietet detaillierte Metriken, Perzentile und Fehleranalysen, um das Anwendungsverhalten zu verstehen und Optimierungsmaßnahmen zu steuern.

Automatisierte Test-Workflows

Ermöglicht geplante und wiederkehrende Tests für die kontinuierliche Leistungsüberwachung und Regressionstests ohne manuelles Eingreifen.

Cost-Efficient Infrastruktur

Verwendet serverlose AWS Fargate-Container mit nutzungsabhängiger Bezahlung, sodass keine spezielle Testinfrastruktur und laufende Abonnementgebühren erforderlich sind.

Schnelle Testbereitstellung

Stellt die Testinfrastruktur innerhalb von Minuten bereit und skaliert sie, ohne Server bereitstellen oder verwalten zu müssen.

Einfache Abfrage der Testergebnisse

Lässt sich über einen optionalen Model Context Protocol (MCP) -Server in KI-Entwicklungstools integrieren und ermöglicht Abfragen in natürlicher Sprache und eine intelligente Analyse von Lasttestdaten für schnellere Einblicke und Problembehebung.

Anwendungsfälle

Pre-Production Validierung

Testen Sie Web- und Mobilanwendungen unter produktionsähnlichen Lastbedingungen, bevor Sie eine neue Version auf den Markt bringen, um die Leistung zu überprüfen und Probleme zu identifizieren.

Kapazitätsplanung

Ermitteln Sie die maximale Anzahl gleichzeitiger Benutzer, die Ihre Anwendung mit der aktuellen Infrastruktur unterstützen kann, und ermitteln Sie, wann eine Skalierung erforderlich ist.

Überprüfung bei Spitzenlast

Stellen Sie sicher, dass Ihre Infrastruktur Spitzenlasten, saisonale Verkehrsspitzen oder unerwartete Bedarfsspitzen ohne Leistungseinbußen bewältigen kann.

Optimierung der Leistung

Identifizieren Sie Leistungsengpässe wie langsame Datenbankabfragen, ineffiziente Codeausführung, Netzwerklatenz oder Ressourcenbeschränkungen.

Regressionstests

Planen Sie wiederkehrende Lasttests, um Leistungseinbußen zu erkennen, die durch neue Codebereitstellungen oder Infrastrukturänderungen verursacht werden.

Weltweite Leistungsbeurteilung

Evaluieren Sie die Anwendungsleistung in mehreren geografischen Regionen, um ein einheitliches Benutzererlebnis für ein globales Publikum zu bieten.

API-Lasttests

Testen Sie REST-APIs, GraphQL-Endpunkte oder Microservices, um Antwortzeiten, Durchsatz und Fehlerraten unter Last zu überprüfen.

CI/CD Pipeline-Integration

Integrieren Sie automatisierte Leistungstests in kontinuierliche Integrations- und Bereitstellungspipelines, um Leistungsprobleme schon früh im Entwicklungszyklus zu erkennen.

Third-Party Testen von Diensten

Testen Sie die Leistung und Zuverlässigkeit von APIs oder Diensten von Drittanbietern, von denen Ihre Anwendung abhängt, unter verschiedenen Lastbedingungen.

Konzepte und Definitionen

In diesem Abschnitt werden die wichtigsten Konzepte beschrieben und die für diese Lösung spezifische Terminologie definiert:

Szenario

Testdefinition, einschließlich Testname, Beschreibung, Traffic-Shape-Modus, Anzahl der Aufgaben, Parallelität, AWS-Region, Hochlauf, Hold-for, Testtyp, Zeitplandatum und Wiederholungskonfigurationen.

Traffic Shape-Modus

Welche Seite steuert die Last, die der Test generiert. Im Standardmodus steuert die Lösung dies anhand der Werte, die Sie für Parallelität, Hochlauf und Halten für festgelegt haben. Im systemeigenen Modus wird der Vorgang durch das von Ihnen hochgeladene Skript gesteuert, und die Lösung legt lediglich die Anzahl der Aufgaben und eine Sicherheitsdauer fest. Standard ist die Standardeinstellung. Die folgenden Konzepte für Parallelität, Hochlauf und Wartezeit gelten für den Standardmodus.

Anzahl der Aufgaben

Anzahl der Container, die im Fargate-Cluster gestartet werden, um das Testszenario auszuführen. Zusätzliche Aufgaben werden nicht mehr erstellt, sobald das Kontolimit für Fargate-Ressourcen erreicht ist. Aufgaben, die bereits ausgeführt werden, werden jedoch fortgesetzt.

concurrency

Die Parallelität (Anzahl gleichzeitiger virtueller Benutzer pro Aufgabe). Basierend auf den Standardeinstellungen wird eine Parallelität von 200 empfohlen. Die Parallelität ist durch CPU und Arbeitsspeicher begrenzt. Bei Tests, die auf Apache JMeter basieren, erhöht eine höhere Parallelität den von der JVM für die ECS-Task genutzten Speicher. Die standardmäßige ECS-Aufgabendefinition erstellt Aufgaben mit 4 GB Arbeitsspeicher. Es wird empfohlen, mit niedrigeren Parallelitätswerten für eine Aufgabe zu beginnen und die CloudWatch ECS-Metriken für den Task-Cluster zu überwachen. Weitere Informationen finden Sie in den Kennzahlen zur Nutzung von Amazon ECS-Clustern.

Hochlauf

Der Zeitraum, in dem schrittweise von Null auf das angestrebte Parallelitätsniveau angehoben werden soll.

gedrückt halten für

Der Zeitraum, in dem das angestrebte Parallelitätsniveau nach Abschluss des Hochlaufs beibehalten werden soll.

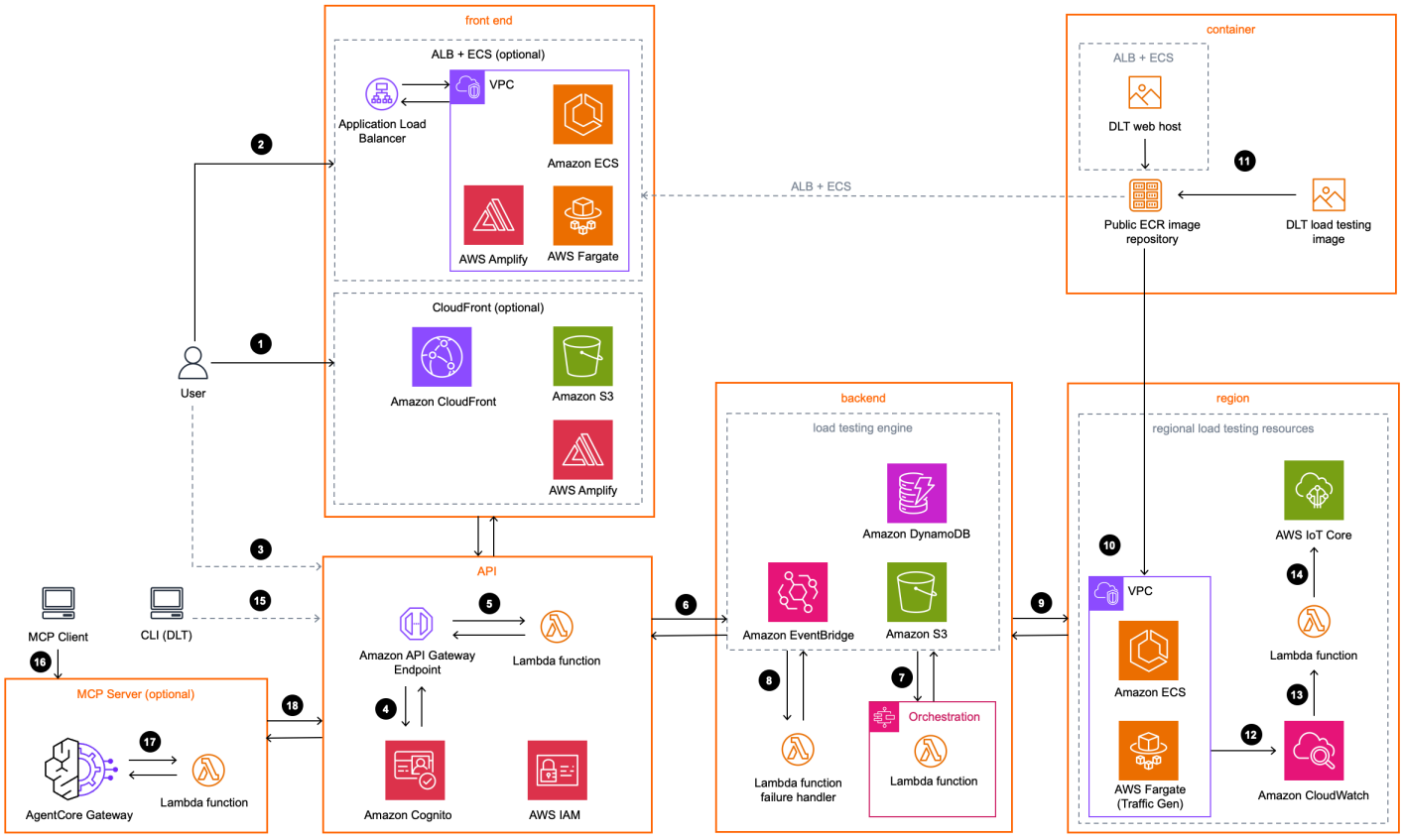
Eine allgemeine Referenz zu AWS-Begriffen finden Sie im AWS-Glossar.

Übersicht über die Architektur

Architekturdiagramm

Durch die Bereitstellung dieser Lösung mit den Standardparametern werden die folgenden Komponenten in Ihrem AWS-Konto bereitgestellt.

Verteilte Lasttests auf der AWS-Architektur



> **Note**
>
> CloudFormation AWS-Ressourcen werden aus Konstrukten des AWS Cloud Development Kit (AWS CDK) erstellt.

Der allgemeine Prozessablauf für die mit der CloudFormation AWS-Vorlage bereitgestellten Lösungskomponenten sieht wie folgt aus:

1. (CloudFront + S3-Hosting-Bereitstellungsoption) Der Konsolenbenutzer greift über Amazon auf die Webkonsole zu CloudFront, die die AWS Amplify Amplify-Anwendung bereitstellt, die in einem Amazon Simple Storage Service (Amazon S3) -Bucket gehostet wird.
2. (ALB + ECS Fargate-Hosting-Bereitstellungsoption) Der Konsolenbenutzer greift über einen Application Load Balancer auf die Webkonsole zu, der den Datenverkehr an die AWS Amplify Amplify-Anwendung weiterleitet, die auf Amazon Elastic Container Service (Amazon ECS) auf AWS Fargate in einer Amazon Virtual Private Cloud (Amazon VPC) ausgeführt wird.
3. (Headless-Bereitstellungsoption) Es wird kein öffentliches Frontend bereitgestellt. Die Lösung stellt die Webkonsole als herunterladbare ZIP-Datei in einem privaten Amazon S3 S3-Bucket bereit. Der Konsolenbenutzer kann von einem selbst gehosteten Webserver aus auf die Konsole zugreifen.
4. Während der Erstkonfiguration erstellt die Lösung einen Standard-Administratorbenutzer im Amazon Cognito Cognito-Benutzerpool und sendet eine E-Mail zur Kontoerstellung an die von Ihnen angegebene E-Mail-Adresse. Der Cognito-Benutzerpool verwaltet den Benutzerzugriff auf die Webkonsole, die REST-API, die CLI und den MCP-Server.
5. Amazon API Gateway ruft AWS Lambda Lambda-Mikroservices auf, die die Geschäftslogik für die Verwaltung von Testdaten und die Ausführung der Tests bereitstellen.
6. Die Microservices interagieren mit Amazon S3, Amazon DynamoDB und Amazon, um Details zu Testszenarien EventBridge zu speichern und Testpläne zu verwalten. Wenn Sie einen Test so planen, dass er zu einem future Zeitpunkt oder in einem wiederkehrenden Intervall ausgeführt wird, erstellen die Microservices einen EventBridge Scheduler-Zeitplan, der den Microservice zur geplanten Zeit aufruft.
7. Um einen Test auszuführen, rufen die Microservices AWS Step Functions auf, das die Testausführung orchestriert.
8. EventBridge Regeln leiten die Fehlerereignisse von Amazon ECS-Aufgaben und Step Functions an eine Lambda-Funktion für die Fehlerbehandlung weiter.
9. Step Functions startet Amazon Elastic Container Service (Amazon ECS) -Aufgaben auf AWS Fargate in jeder AWS-Region, die Sie ausgewählt haben.
10. Jede Aufgabe wird in einer Amazon Virtual Private Cloud (Amazon VPC) in der ausgewählten Region ausgeführt.
11. Lasttestcontainer verwenden ein Amazon Linux 2023-Basis-Image, und das von einer Aufgabe verwendete Image hängt vom Traffic Shape-Modus des Tests ab. Im Standardmodus ist auf dem Image das Taurus-Testautomatisierungs-Framework installiert. Taurus führt Ihren JMeter-, k6-, Locust- oder Simple HTTP Endpoint-Test mit den Ladeparametern aus, die Sie in der Konsole festgelegt haben. Im nativen Modus führt ein spezielles Image für jedes Framework dieses

Framework direkt gegen Ihr Skript aus, ohne Taurus und ohne Ladeparameter aus der Lösung. Einzelheiten zu den Modi finden Sie unter Verkehrsshape-Modi. Informationen zur Bereitstellung der einzelnen Test-Frameworks finden Sie unter Bereitstellung von Test-Frameworks. Die Option ALB + ECS verwendet den Webhost-Container. Die Container-Images werden von AWS in einem öffentlichen Repository von Amazon Elastic Container Registry (Amazon ECR) gehostet.

12. Jede Fargate-Aufgabe schreibt ihre Testergebnisse pro Region in Amazon S3 und sendet Protokolle an Amazon. CloudWatch Wenn alle Regionen abgeschlossen sind, aggregieren die Microservices die Ergebnisse in DynamoDB.
13. Wenn Sie die Live-Datenoption aktivieren, empfängt eine Lambda-Funktion während des CloudWatch Tests Protokolle von Fargate-Aufgaben.
14. Die Lambda-Funktion veröffentlicht die Protokolle zu einem Thema in AWS IoT Core in der Region, in der der Haupt-Stack bereitgestellt wird. Die Webkonsole abonniert das Thema, um während der Testausführung Echtzeit-Metriken anzuzeigen.
15. (Optionaler CLI-Zugriff) Benutzer können die DLT-Befehlszeilenschnittstelle (CLI) lokal installieren, um von ihrem Terminal aus mit der Lösung zu interagieren. Die CLI authentifiziert sich über Cognito und ruft die REST-API direkt auf, was eine skriptbasierte Automatisierung und Integration ermöglicht. CI/CD

> **Note**
>
> In den folgenden Schritten wird die optionale MCP-Serverintegration für AI-assisted die Lasttestanalyse beschrieben. Diese Komponente wird nur bereitgestellt, wenn Sie bei der Bereitstellung der Lösung die Option MCP Server auswählen.

16. Ein MCP-Client (KI-Entwicklungstool) stellt eine Verbindung zum Amazon Bedrock AgentCore Gateway-Endpunkt her, um über das Model Context Protocol auf die Daten der Distributed Load Testing-Lösung zuzugreifen. AgentCore Gateway validiert das Amazon Cognito Cognito-Authentifizierungstoken des Benutzers, um den autorisierten Zugriff auf den MCP-Server zu verifizieren.
17. Nach erfolgreicher Authentifizierung leitet AgentCore Gateway die MCP-Tool-Anfrage an die DLT MCP Server Lambda-Funktion weiter.
18. Die Lambda-Funktion ruft die vorhandene DLT-REST-API auf, um die angeforderten Lasttestdaten abzurufen. Anschließend werden die strukturierten Daten an AgentCore Gateway zurückgegeben, das sie zur AI-assisted Analyse und Auswertung an den MCP-Client zurücksendet.

Überlegungen zum Well-Architected AWS-Design

Diese Lösung nutzt die Best Practices des Well-Architected AWS-Frameworks, das Kunden bei der Entwicklung und dem Betrieb zuverlässiger, sicherer, effizienter und kostengünstiger Workloads in der Cloud unterstützt.

In diesem Abschnitt wird beschrieben, wie die Entwurfsprinzipien und bewährten Methoden des Well-Architected Frameworks dieser Lösung zugute kommen.

Operative Exzellenz

In diesem Abschnitt wird beschrieben, wie wir diese Lösung unter Verwendung der Prinzipien und bewährten Verfahren des Pfeilers Operational Excellence konzipiert haben.

- Alle Ressourcen werden mithilfe von CloudFormation AWS-Vorlagen, die aus AWS-CDK-Konstrukten generiert wurden, als Infrastruktur als Code definiert.
- Die Lösung überträgt Metriken in verschiedenen CloudWatch Phasen, um die Beobachtbarkeit von Lambda-Funktionen, ECS-Aufgaben, S3-Buckets und anderen Lösungskomponenten zu gewährleisten.

Sicherheit

In diesem Abschnitt wird beschrieben, wie wir diese Lösung unter Verwendung der Prinzipien und bewährten Verfahren der Sicherheitssäule konzipiert haben.

- Cognito authentifiziert und autorisiert Webkonsolenbenutzer und API-Anfragen.
- Die gesamte dienstübergreifende Kommunikation verwendet AWS Identity and Access Management (IAM) -Rollen mit geringsten Zugriffsrechten, die nur die erforderlichen Mindestberechtigungen enthalten.
- Der gesamte Datenspeicher, einschließlich S3-Buckets und DynamoDB-Tabellen, verschlüsselt Daten im Ruhezustand mithilfe von AWS-verwalteten Schlüsseln.
- Protokollierung, Rückverfolgung und Versionierung werden gegebenenfalls zu Prüfungs- und Compliance-Zwecken aktiviert.
- Der Netzwerkzugriff ist standardmäßig privat, wobei VPC-Endpunkte aktiviert sind, sofern verfügbar, um den Datenverkehr innerhalb des AWS-Netzwerks aufrechtzuerhalten.

Note

Alle Protokollgruppen, die die Lösung erstellt, speichern Protokolle für 10 Jahre, mit Ausnahme der Webkonsole und der WAF-Protokollgruppen in der ALB + ECS Fargate-Vorlage, die 1 Jahr lang aufbewahrt werden:

Protokollgruppe	Aufbewahrung
AWS Step Functions, Lambda-Laufzeit, ECS-Testaufgabe, Cognito-Benutzerpool und API Gateway Gateway-Z ugriffsprotokolle	10 Jahre
Webkonsole und WAF (nur ALB + ECS Fargate-Vorlage)	1 Jahr
ECS Container Insights (automatisch von ECS erstellt)	Läuft nie ab (Standard)

Sie können diese Aufbewahrungsfristen in der CloudWatch Amazon-Konsole ändern.

Zuverlässigkeit

In diesem Abschnitt wird beschrieben, wie wir diese Lösung unter Verwendung der Prinzipien und bewährten Verfahren der Säule Zuverlässigkeit konzipiert haben.

- Die Lösung verwendet, wo immer möglich, serverlose AWS-Services (Beispiele: Lambda, API Gateway, Amazon S3, AWS Step Functions, Amazon DynamoDB und AWS Fargate), die darauf ausgelegt sind, Hochverfügbarkeit und Wiederherstellung nach einem Serviceausfall zu unterstützen.
- Die gesamte Datenverarbeitung verwendet Lambda-Funktionen oder Amazon ECS auf AWS Fargate.
- Daten werden in DynamoDB und Amazon S3 gespeichert, sodass sie standardmäßig in mehreren Availability Zones gespeichert werden.

Leistungseffizienz

In diesem Abschnitt wird beschrieben, wie wir diese Lösung unter Verwendung der Prinzipien und bewährten Verfahren des Pfeilers Leistungseffizienz konzipiert haben.

- Die Lösung verwendet eine serverlose Architektur, die bei Bedarf horizontal skaliert werden kann.
- Die Lösung kann in jeder Region eingeführt werden, die die AWS-Services in dieser Lösung unterstützt, z. B.: AWS Lambda, Amazon API Gateway, Amazon S3, AWS Step Functions, Amazon DynamoDB, Amazon ECS, AWS Fargate und Amazon Cognito.
- Die Lösung verwendet durchgehend Managed Services, um den betrieblichen Aufwand für die Bereitstellung und Verwaltung von Ressourcen zu reduzieren.
- Die Lösung wird täglich automatisch getestet und bereitgestellt, um Konsistenz zu gewährleisten, wenn sich die AWS-Services ändern. Außerdem wird sie von Lösungsarchitekten und Fachexperten auf Bereiche geprüft, in denen experimentierfreudig und verbesserungswürdig ist.

Kostenoptimierung

In diesem Abschnitt wird beschrieben, wie wir diese Lösung unter Verwendung der Prinzipien und bewährten Methoden der Säule Kostenoptimierung konzipiert haben.

- Die Lösung verwendet eine serverlose Architektur. Daher wird den Kunden nur das in Rechnung gestellt, was sie tatsächlich nutzen.
- Amazon DynamoDB skaliert die Kapazität nach Bedarf, sodass Sie nur für die Kapazität zahlen, die Sie tatsächlich nutzen.
- Amazon ECS auf AWS Fargate ermöglicht es Ihnen, nur für die Rechenressourcen zu zahlen, die Sie nutzen, ohne Vorabkosten.
- AgentCore Gateway dient als kostengünstiger Lambda-based Proxy für die API für verteilte Lasttests, wodurch die Notwendigkeit einer dedizierten Infrastruktur entfällt und die Kosten durch serverlose Pay-per-Request-Preise gesenkt werden.

Nachhaltigkeit

In diesem Abschnitt wird beschrieben, wie wir diese Lösung unter Verwendung der Prinzipien und bewährten Verfahren der Säule Nachhaltigkeit konzipiert haben.

- Die Lösung verwendet verwaltete serverlose Dienste, um die Umweltbelastung der Back-End-Dienste im Vergleich zu kontinuierlich betriebenen lokalen Diensten zu minimieren.
- Serverlose Dienste ermöglichen es Ihnen, nach Bedarf nach oben oder unten zu skalieren.

Einzelheiten zur Architektur

In diesem Abschnitt werden die Komponenten und AWS-Services beschrieben, aus denen diese Lösung besteht, sowie die Architekturdetails dazu, wie diese Komponenten zusammenarbeiten.

Die Lösung Distributed Load Testing on AWS besteht aus drei Komponenten auf hoher Ebene: einem Frontend, einem Backend und einem optionalen MCP-Server.

Frontend

Das Frontend bietet die Schnittstellen für die Interaktion mit der Lösung und umfasst:

- Eine Lasttest-API für programmatischen Zugriff
- Eine Webkonsole zum Erstellen, Planen und Ausführen von Leistungstests
- Ein optionaler MCP-Server zur AI-assisted Analyse von Testergebnissen und Fehlern

Test-API laden

Distributed Load Testing auf AWS konfiguriert Amazon API Gateway so, dass es die RESTful-API der Lösung hostet. Benutzer können über die mitgelieferte Webkonsole, die RESTful-API und den optionalen MCP-Server sicher mit dem Lasttestsystem interagieren. Die API fungiert als „Eingangstür“ für den Zugriff auf Testdaten, die in Amazon DynamoDB gespeichert sind. Sie können die APIs auch verwenden, um auf alle erweiterten Funktionen zuzugreifen, die Sie in die Lösung integriert haben.

Diese Lösung nutzt die Benutzerauthentifizierungsfunktionen der Amazon Cognito Cognito-Benutzerpools. Nach erfolgreicher Authentifizierung eines Benutzers gibt Amazon Cognito ein JSON-Web-Token aus, das der Konsole ermöglicht, Anfragen an die APIs der Lösung (Amazon API Gateway Gateway-Endpunkte) zu senden. HTTPS-Anfragen werden von der Konsole mit dem Autorisierungsheader, der das Token enthält, an die APIs gesendet.

Basierend auf der Anfrage ruft API Gateway die entsprechende AWS Lambda Lambda-Funktion auf, um die erforderlichen Aufgaben mit den in den DynamoDB-Tabellen gespeicherten Daten auszuführen, Testszenarien als JSON-Objekte in Amazon S3 zu speichern, Amazon CloudWatch Metrics-Bilder abzurufen und Testszenarien an die AWS Step Functions Functions-Zustandsmaschine zu senden.

Weitere Informationen zur API der Lösung finden Sie im Abschnitt Distributed Load Testing API dieses Handbuchs.

Web-Konsole

Diese Lösung umfasst eine Webkonsole, mit der Sie Tests konfigurieren und ausführen, laufende Tests überwachen und detaillierte Testergebnisse anzeigen können. Die Konsole ist eine ReactJS-Anwendung, die mit Cloudscape, einem Open-Source-Designsystem für die Erstellung intuitiver Webanwendungen, erstellt wurde. Die Anwendung nutzt AWS Amplify zur Integration mit Amazon Cognito, um Benutzer zu authentifizieren. Die Webkonsole enthält auch eine Option zum Anzeigen von Live-Daten für einen laufenden Test, in dem sie das entsprechende Thema in AWS IoT Core abonniert.

Die Lösung unterstützt drei Hosting-Optionen für Webkonsolen. Die Backend-Architektur und die Cognito-Authentifizierung sind für alle Optionen identisch:

- CloudFront + S3 (Standard) — Die Konsole wird in Amazon S3 gehostet und der Zugriff erfolgt über Amazon CloudFront. Die URL der Webkonsole ist der Name der CloudFront Distributionsdomain, der in den CloudFormation Ausgaben als consoleUrl zu finden ist. Nachdem Sie die CloudFormation Vorlage gestartet haben, erhalten Sie außerdem eine E-Mail mit der URL der Webkonsole und dem Einmalkennwort für die Anmeldung.
- ALB + ECS Fargate — Die Konsole läuft auf einem ECS Fargate-Dienst hinter einem Application Load Balancer mit einem vom Kunden bereitgestellten ACM-Zertifikat und einer benutzerdefinierten Domain. Eine AWS WAF WAF-Web-ACL wird vor der ALB bereitgestellt, um gängige webbasierte Angriffe zu filtern. Diese Option eignet sich für Umgebungen, in denen VPC Block Public Access (BPA) -Richtlinien den Datenverkehr von öffentlichen CloudFront Verteilern verhindern, oder für Organisationen, die keine Anforderungen an die Gefährdung des öffentlichen Internets stellen. Anweisungen zur Bereitstellung finden Sie unter Deploy using ALB + ECS Fargate.
- Headless (bringen Sie Ihren eigenen Webserver mit) — Die Lösung stellt nur das Backend bereit und bietet ein herunterladbares ZIP-Archiv der statischen Ressourcen der Webkonsole. Sie hosten die Konsole auf Ihrem eigenen Webserver. Diese Option eignet sich für Unternehmen, die eine vollständige Netzwerkisolierung ohne öffentliche AWS-Endpunkte benötigen oder die die Konsole auf einer vorhandenen lokalen oder vom Unternehmen verwalteten Infrastruktur hosten müssen. Anweisungen zur Bereitstellung finden Sie unter Bereitstellen mithilfe einer kopflosen Vorlage (bringen Sie Ihren eigenen Webserver mit).

MCP-Server (optional)

Der optionale Model Context Protocol (MCP) -Server bietet eine zusätzliche Schnittstelle für KI-Entwicklungstools, um über Interaktionen in natürlicher Sprache auf Lasttestdaten zuzugreifen und diese zu analysieren. Diese Komponente wird nur bereitgestellt, wenn Sie bei der Bereitstellung der Lösung die Option MCP-Server auswählen.

Mit dem MCP-Server können KI-Agenten mithilfe von Tools wie Kiro CLI, Claude und anderen MCP-compatible KI-Assistenten Testergebnisse abfragen, Leistungskennzahlen analysieren und Einblicke in Ihre Lasttestdaten gewinnen. Ausführliche Informationen zur Architektur und Konfiguration des MCP-Servers finden Sie in diesem Abschnitt unter MCP Server.

Backend

Das Backend besteht aus einer Container-Image-Pipeline und einer Load-Test-Engine, mit der Sie die Last für die Tests generieren. Sie interagieren mit dem Backend über das Frontend. Darüber hinaus werden die für jeden Test gestarteten Aufgaben von Amazon ECS auf AWS Fargate mit einer eindeutigen Test-ID (ID) gekennzeichnet. Diese Test-ID-Tags können Ihnen helfen, die Kosten für diese Lösung zu überwachen. Weitere Informationen finden Sie unter User-Defined Cost Allocation Tags im AWS Billing and Cost Management-Benutzerhandbuch.

Pipeline für Container-Images

Diese Lösung verwendet Container-Images, die mit Amazon Linux 2023 erstellt wurden, als Basis-Image. AWS hostet diese Images in einem öffentlichen Repository von Amazon Elastic Container Registry (Amazon ECR). Die Lösung verwendet sie, um Aufgaben im Amazon ECS on AWS Fargate-Cluster auszuführen.

Welches Bild eine Aufgabe verwendet, hängt vom Traffic-Shape-Modus des Tests ab. Weitere Informationen zu den Modi finden Sie unter Traffic Shape-Modi.

- Der Standardmodus verwendet ein einzelnes Image, auf dem das Taurus Load Testing Framework installiert ist. Taurus ist ein Open-Source-Framework zur Testautomatisierung, das JMeter, k6, Locust und andere Testtools unterstützt. Es wendet die Ladeparameter, die Sie in der Konsole festgelegt haben, auf das Framework an, das der Test verwendet.
- Der native Modus verwendet ein dediziertes Image für jedes Test-Framework (JMeter, k6 oder Locust), ohne Taurus. Jedes Image führt sein Framework direkt gegen Ihr Skript aus.

Weitere Informationen finden Sie im Abschnitt zur Anpassung von Container-Images in diesem Handbuch.

Testen der Framework-Bereitstellung

Die drei unterstützten Test-Frameworks werden zu unterschiedlichen Zeitpunkten im Lösungslebenszyklus bereitgestellt, um ein ausgewogenes Verhältnis zwischen Image-Größe und Versionsflexibilität zu gewährleisten:

- Apache JMeter — Wird während der Stack-Bereitstellung in einem S3-Bucket in Ihrem Konto bereitgestellt und zur Testlaufzeit extrahiert, wenn ein JMeter- oder Simple HTTP Endpoint-Test ausgeführt wird.
- Grafana k6 — Direkt vom Framework-Anbieter heruntergeladen und zur Testlaufzeit extrahiert, nur wenn ein k6-Test ausgeführt wird.
- Locust — Wird zur Build-Zeit im Container-Image installiert und bleibt inaktiv, bis ein Locust-Test ausgeführt wird.

Die Bereitstellung funktioniert in beiden Traffic-Shape-Modi auf die gleiche Weise. Die Images im systemeigenen Modus beziehen jedes Framework zum selben Zeitpunkt im Lebenszyklus wie das Image im Standardmodus. Die folgenden Netzwerk- und Bereitstellungsanforderungen gelten daher für beide.

> **Note**
>
> k6 benötigt zur Testlaufzeit ausgehenden Netzwerkzugriff auf die Grafana k6-Version. Eingeschränkter Ausgang führt dazu, dass k6-Tests beim Herunterladen fehlschlagen.

Infrastruktur wird getestet

Zusätzlich zur CloudFormation Hauptvorlage bietet die Lösung eine regionale Vorlage, mit der die erforderlichen Ressourcen für die Durchführung von Tests in mehreren Regionen bereitgestellt werden können. Die Lösung speichert diese Vorlage in Amazon S3 und stellt in der Webkonsole einen Link dazu bereit. Jeder regionale Stack umfasst eine VPC, einen AWS Fargate-Cluster und eine Lambda-Funktion zur Verarbeitung von Live-Daten.

Weitere Informationen zur Bereitstellung der Testinfrastruktur in weiteren Regionen finden Sie im Abschnitt zur Multi-Region Bereitstellung dieses Handbuchs.

Test-Engine auslasten

Die Distributed Load Testing-Lösung verwendet Amazon Elastic Container Service (Amazon ECS) und AWS Fargate, um Tausende von gleichzeitigen Benutzern in mehreren Regionen zu simulieren und HTTP-Anfragen mit kontinuierlicher Geschwindigkeit zu generieren.

Sie definieren die Testparameter mithilfe der mitgelieferten Webkonsole. Die Lösung verwendet diese Parameter, um ein JSON-Testszenario zu generieren und es in Amazon S3 zu speichern. Weitere Informationen zu Testskripten und Testparametern finden Sie unter Testtypen in diesem Abschnitt.

Eine AWS Step Functions Functions-Zustandsmaschine führt Amazon ECS-Aufgaben in einem AWS Fargate-Cluster aus und überwacht sie. Die AWS Step Functions Functions-Zustandsmaschine umfasst eine AWS-Lambda-Funktion ecr-checker, eine AWS-Lambda-Funktion zur Aufgabenstatusprüfung, eine AWS-Lambda-Funktion zum Task-Runner, eine AWS-Lambda-Funktion zum Abbrechen von Aufgaben und eine AWS-Lambda-Funktion zum Parsen von Ergebnissen. Weitere Informationen zum Workflow finden Sie im Abschnitt zum Ablauf der Testausführung in diesem Handbuch. Weitere Informationen zu Testergebnissen finden Sie im Abschnitt „Testergebnisse untersuchen“ dieses Handbuchs. Weitere Informationen zum Ablauf der Teststornierung finden Sie im Abschnitt Einen Test stornieren dieses Handbuchs.

Wenn Sie Live-Daten auswählen, initiiert die Lösung in jeder Region eine Lambda-Funktion für Echtzeitdaten-Publisher anhand der CloudWatch Protokolle, die den Fargate-Aufgaben in dieser Region entsprechen. Die Lösung verarbeitet und veröffentlicht dann die Daten zu einem Thema in AWS IoT Core in der Region, in der Sie den Haupt-Stack gestartet haben. Weitere Informationen finden Sie im Abschnitt Überwachung mit Live-Daten in diesem Handbuch.

MCP-Server

Die optionale Serverintegration des Model Context Protocol (MCP) ermöglicht es KI-Agenten, über Interaktionen in natürlicher Sprache programmgesteuert auf Ihre Lasttestdaten zuzugreifen und diese zu analysieren. Diese Komponente wird nur bereitgestellt, wenn Sie bei der Bereitstellung der Lösung die Option MCP-Server auswählen.

Der MCP-Server fungiert als Brücke zwischen KI-Entwicklungstools und Ihrer DLT-Bereitstellung und bietet eine standardisierte Schnittstelle für die intelligente Analyse der Ergebnisse von Leistungstests. Die Architektur integriert mehrere AWS-Services, um eine sichere, skalierbare Schnittstelle für Interaktionen mit KI-Agenten zu schaffen:

Amazon Bedrock Gateway AgentCore

Amazon Bedrock AgentCore Gateway ist ein vollständig verwalteter Service, der standardisiertes Hosting und Protokollmanagement für MCP-Server bietet. In dieser Lösung dient AgentCore Gateway als öffentlicher Endpunkt, mit dem KI-Agenten eine Verbindung herstellen, wenn sie Zugriff auf Ihre Lasttestdaten anfordern.

Der Service wickelt die gesamte MCP-Protokollkommunikation ab, einschließlich der Erkennung von Tools, der Validierung von Authentifizierungstoken und der Weiterleitung von Anfragen. AgentCore Gateway arbeitet als Mehrmandantendienst mit integrierten Sicherheitsvorkehrungen gegen allgemeine Bedrohungen für öffentliche Endgeräte und validiert gleichzeitig die Signaturen und Ansprüche von Cognito-Tokens für jede Anfrage.

DLT MCP Server Lambda

Die DLT MCP Server Lambda-Funktion ist eine benutzerdefinierte serverlose Komponente, die MCP-Anfragen von AI-Agenten verarbeitet und sie in Abfragen für Ihre DLT-Ressourcen übersetzt.

Diese Lambda-Funktion fungiert als Informationsebene der MCP-Integration. Sie ruft Testergebnisse aus DynamoDB-Tabellen ab, greift auf Leistungsartefakte zu, die in S3-Buckets gespeichert sind, und fragt CloudWatch Logs nach detaillierten Ausführungsinformationen ab. Sie wandelt DLT-Rohdaten in strukturierte Formate um, AI-friendly die Agenten einfach interpretieren und analysieren können. Die IAM-Berechtigungen der Funktion richten sich nach dem MCP-Serverzugriffsmodus-Parameter, den Sie bei der Bereitstellung festgelegt haben. Der `ReadOnly` Standardmodus erlaubt nur `GET` Anfragen an die API der Lösung. `ReadWrite`Der Modus erlaubt auch`POST`,`PUT`, und`DELETE`.

Integration der Authentifizierung

Das Authentifizierungssystem nutzt Ihre bestehende Cognito-Benutzerpool-Infrastruktur, um konsistente Zugriffskontrollen sowohl für die Webkonsole als auch für die MCP-Serverschnittstellen aufrechtzuerhalten.

Diese Integration verwendet die tokenbasierte Authentifizierung auf OAuth 2.0-Basis. Benutzer authentifizieren sich einmal über den Cognito-Anmeldevorgang und erhalten Token, die sowohl für Benutzeroberflächeninteraktionen als auch für den MCP-Serverzugriff funktionieren. Das System verwendet dieselben Berechtigungsgrenzen und Zugriffskontrollen wie die Weboberfläche und ist so konzipiert, dass der Zugriff durch KI-Agenten auf dieselben Lasttestdaten beschränkt wird, auf die sie über die Konsole zugreifen können.

AWS-Services in dieser Lösung

Die folgenden AWS-Services sind in dieser Lösung enthalten:

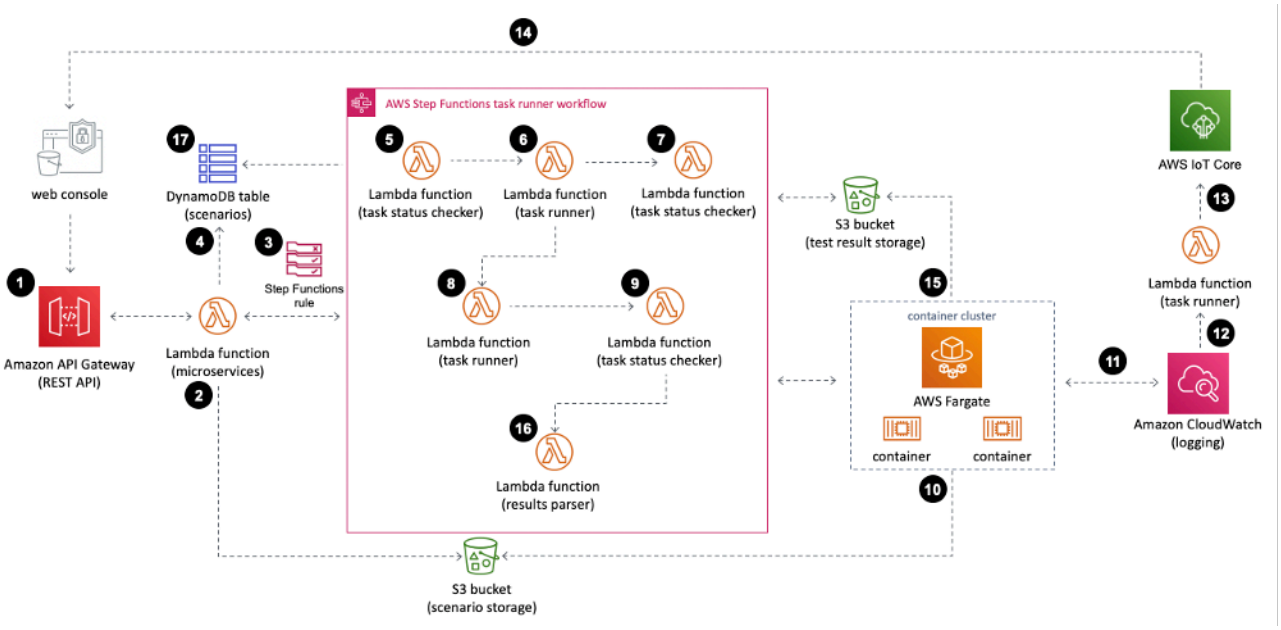
AWS Service	Description
Amazon API Gateway	Kern. Hostet REST-API-Endpunkte in der Lösung.
AWS CloudFormation	Kern. Verwaltet Bereitstellungen für die Lösungsinfrastruktur.
Amazon CloudFront	Core (nur Standardbereitstellung). Stellt die in Amazon S3 gehosteten Webinhalte bereit, wenn die Standardhosting-Option CloudFront + S3 verwendet wird.
Elastic Load Balancing (Application Load Balancer)	Core (nur Bereitstellung von ALB + ECS Fargate). Dient der Webkonsole, wenn die Bereitstellungsoption ALB+ ECS Fargate verwendet wird.
AWS WAF	Unterstützend, optional (nur Bereitstellung mit ALB + ECS Fargate). Bietet Firewall-Schutz für Webanwendungen für die von ALB und ECS gehostete Webkonsole.
Amazon CloudWatch	Kern. Speichert die Lösungsprotokolle und Metriken.
Amazon Cognito	Kern. Verwaltet die Benutzerverwaltung und Authentifizierung für die API.
Amazon-DynamoDB	Kern. Speichert Bereitstellungsinformationen und testet Szenariodetails und Ergebnisse.
Amazon Elastic Container Service	Kern. Stellt unabhängige Amazon ECS-Aufgaben auf AWS Fargate-Containern bereit und verwaltet sie.
AWS Fargate	Kern. Hostet die Amazon ECS-Container der Lösung
AWS Identity and Access Management	Kern. Kümmert sich um die Verwaltung von Benutzerrollen und Berechtigungen.

AWS Service	Description
AWS Lambda	Kern. Stellt Logik für die Implementierung von APIs, das Analysieren von Testergebnissen und das Starten von workers/leader Aufgaben bereit.
AWS Step Functions	Kern. Orchestriert die Bereitstellung von Amazon ECS-Containern für AWS Fargate-Aufgaben in den angegebenen Regionen
AWS Amplify	Unterstützend. Stellt eine von AWS Amplify betriebene Webkonsole bereit.
Amazon EventBridge	Unterstützend. Leitet Fehlerereignisse für Aufgaben und Step Functions von Amazon ECS an Lambda-Funktionen für die Fehlerbehandlung weiter und plant mithilfe von EventBridge Scheduler, dass Tests automatisch an einem bestimmten Datum oder an wiederkehrenden Terminen beginnen.
Amazon Elastic Container Registry	Unterstützend. Hostet das Container-Image in einem öffentlichen ECR-Repository.
AWS IoT Core	Unterstützend. Ermöglicht die Anzeige von Live-Daten für einen laufenden Test, indem Sie das entsprechende Thema in AWS IoT Core abonnieren.
AWS Systems Manager	Unterstützend. Bietet Ressourcenüberwachung und Visualisierung von Ressourcenoperationen und Kostendaten auf Anwendungsebene.
Amazon S3	Unterstützend. Hostet die statischen Webinhalte, Protokolle, Metriken und Testdaten.
Amazon Virtual Private Cloud	Unterstützend. Enthält die Amazon ECS-Container der Lösung, die auf AWS Fargate ausgeführt werden.
Amazon Bedrock AgentCore	Unterstützend, optional. Hostet den optionalen Remote Model Context Protocol (MCP) -Server der Lösung für die Integration von KI-Agenten mit der API.

So funktioniert Distributed Load Testing auf AWS

Die folgende detaillierte Aufschlüsselung zeigt die Schritte, die zur Ausführung eines Testszenarios erforderlich sind.

Test-Workflow



1. Sie verwenden die Webkonsole, um ein Testszenario mit den Konfigurationsdetails an die API der Lösung zu senden.
2. Die Konfiguration des Testszenarios wird als JSON-Datei () in den Amazon Simple Storage Service (Amazon S3`s3://<bucket-name>/test-scenarios/<$TEST_ID>/<$TEST_ID>.json`) hochgeladen.
3. Ein AWS Step Functions Functions-Zustandsmaschine wird mit der Test-ID, der Anzahl der Aufgaben, dem Testtyp und dem Dateityp als Eingabe für die AWS Step Functions Functions-Zustandsmaschine ausgeführt. Wenn der Test geplant ist, erstellt er zunächst einen Amazon EventBridge Scheduler-Zeitplan, der AWS Step Functions am angegebenen Datum auslöst. Weitere Informationen zur Planung finden Sie unter Planung von Tests in diesem Abschnitt.
4. Konfigurationsdetails werden in der Amazon DynamoDB-Szenario-Tabelle gespeichert.
5. Im Task-Runner-Workflow von AWS Step Functions prüft die AWS-Lambda-Funktion zur Überprüfung des Aufgabenstatus, ob Amazon Elastic Container Service (Amazon ECS) -Aufgaben bereits für dieselbe Test-ID ausgeführt werden. Wenn festgestellt wird, dass Aufgaben mit derselben Test-ID ausgeführt werden, führt dies zu einem Fehler. Wenn im AWS Fargate-Cluster

keine Amazon ECS-Aufgaben ausgeführt werden, gibt die Funktion die Test-ID, die Anzahl der Aufgaben und den Testtyp zurück.

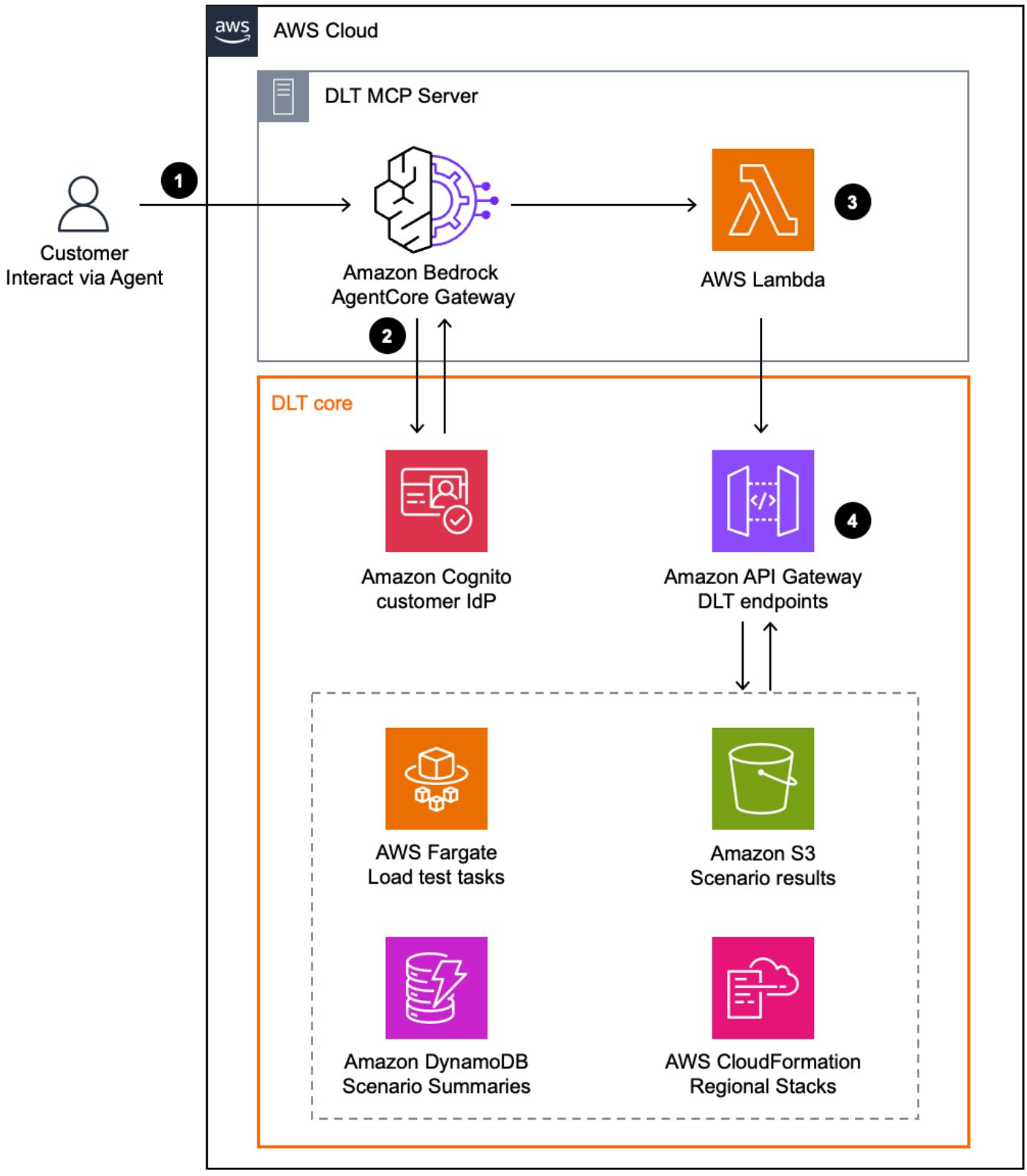
6. Die Task-Runner-Funktion AWS Lambda ruft die Aufgabendetails aus dem vorherigen Schritt ab und führt die Amazon ECS-Worker-Aufgaben im AWS Fargate-Cluster aus. Die Amazon ECS-API verwendet die RunTask Aktion, um die Worker-Aufgaben auszuführen. Diese Worker-Aufgaben werden gestartet und warten dann auf eine Startnachricht von der Leader-Aufgabe, um mit dem Test zu beginnen. Die RunTask Aktion ist auf 10 Aufgaben pro Definition begrenzt. Wenn die Anzahl Ihrer Aufgaben mehr als 10 beträgt, wird die Aufgabendefinition mehrmals ausgeführt, bis alle Worker-Aufgaben gestartet wurden. Die Funktion generiert auch ein Präfix, um den aktuellen Test in der AWS Lambda Lambda-Funktion zur Ergebnisanalyse zu unterscheiden.
7. Die AWS Lambda Lambda-Funktion task-status-checker prüft, ob alle Amazon ECS-Worker-Tasks mit derselben Test-ID ausgeführt werden. Wenn die Aufgaben noch bereitgestellt werden, wartet sie eine Minute und prüft erneut. Sobald alle Amazon ECS-Aufgaben ausgeführt werden, werden die Test-ID, die Anzahl der Aufgaben, der Testtyp, alle Aufgaben-IDs und das Präfix zurückgegeben und an die Task-Runner-Funktion übergeben.
8. Die Task-Runner-Funktion AWS Lambda wird erneut ausgeführt. Diesmal wird eine einzelne Amazon ECS-Task gestartet, die als Leader-Knoten fungiert. Diese ECS-Aufgabe sendet eine Starttestnachricht an jede der Worker-Aufgaben, damit die Tests gleichzeitig gestartet werden können.
9. Die AWS Lambda Lambda-Funktion task-status-checker überprüft erneut, ob Amazon ECS-Aufgaben mit derselben Test-ID ausgeführt werden. Wenn Aufgaben noch ausgeführt werden, wartet sie eine Minute und prüft erneut. Sobald keine Amazon ECS-Aufgaben ausgeführt werden, werden die Test-ID, die Anzahl der Aufgaben, der Testtyp und das Präfix zurückgegeben.
10. Wenn die Task-Runner-Funktion AWS Lambda die Amazon ECS-Aufgaben im AWS Fargate-Cluster ausführt, lädt jede Aufgabe die Testkonfiguration von Amazon S3 herunter und startet den Test.
11. Sobald die Tests ausgeführt werden, werden die durchschnittliche Antwortzeit, die Anzahl der gleichzeitigen Benutzer, die Anzahl der erfolgreichen Anfragen und die Anzahl der fehlgeschlagenen Anfragen für jede Aufgabe in Amazon protokolliert CloudWatch und können in einem CloudWatch Dashboard eingesehen werden.
12. Wenn Sie Live-Daten in den Test aufgenommen haben, filtert die Lösung Echtzeit-Testergebnisse CloudWatch mithilfe eines Abonnementfilters. Dann übergibt die Lösung die Daten an eine Lambda-Funktion.
13. Die Lambda-Funktion strukturiert dann die empfangenen Daten und veröffentlicht sie in einem AWS IoT Core Core-Thema.

14. Die Webkonsole abonniert das AWS IoT Core Core-Thema für den Test und empfängt die zum Thema veröffentlichten Daten, um die Echtzeitdaten während der Ausführung des Tests grafisch darzustellen.
15. Wenn der Test abgeschlossen ist, exportieren die Container-Images einen detaillierten Bericht als XML-Datei nach Amazon S3. Jede Datei erhält eine UUID für den Dateinamen. Zum Beispiel s3://amzn-s3-demo - bucket/test -scenarios/ **<**$TEST_ID**>** /results/ **<**$UUID**>** .xml.
16. Wenn die XML-Dateien auf Amazon S3 hochgeladen werden, liest die AWS Lambda Lambda-Funktion für die Ergebnisanalyse die Ergebnisse in den XML-Dateien, beginnend mit dem Präfix, analysiert und aggregiert alle Ergebnisse zu einem zusammengefassten Ergebnis.
17. Die AWS Lambda Lambda-Funktion für Ergebnisanalyse schreibt das Gesamtergebnis in eine Amazon DynamoDB-Tabelle.

MCP-Server-Workflow (optional)

Wenn Sie die optionale MCP-Serverintegration einsetzen, können KI-Agenten über den folgenden Workflow auf Ihre Lasttestdaten zugreifen und diese analysieren:

MCP-Server-Arbeitsablauf


AWS Cloud
DLT MCP Server
Customer
Interact via Agent
1
Amazon Bedrock
AgentCore Gateway
2
AWS Lambda
3
DLT core
Amazon Cognito
customer IdP
Amazon API Gateway
DLT endpoints
4
AWS Fargate
Load test tasks
Amazon S3
Scenario results
Amazon DynamoDB
Scenario Summaries
AWS CloudFormation
Regional Stacks

1. Kundeninteraktion — Der Kunde interagiert über einen KI-Agenten mit der Distributed Load Testing-Lösung. Der Agent stellt eine Verbindung zum MCP-Endpunkt her, um Zugriff auf Lasttestdaten anzufordern.
2. Autorisierung — AgentCore Gateway validiert das Amazon Cognito Cognito-Authentifizierungstoken des Benutzers, um sicherzustellen, dass der Benutzer berechtigt ist, auf den DLT MCP-Server zuzugreifen. Die verfügbaren Tools hängen vom Parameter MCP-Serverzugriffsmodus ab.
3. Toolaufruf — AgentCore Gateway leitet autorisierte MCP-Tool-Anfragen an die DLT MCP Server Lambda-Funktion weiter. Die Lambda-Funktion implementiert die Tools, die KI-Agenten zum Abrufen von Lasttestinformationen verwenden.
4. API-Zugriff — Die DLT MCP Server Lambda-Funktion ruft die vorhandenen DLT API Gateway Gateway-Endpunkte auf, um Testdaten von DynamoDB und Amazon S3 abzurufen. Im `ReadOnly` Standardzugriffsmodus bietet die Funktion schreibgeschützte Tools zum Abrufen von Testszenarien, Testläufen, Basisvergleichen und Testlaufartefakten. Im `ReadWrite` Modus bietet sie auch Tools zum Erstellen, Ändern, Löschen und Starten von Testszenarien. Weitere Informationen zu den verfügbaren Tools und ihren Parametern finden Sie in der MCP-Tools-Spezifikation im Developer Guide.

Die MCP-Serverintegration nutzt die bestehende DLT-Infrastruktur (API Gateway, Cognito, DynamoDB, S3), um sicheren Zugriff auf Testdaten für Analysen und Erkenntnisse zu ermöglichen. AI-powered

Designüberlegungen

In diesem Abschnitt werden wichtige Designentscheidungen und Konfigurationsoptionen für die Distributed Load Testing on AWS-Lösung beschrieben, einschließlich unterstützter Anwendungen, Testtypen, Planungsoptionen und Überlegungen zur Bereitstellung.

Unterstützte Anwendungen

Diese Lösung unterstützt das Testen von cloudbasierten Anwendungen und lokalen Anwendungen, sofern Sie über eine Netzwerkverbindung zwischen Ihrem AWS-Konto und Ihrer Anwendung verfügen. Die Lösung unterstützt APIs, die HTTP- oder HTTPS-Protokolle verwenden.

Testtypen

Distributed Load Testing auf AWS unterstützt mehrere Testtypen: einfache HTTP-Endpunkttests, JMeter, k6 und Locust. Jeder Testtyp mit Ausnahme des einfachen HTTP-Endpunkts kann in einem der beiden Traffic-Shape-Modi ausgeführt werden. Weitere Informationen finden Sie unter Traffic Shape-Modi.

> **Note**
>
> Die Lösung verteilt JMeter, k6 und Locust als Komponenten von Drittanbietern ohne Änderungen. Sicherheitsüberlegungen, Patching-Optionen und Lizenzinformationen finden Sie unter Test-Frameworks. Third-party

Einfache HTTP-Endpunkttests

Die Webkonsole bietet eine Schnittstelle zur Konfiguration von HTTP-Endpunkten, mit der Sie jeden HTTP- oder HTTPS-Endpunkt testen können, ohne benutzerdefinierte Skripts schreiben zu müssen. Sie definieren die Endpunkt-URL, wählen die HTTP-Methode (GET, POST, PUT, DELETE usw.) aus einem Dropdownmenü aus und fügen optional benutzerdefinierte Anforderungsheader und Body-Payloads hinzu. Diese Konfiguration ermöglicht es Ihnen, APIs mit benutzerdefinierten Autorisierungstoken, Inhaltstypen oder anderen HTTP-Headern und Anforderungstexten zu testen, die für Ihre Anwendung erforderlich sind.

Wenn Sie einen HTTP-Endpunkt konfigurieren, konvertiert die Lösung Ihre Konfiguration in einen Testplan, der von der mitgelieferten Apache JMeter-Binärdatei über das Taurus-Framework ausgeführt wird. Einfache HTTP-Endpunkttests akzeptieren kein Testarchiv, sodass sie die mitgelieferte JMeter-Binärdatei oder -Plugins nicht überschreiben können. Wenn Sie HTTP-Endpunkttests mit einem gepatchten JMeter ausführen müssen, verwenden Sie stattdessen den JMeter-Testtyp. Informationen zur Sicherheit finden Sie unter Apache JMeter.

Da die Lösung den Testplan für diesen Testtyp generiert, werden Simple HTTP Endpoint-Tests nur im Standardmodus ausgeführt. Für den systemeigenen Modus ist ein Skript erforderlich, das Sie hochladen. Weitere Informationen finden Sie unter Traffic Shape Modes.

JMeter-Tests

Wenn Sie ein Testszenario mit der Webkonsole erstellen, können Sie ein JMeter-Testskript hochladen. Die Lösung lädt das Skript in den S3-Bucket des Szenarios hoch. Wenn Amazon ECS-Aufgaben ausgeführt werden, laden sie das JMeter-Skript von S3 herunter und führen den Test aus.

> **Important**
>
> Im Standardmodus kann Ihr JMeter-Skript Parallelität (virtuelle Benutzer), Transaktionsraten (TPS), Anlaufzeiten und andere Ladeparameter definieren. Die Lösung überschreibt sie alle mit den Werten, die Sie bei der Testerstellung im Traffic Shape-Bildschirm angeben. Diese Konfiguration steuert die Anzahl der Aufgaben, die Parallelität (virtuelle Benutzer pro Aufgabe), die Dauer des Hochlaufs und die Haltedauer für die Testausführung.
> Im systemeigenen Modus wird die Lösung `jmeter -n -t` anhand Ihres Skripts ausgeführt und übergibt keine Ladeparameter. Ihre Thread-Gruppen und Timer werden genau so ausgeführt, wie sie erstellt wurden. Weitere Informationen finden Sie unter Traffic Shape-Modi.

Wenn Sie über JMeter-Eingabedateien verfügen, können Sie die Eingabedateien zusammen mit dem JMeter-Skript komprimieren. Sie können die Zip-Datei auswählen, wenn Sie ein Testszenario erstellen.

Wenn Sie Plugins einbeziehen möchten, werden alle .jar-Dateien, die in einem /plugins-Unterverzeichnis der mitgelieferten ZIP-Datei enthalten sind, in das JMeter-Erweiterungsverzeichnis kopiert und stehen dort für Auslastungstests zur Verfügung.

> **Note**
>
> Wenn Sie JMeter-Eingabedateien in Ihre JMeter-Skriptdatei aufnehmen, müssen Sie den relativen Pfad der Eingabedateien in Ihre JMeter-Skriptdatei aufnehmen. Außerdem müssen sich die Eingabedateien im relativen Pfad befinden. Wenn sich Ihre JMeter-Eingabedateien und die Skriptdatei beispielsweise im home/user Verzeichnis/befinden und Sie auf die Eingabedateien in der JMeter-Skriptdatei verweisen, muss der Pfad der Eingabedateien lauten. /INPUT_FILES. Wenn Sie stattdessen/home/user/INPUT_FILES verwenden, schlägt der Test fehl, da er die Eingabedateien nicht finden kann.

Wenn Sie JMeter-Plugins einbeziehen, müssen die .jar-Dateien in einem Unterverzeichnis namens / plugins im Stammverzeichnis der ZIP-Datei gebündelt werden. Relativ zum Stammverzeichnis der Zip-Datei muss der Pfad zu den JAR-Dateien sein. /plugins/BUNDLED_PLUGIN.jar.

Weitere Informationen zur Verwendung von JMeter-Skripten finden Sie im JMeter-Benutzerhandbuch.

k6-Tests

Die Lösung unterstützt Tests auf Basis des K6-Frameworks. Sie können die K6-Testdatei zusammen mit allen erforderlichen Eingabedateien in eine Archivdatei hochladen. Die Webkonsole zeigt eine Meldung zur Lizenzbestätigung an, wenn Sie einen neuen K6-Test erstellen. Lizenz- und Sicherheitsdetails finden Sie unter Grafana k6.

> **Important**
>
> Im Standardmodus kann Ihr k6-Skript Parallelität (virtuelle Benutzer), Stufen, Schwellenwerte und andere Ladeparameter definieren. Die Lösung überschreibt sie alle mit den Werten, die Sie bei der Testerstellung im Traffic Shape-Bildschirm angeben. Diese Konfiguration steuert die Anzahl der Aufgaben, die Parallelität (virtuelle Benutzer pro Aufgabe), die Dauer des Hochlaufs und die Haltedauer für die Testausführung.
> Im systemeigenen Modus wird die Lösung `k6 run` anhand Ihres Skripts ausgeführt und übergibt keine Ladeparameter. k6 wendet Ihren Optionsblock, Ihre Szenarien, Stufen und Schwellenwerte genau so an, wie sie geschrieben wurden. Weitere Informationen finden Sie unter Traffic Shape Modes.

Heuschrecken-Tests

Die Lösung unterstützt Tests auf Basis des Locust Frameworks. Sie können die Locust-Testdatei zusammen mit allen erforderlichen Eingabedateien in eine Archivdatei hochladen.

> **Important**
>
> Im Standardmodus kann Ihr Locust-Skript Parallelität (Benutzeranzahl), Spawn-Rate und andere Ladeparameter definieren. Die Lösung überschreibt sie alle mit den Werten, die Sie bei der Testerstellung im Traffic Shape-Bildschirm angeben. Diese Konfiguration steuert die Anzahl der Aufgaben, die Parallelität (virtuelle Benutzer pro Aufgabe), die Dauer des Hochlaufs und die Haltedauer für die Testausführung.

Im systemeigenen Modus wird die Lösung `locust --headless` anhand Ihres Skripts ausgeführt und übergibt keine Ladeparameter. Locust wendet Ihre `LoadTestShape` Klassen und gewichteten Aufgabensätze genau so an, wie sie geschrieben wurden. Ihr Skript darf nicht festgelegt werden`processes`, da die Lösung Anfragen nur zählt, wenn Locust als ein einziger Prozess ausgeführt wird. Weitere Informationen finden Sie unter Traffic Shape Modes.

Benennung von Testskripten

Wenn Sie eine einzelne `.py` Datei hochladen, speichert die Lösung sie unter der Test-ID und verweist direkt darauf, sodass die Datei einen beliebigen Namen haben kann. Wenn Sie ein `.zip` Archiv hochladen, durchsucht die Lösung das Archiv nach einer Datei mit dem Namen`locustfile.py`. Wenn das Archiv ein Python-Skript unter einem anderen Namen enthält, schlägt der Test beim Start des Containers mit der Meldung fehlNo `test script (.py) in zip file`.

Benutzerdefinierte Python-Abhängigkeiten

Der Lasttestcontainer enthält Locust und seine Abhängigkeiten. Es enthält keine Python-Pakete von Drittanbietern. Wenn Ihr Locust-Skript ein Paket importiert, das nicht im Container vorhanden ist, schlägt der Test mit `ModuleNotFoundError` fehl. Um zusätzliche Pakete verfügbar zu machen, fügen Sie eine `requirements.txt` Datei in das Stammverzeichnis Ihres `.zip` Archivs ein. Der Container installiert die in aufgelisteten Pakete, `requirements.txt` bevor der Test beginnt.

Sie können Abhängigkeiten auf zwei Arten angeben:

Von PyPI installieren

Schließt nur eine Datei ein`requirements.txt`. Der Container installiert die in `requirements.txt` PyPI aufgelisteten Pakete beim Start der Aufgabe. Dies erfordert ausgehenden Internetzugang aus den Subnetzen, in denen die Lasttestaufgaben ausgeführt werden.

Die Installation erfolgt über die mitgelieferten Räder (offline)

Schließt eine `requirements.txt` Datei und ein `packages` Unterverzeichnis mit Python Wheel (`.whl`) -Dateien ein. Der Container wird nur von den mitgelieferten Rädern aus installiert und berührt PyPI nicht. Diese Option funktioniert in Umgebungen ohne ausgehenden Internetzugang.

Bundling Wheels legt auch die genauen Paketversionen fest, sodass eine neue PyPI-Version Ihre Testumgebung nicht zwischen den Läufen ändern kann.

Das folgende Beispiel zeigt das Archiv-Layout:

```
my-test.zip
### locustfile.py         # Required — must use this name
### requirements.txt      # Optional — packages to install
### packages/             # Optional — wheels, for offline install only
    ### *.whl
```

Beides `requirements.txt` und das `packages` Unterverzeichnis müssen sich `locustfile.py` nebenan im Stammverzeichnis des Archivs befinden. Lassen Sie beide weg, wenn Ihr Skript nur Pakete importiert, die der Container bereits bereitstellt. Das `packages` Unterverzeichnis wird nur zusammen mit einer `requirements.txt` Datei wirksam. Für sich genommen wird es ignoriert und es werden keine Pakete installiert.

> **Transitive Abhängigkeiten**
>
> Wenn Sie Räder bündeln, `requirements.txt` müssen Sie alle Pakete auflisten, die Ihre Abhängigkeiten benötigen, nicht nur die Pakete, die Sie direkt importieren. Die Offline-Installation kontaktiert PyPI nicht. Eine fehlende transitive Abhängigkeit führt dazu, dass die Installation fehlschlägt und die Aufgabe vor dem Start des Tests beendet wird.

Räder für den Container vorbereiten

Der Lasttestcontainer läuft unter Linux auf der x86_64-Architektur mit Python 3.11. Räder, die für ein anderes Betriebssystem, eine andere Architektur oder eine andere Python-Version kompiliert wurden, werden nicht installiert. In reinem Python geschriebene Pakete werden als plattformunabhängige Räder verteilt und funktionieren überall, aber Pakete, die kompilierte Erweiterungen enthalten, benötigen ein Rad, das für die Plattform des Containers gebaut wurde. Da der Container keinen Compiler enthält, kann er beim Start der Aufgabe keine Quelldistribution erstellen.

Führen Sie den folgenden Befehl aus, um Räder herunterzuladen, die mit der Plattform des Containers kompatibel sind. Sie können diesen Befehl von jedem Betriebssystem aus ausführen, einschließlich macOS und Windows. Fügen Sie dann das resultierende `packages` Verzeichnis in Ihr Archiv ein:

```
pip download -r requirements.txt \
  --dest packages \
  --platform manylinux2014_x86_64 \
  --python-version 3.11 \
  --only-binary=:all:
```

Die `--python-version` Optionen `--platform` und zielen auf den Container ab, nicht auf den Computer, auf dem Sie den Befehl ausführen. Die `--only-binary=:all:` Option führt dazu, dass der Befehl fehlschlägt, anstatt im Hintergrund auf eine Quelldistribution zurückzugreifen, die der Container nicht erstellen kann. Das `manylinux2014` Tag spezifiziert ein Rad, das mit Glibc 2.17 und höher kompatibel ist, was die Version des Containers beinhaltet.

Um ein Paket zu bündeln, das Sie selbst verwalten, bauen Sie ein Wheel aus dem Quellverzeichnis Ihres Pakets mit `pip wheel . --wheel-dir packages` und fügen Sie dann den Paketnamen hinzu. `requirements.txt`

Modi der Verkehrsform

Jeder Test wird in einem von zwei Traffic-Shape-Modi, Standard oder Nativ, ausgeführt. Der Modus bestimmt drei Dinge: welche Seite den Ladevorgang steuert, welches Container-Image die Fargate-Aufgaben verwenden und welche Ladeparameter die Lösung an das Test-Framework sendet. Anleitungen zur Auswahl eines Modus bei der Erstellung eines Tests finden Sie unter Verkehrsshape-Modi im Abschnitt Verwenden der Lösung.

Standardmodus

Die Fargate-Aufgaben verwenden das Image, auf dem Taurus installiert ist. Taurus erhält von der Lösung die Anzahl der Aufgaben, die Parallelität, den Hochlauf und die Dauer der Warteschleife. Es übersetzt diese Werte in die eigenen Ladesteuerungen des zugrunde liegenden Frameworks. Taurus hat Vorrang vor der Last, die Ihr Skript deklariert. Es schreibt einen k6-Optionsblock, eine Locust `LoadTestShape` - oder eine JMeter-Thread-Gruppe um oder ignoriert ihn. Die virtuellen Benutzer, die eine Region generiert, sind die Anzahl der Aufgaben multipliziert mit der Parallelität für jede Aufgabe. Diese Form ist für jedes Framework gleich. So hat die Lösung jeden Test vor Version 4.3.0 ausgeführt.

Nativer Modus

Die Fargate-Aufgaben verwenden das spezielle Image für das Test-Framework, das Taurus nicht enthält. Anstatt eine Taurus-Konfiguration zu schreiben, ruft die Lösung das Framework direkt

auf:`jmeter -n -t`,, oder. `k6 run` `locust --headless` Es übergibt keine Ladeparameter. Ihr Skript ist die alleinige Autorität für den Traffic, den es generiert.

Aus diesem Design ergeben sich zwei Konsequenzen, und beide wirken sich darauf aus, wie Sie einen Test dimensionieren:

- Aufgaben multiplizieren die Last. Jede Aufgabe führt einen unabhängigen Framework-Prozess ohne Koordination zwischen den Aufgaben aus. Infolgedessen generiert eine Region für jede Aufgabe eine vollständige Kopie der in Ihrem Skript deklarierten Last. Beispiel: Ein k6-Skript mit 200 virtuellen Benutzern, das auf fünf Aufgaben ausgeführt wird, bringt 1.000 virtuelle Benutzer auf das Ziel. Die Anzahl der Aufgaben ist die einzige Laststeuerung, die die Lösung in diesem Modus bietet. Es verschiebt ganze Vielfache dessen, was das Skript deklariert.
- Eine Sicherheitsdauer begrenzt den Lauf. Da das Skript entscheidet, wann der Test endet, benötigt die Lösung eine Sicherheitsdauer von bis zu 24 Stunden. Wenn der Test nach Ablauf der Dauer noch läuft, stoppt die Lösung das Framework. Es sammelt die Ergebnisse für den Teil, der ausgeführt wurde, und zeichnet den Testlauf als abgeschlossen und nicht als fehlgeschlagen auf.

Planung von Tests

Die Lösung bietet drei Ausführungszeitoptionen für die Ausführung von Lasttests:

- Jetzt ausführen — Führt den Belastungstest unmittelbar nach der Erstellung aus
- Einmal ausführen — Führen Sie den Test an einem bestimmten Datum und zu einer bestimmten Uhrzeit in der future aus
- Nach einem Zeitplan ausführen — Erstellen Sie wiederkehrende Tests mithilfe von Cron-Ausdrücken, um den Zeitplan zu definieren

Wenn Sie Einmal ausführen auswählen, geben Sie die Laufzeit im 24-Stunden-Format und das Ausführungsdatum an, an dem der Belastungstest ausgeführt werden soll.

Wenn Sie „Nach Zeitplan ausführen“ auswählen, können Sie entweder manuell einen Cron-Ausdruck eingeben oder aus gängigen Cron-Mustern wählen (z. B. stündlich, täglich zu einer bestimmten Zeit, wochentags oder monatlich). Der Cron-Ausdruck verwendet ein detailliertes Zeitplanformat mit Feldern für Minuten, Stunden, Monatstag, Monat, Wochentag und Jahr. Sie müssen auch ein Ablaufdatum angeben, das festlegt, wann der geplante Test nicht mehr ausgeführt werden soll. Weitere Informationen zu den Regeln für die Terminierung von Validierungsregeln finden Sie im Abschnitt Einschränkungen bei der Terminplanung in diesem Handbuch.

Note

- Testdauer: Berücksichtigen Sie bei der Planung die Gesamtdauer der Tests. Ein Test mit einer Anlaufzeit von 10 Minuten und einer Haltezeit von 40 Minuten dauert beispielsweise etwa 80 Minuten.
- Mindestintervall: Stellen Sie sicher, dass das Intervall zwischen den geplanten Tests länger ist als die geschätzte Testdauer. Wenn der Test beispielsweise etwa 80 Minuten dauert, sollten Sie ihn so planen, dass er nicht häufiger als alle 3 Stunden ausgeführt wird.
- Stündliche Begrenzung: Das System erlaubt es nicht, Tests mit einem Unterschied von nur einer Stunde zu planen, selbst wenn die geschätzte Testdauer weniger als eine Stunde beträgt.

Gleichzeitige Tests

Jedes Mal, wenn ein Auslastungstest ausgeführt wird, erstellt die Task-Runner-AWS-Lambda-Funktion ein CloudWatch Amazon-Dashboard, das `EcsLoadTesting-<testId>-<region>` in jeder Region benannt ist, in der der Test ausgeführt wird. Das CloudWatch Dashboard zeigt die kombinierte Ausgabe aller Aufgaben, die im Amazon ECS-Cluster ausgeführt werden, in Echtzeit an: durchschnittliche Antwortzeit, Anzahl gleichzeitiger Benutzer, Anzahl erfolgreicher Anfragen und Anzahl fehlgeschlagener Anfragen. Die Lösung aggregiert jede Metrik sekundengenau und aktualisiert das Dashboard jede Minute.

Bei nachfolgenden Durchläufen desselben Testszenarios wird dasselbe Dashboard aktualisiert, sodass Ihr Konto ein Dashboard für jedes Testszenario in jeder Region enthält. Diese Dashboards verbleiben nach Abschluss der Tests in Ihrem Konto. Für sie fällt eine monatliche Gebühr an, bis Sie sie löschen. Die Lösung löscht die Dashboards eines Szenarios, wenn Sie das Testszenario löschen (z. B. über die Webkonsole). Die Dashboards werden nicht gelöscht, wenn Sie die Stacks der Lösung löschen. CloudFormation Weitere Informationen finden Sie in den Abschnitten Kosten und Manuelles Löschen einbehaltener Ressourcen in diesem Handbuch.

Benutzerverwaltung

Bei der Erstkonfiguration geben Sie einen Benutzernamen und eine E-Mail-Adresse an, die Amazon Cognito verwendet, um Ihnen Zugriff auf die Webkonsole der Lösung zu gewähren. Die Konsole bietet keine Benutzerverwaltung. Um weitere Benutzer hinzuzufügen, müssen Sie die Amazon

Cognito Cognito-Konsole verwenden. Weitere Informationen finden Sie unter Managing Users in User Pools im Amazon Cognito Developer Guide.

Informationen zur Migration vorhandener Benutzer zu Amazon Cognito-Benutzerpools finden Sie im AWS-Blog Approaches for migration users to Amazon Cognito user pools.

Identitätsanbieterverbund

Der Amazon Cognito Cognito-Benutzerpool der Lösung unterstützt den Verbund mit externen Identitätsanbietern (IdPs), die die Protokolle SAML 2.0 oder OpenID Connect (OIDC) verwenden. Federation ermöglicht es Benutzern, sich mit ihren vorhandenen Unternehmens- oder Organisationsanmeldedaten anstelle von Anmeldeinformationen an der Webkonsole anzumelden. Cognito-native Verbundbenutzer erhalten dieselben Zugriffsberechtigungen wie Benutzer, die direkt im Cognito-Benutzerpool erstellt wurden.

Die Lösung stellt bereits den Cognito-Benutzerpool, die Domäne, den App-Client und die gehostete Benutzeroberfläche bereit. Um den Verbund zu aktivieren, müssen Sie nur Ihren Identitätsanbieter registrieren und ihn auf dem vorhandenen App-Client aktivieren.

Wenn Sie die optionale MCP-Serverintegration bereitstellen, können Verbundbenutzer auch mit denselben Cognito-Benutzerpool-Anmeldeinformationen auf den MCP-Server zugreifen.

Voraussetzungen

Bevor Sie den Verbund konfigurieren, benötigen Sie Folgendes:

- Ein externer Identitätsanbieter, der SAML 2.0 oder OIDC unterstützt
- Administratorzugriff zur Konfiguration des externen IdP (um Weiterleitungs-URIs oder ACS-URLs festzulegen)
- Die Cognito-Benutzerpool-ID der Lösung (verfügbar in den CloudFormation Stack-Ressourcen oder der Amazon Cognito Cognito-Konsole)
- Das Cognito-Domainpräfix der Lösung (verfügbar in den CloudFormation Stack-Ausgaben oder in der Cognito-Konsole unter App-Integration > Domain)

Schritt 1: Konfigurieren Sie Ihren Identitätsanbieter

Konfigurieren Sie Ihren externen Identitätsanbieter mit den folgenden Werten, sodass er mit dem Cognito-Benutzerpool der Lösung kommunizieren kann.

Für SAML-Identitätsanbieter:

- SP-Entitäts-ID: `urn:amazon:cognito:sp:_<UserPoolId>_`
- ACS-URL: `\https://<cognito-domain>.auth.<region>.amazoncognito.com/saml2/idpresponse`

Für OIDC-Identitätsanbieter:

- URI umleiten: `\https://<cognito-domain>.auth.<region>.amazoncognito.com/oauth2/idpresponse`

Einzelheiten zu den Anforderungen Ihres IdP finden Sie unter Hinzufügen von SAML-Identitätsanbietern zu einem Benutzerpool oder Hinzufügen von OIDC-Identitätsanbietern zu einem Benutzerpool im Amazon Cognito Developer Guide.

Schritt 2: Registrieren Sie den Identitätsanbieter in Cognito

Fügen Sie mithilfe der Amazon Cognito-Konsole Ihren externen Identitätsanbieter zum bestehenden Cognito-Benutzerpool der Lösung hinzu.

Eine schrittweise Anleitung finden Sie unter Hinzufügen einer Benutzerpool-Anmeldung über einen Drittanbieter im Amazon Cognito Developer Guide.

Schritt 3: Attributzuordnungen konfigurieren

Konfigurieren Sie Attributzuordnungen zwischen den Ansprüchen Ihres Identitätsanbieters und den Attributen des Cognito-Benutzerpools. Ordnen Sie mindestens den E-Mail-Anspruch des Benutzers vom externen Anbieter dem `email` Cognito-Attribut zu. Erwägen Sie auch die Zuordnung `name` oder `nickname` ob Ihr Identitätsanbieter sie bereitstellt.

Anweisungen finden Sie unter Spezifizieren von Identitätsanbieter-Attributzuordnungen für Ihren Benutzerpool im Amazon Cognito Developer Guide.

Schritt 4: Aktivieren Sie den Identitätsanbieter auf dem App-Client

Suchen Sie in der Amazon Cognito Cognito-Konsole den App-Client, der von der Lösung erstellt wurde, und aktivieren Sie Ihren neuen Identitätsanbieter unter den Einstellungen für die gehostete Benutzeroberfläche.

Anweisungen finden Sie unter Konfiguration eines Benutzerpool-App-Clients im Amazon Cognito Developer Guide.

> **Note**
>
> Die Lösung konfiguriert bereits die Callback- und Abmelde-URLs, die OAuth-Bereiche und die gehostete UI-Domain des App-Clients. Sie müssen diese Einstellungen nicht ändern — aktivieren Sie lediglich Ihren Identitätsanbieter auf dem vorhandenen App-Client.

> **Important**
>
> Die Lösung lässt die `SupportedIdentityProviders` Eigenschaft absichtlich in der CloudFormation App-Client-Konfiguration aus. Auf diese Weise können Sie nach der Bereitstellung Identitätsanbieter hinzufügen, ohne dass eine Drift-Erkennung ausgelöst wird CloudFormation . Wenn diese Eigenschaft in der Vorlage festgelegt wäre, würden alle manuellen IdP-Änderungen über die Konsole oder CLI beim nächsten Stack-Update überschrieben, sodass der App-Client nur auf die in der Vorlage aufgeführten Anbieter zurückgesetzt würde.
> Da diese Eigenschaft weggelassen wird, wird CloudFormation nicht nachverfolgt oder verwaltet, welche Identitätsanbieter auf dem App-Client aktiviert sind. Nachdem Sie den Verbund konfiguriert haben, sind Sie für die Verwaltung der Inhalte von `SupportedIdentityProviders` auf dem App-Client verantwortlich. Um unbefugte Änderungen zu überwachen, aktivieren Sie die CloudTrailAWS-Protokollierung und erstellen Sie EventBridge Amazon-Regeln für Warnmeldungen `CreateIdentityProvider` und `UpdateUserPoolClient` API-Aufrufe, die auf den Cognito-Benutzerpool der Lösung abzielen.

> **Note**
>
> - Durch das Hinzufügen eines externen Identitätsanbieters wird bestehenden Cognito-native Benutzern nicht die Möglichkeit genommen, sich mit ihren aktuellen Anmeldeinformationen anzumelden.
> - Verbundbenutzer unterliegen denselben regionalen Verfügbarkeitsbeschränkungen wie der Cognito-Benutzerpool. Weitere Informationen finden Sie unter Regionaler Einsatz.
> - Testen Sie die Verbundanmeldung mit einer kleinen Benutzergruppe, bevor Sie sie in Ihrer Organisation einführen.

Den Standard-Cognito-Benutzer deaktivieren oder löschen

Nach der Konfiguration des Verbunds möchten Sie möglicherweise den Standardbenutzer, der während der Stack-Bereitstellung erstellt wurde, deaktivieren oder löschen. Dies ist optional — der Standardbenutzer funktioniert weiterhin zusammen mit der Verbundanmeldung.

Um einen Benutzer zu deaktivieren, navigieren Sie in der Amazon Cognito-Konsole zum Cognito-Benutzerpool der Lösung, wählen Sie die Registerkarte Benutzer aus, wählen Sie den Benutzer aus und wählen Sie Benutzerzugriff deaktivieren aus. Um einen Benutzer zu löschen, müssen Sie ihn zuerst deaktivieren und dann Benutzer löschen wählen. Wenn Sie einen Benutzer deaktivieren, werden dessen Token gesperrt und die Anmeldung verhindert, während das Konto erhalten bleibt. Durch das Löschen wird es dauerhaft entfernt.

Weitere Informationen finden Sie unter Benutzerkonten verwalten und suchen im Amazon Cognito Developer Guide.

Regionale Bereitstellung

Diese Lösung verwendet Amazon Cognito, das nur in bestimmten AWS-Regionen verfügbar ist. Daher müssen Sie diese Lösung in einer Region bereitstellen, in der Amazon Cognito verfügbar ist. Die aktuelle Serviceverfügbarkeit nach Regionen finden Sie in der regionalen AWS-Serviceliste.

Planen Sie Ihren Einsatz

In diesem Abschnitt werden Kosten, Sicherheit, unterstützte Regionen, Kontingente und andere Überlegungen beschrieben, die Sie vor der Bereitstellung der Lösung überprüfen sollten.

Cost (Kosten)

Sie sind für die Kosten der AWS-Services verantwortlich, die beim Betrieb dieser Lösung verwendet werden. Die Gesamtkosten hängen von der Anzahl der ausgeführten Lasttests, der Dauer dieser Tests und der Menge der generierten Daten ab. Zum jetzigen Zeitpunkt belaufen sich die geschätzten Kosten für die Ausführung dieser Lösung mit Standardeinstellungen in der Region USA Ost (Nord-Virginia) auf etwa 30,90$ pro Monat.

Die folgende Tabelle enthält ein Beispiel für eine Aufschlüsselung der Kosten für die Bereitstellung dieser Lösung mit den Standardparametern in der Region USA Ost (Nord-Virginia) für einen Monat.

AWS Service	Dimensionen	Kosten [USD]
AWS Fargate	10 On-Demand-Aufgaben (mit zwei vCPUs und 4 GB Arbeitsspeicher), die 30 Stunden lang ausgeführt werden	29,62$
Amazon DynamoDB	1.000 On-Demand-Schreibkapazitätseinheiten 1.000 On-Demand-Lesekapazitätseinheiten	0,0015$
AWS Lambda	1.000 Anfragen Gesamtdauer 10 Minuten	1,25$
AWS Step Functions	1.000 Zustandsübergänge	0,025 USD
Insgesamt:		30,90$ pro Monat

Die Lösungsressourcen sind mit dem Schlüssel= SolutionId und dem Wert=SO0062 gekennzeichnet. Sie können den Tag-Schlüssel aktivieren, indem Sie der Dokumentation activating-tags folgen SolutionId . Sobald das Tag aktiviert ist, können Sie eine Kostenkategorienregel erstellen, indem Sie der Dokumentation zur Erstellung von Kostenkategorien folgen. Sie können die für die Lösung angefallenen Kosten einsehen, indem Sie die Kostenkategorienkonsole überwachen und den Namen der Kostenkategorie auswählen.

Wir empfehlen, über den AWS Cost Explorer ein Budget zu erstellen, um die Kosten besser verwalten zu können. Die Preise sind freibleibend. Vollständige Informationen finden Sie auf der Preisseite für jeden AWS-Service, der in dieser Lösung verwendet wird.

> **Note**
>
> Die standardmäßige Taskkonfiguration verwendet 2 vCPUs und 4 GB Arbeitsspeicher pro Aufgabe. Wenn Ihre Auslastungstests diese Ressourcen nicht benötigen, können Sie sie reduzieren, um die Kosten zu senken. Umgekehrt können Sie die Ressourcen erhöhen, um eine höhere Parallelität pro Aufgabe zu unterstützen. Weitere Informationen finden Sie im Abschnitt Erhöhen der Anzahl der Container-Ressourcen in diesem Handbuch.

> **Note**
>
> Diese Lösung bietet die Möglichkeit, Live-Daten bei der Ausführung eines Tests einzubeziehen. Für diese Funktion sind eine zusätzliche AWS Lambda Lambda-Funktion und ein AWS IoT Core Core-Thema erforderlich, für die zusätzliche Kosten anfallen.

> **Note**
>
> Die Kosten von Fargate hängen davon ab, wie lange die Aufgaben ausgeführt werden. Im Standardmodus wird dies von den von Ihnen festgelegten Hochlauf- und Hold-For-Werten bestimmt. Im systemeigenen Modus bestimmt Ihr Skript dies, begrenzt durch die von Ihnen festgelegte Sicherheitsdauer. Diese Dauer begrenzt auch, wie lange ein systemeigener Test abgerechnet werden kann. Weitere Informationen finden Sie unter Verkehrsshape-Modi.

Kosten für die Integration von DevOps AWS-Agenten

Diese Lösung stellt eine Infrastruktur zur Unterstützung der Integration mit AWS DevOps Agent bereit, darunter zwei Amazon DynamoDB-Tabellen für die Registrierung von Agentenbereichen und die Nachverfolgung von Ermittlungen. Diese Tabellen verwenden den On-Demand-Kapazitätsmodus, sodass AWS Ihnen je nach Nutzung eine Gebühr pro Lese- und Schreibanfrage berechnet. Weitere Informationen zu Preisen finden Sie unter Amazon DynamoDB – Preise.

Wenn Sie die DevOps Agent-Integration verwenden, um Untersuchungen Ihrer Testergebnisse durchzuführen, wird der DevOps AWS-Agent-Service separat pro Agenten-Sekunde in Rechnung gestellt. Diese Kosten sind in der obigen Schätzung nicht enthalten. Vollständige Preisinformationen finden Sie unter Preise für DevOps AWS-Agenten.

CloudWatch Dashboard-Kosten pro Testszenario

Jedes Mal, wenn ein Auslastungstest ausgeführt wird, erstellt diese Lösung ein CloudWatch Amazon-Dashboard, das `EcsLoadTesting-<testId>-<region>` in jeder Region benannt ist, in der der Test ausgeführt wird. Bei nachfolgenden Durchläufen desselben Testszenarios wird dasselbe Dashboard aktualisiert, sodass Ihr Konto ein Dashboard für jedes Testszenario in jeder Region enthält. Diese Dashboards verbleiben nach Abschluss der Tests in Ihrem Konto. Für sie fällt eine monatliche Gebühr an, bis Sie sie löschen. Diese Kosten sind in der obigen Schätzung nicht enthalten.

Servicekomponente	Dimensionen	Kosten [USD]
Amazon CloudWatch	1 Dashboard pro Testszenario pro Region	3,00$ pro Dashboard pro Monat
Beispiel: 10 Testszenarien, die jeweils in 2 Regionen laufen	20 Dashboards	60,00$ pro Monat (vor dem kostenlosen Kontingent)

Das CloudWatch kostenlose Kontingent umfasst drei Dashboards (jeweils bis zu 50 Messwerte) pro Monat kostenlos. Bei Konten mit vielen Testszenarien oder bei Tests, die über mehrere Regionen verteilt sind, können sich diese Kosten schnell erhöhen. Um laufende Gebühren zu vermeiden, löschen Sie Testszenarien, die Sie nicht mehr benötigen, über die Webkonsole der Lösung, die auch die zugehörigen Dashboards löscht, oder löschen Sie die Dashboards manuell. Anweisungen zum

manuellen Löschen finden Sie im Abschnitt Manuelles Löschen von zurückgehaltenen Ressourcen in diesem Handbuch.

Die Preise sind freibleibend. Vollständige Informationen finden Sie in den CloudWatch Amazon-Preisen.

Zusätzliche Kosten für MCP-Server (optional)

Die folgende Tabelle enthält eine Aufschlüsselung der Kosten für die MCP-Serverintegration mit Preisen in der Region USA Ost (Nord-Virginia) für einen Monat.

Servicekomponente	Dimensionen	Kosten [USD]
AgentCore Gateway — Indizierung von Tools	10 Werkzeuge × 0,02$ pro 100 Werkzeuge	0,002$
AgentCore Gateway — Such-API	10.000 Interaktionen × 0,025$ pro 1.000	0,25$
AgentCore Gateway — API-Aufrufe	50.000 Aufrufe × 0,005 USD pro 1.000	0,25$
AWS Lambda-Funktion	Variabel je nach Nutzung (typische Workloads)	5,00$ — 20,00$
Geschätzte zusätzliche Kosten insgesamt:		5,50$ bis 20,50$ pro Monat

Die Preise sind freibleibend. Vollständige Informationen zu den AgentCore Gateway-Preisen finden Sie unter Amazon Bedrock Pricing (Abschnitt AgentCore Gateway). Die Lambda-Preise finden Sie unter AWS Lambda Pricing.

Zusätzliche Kosten für die von ALB + ECS Fargate gehostete Webkonsole (optional)

Wenn Sie sich für die Bereitstellungsoption ALB + ECS Fargate entscheiden, fallen die folgenden zusätzlichen Kosten an. Die Preise gelten für die Region USA Ost (Nord-Virginia).

Servicekomponente	Dimensionen	Kosten [USD]
Elastic Load Balancing	1 Application Load Balancer	19,35$
AWS Fargate	2 Aufgaben (0,25 vCPU, 0,5 GB Speicher) werden ausgeführt 24/7	17,77$
Amazon Virtual Private Cloud (VPC)	3 Schnittstellen-Endpunkte, 2 Availability Zones	43,21$
Amazon CloudWatch	10 Metriken, 1.000 GetMetricData Anfragen, 1.000 API-Anfragen, 5 GB Protokollaufnahme, 1 Dashboard, 10 Alarme	6,54$
Amazon S3	0,1 GB S3-Standardspeicher, 100 PUT/COPY/ POST/LIST Anfragen, 10.000 Anfragen GET/ SELECT	0,01$
AWS WAF	1 Web-ACL, 3 verwaltete Regelgruppen (CRS,, AnonymousIpList) AmazonIpReputationList, 10.000 Anfragen (die Anforderungskosten sind vernachlässigbar)	8,00$
Geschätzte zusätzlic he Kosten insgesamt:		94,88$ pro Monat

Die Preise sind freibleibend. Vollständige Informationen finden Sie unter Elastic Load Balancing Balancing-Preise, AWS Fargate-Preise und AWS WAF WAF-Preise.

Sicherheit

Wenn Sie Systeme auf der AWS-Infrastruktur aufbauen, werden die Sicherheitsaufgaben zwischen Ihnen und AWS aufgeteilt. Dieses Modell der geteilten Verantwortung reduziert Ihren betrieblichen Aufwand, da AWS die Komponenten wie das Host-Betriebssystem, die Virtualisierungsebene und die physische Sicherheit der Einrichtungen, in denen die Services betrieben werden, betreibt, verwaltet und kontrolliert. Weitere Informationen zur AWS-Sicherheit finden Sie unter AWS Cloud Security.

IAM-Rollen

AWS Identity and Access Management (IAM) -Rollen ermöglichen es Kunden, Services und Benutzern in der AWS-Cloud detaillierte Zugriffsrichtlinien und -berechtigungen zuzuweisen. Diese Lösung erstellt IAM-Rollen, die den AWS Lambda Lambda-Funktionen der Lösung Zugriff gewähren, um regionale Ressourcen zu erstellen.

Amazon CloudFront

Diese Lösung stellt eine Web-Benutzeroberfläche bereit, die in einem Amazon S3 S3-Bucket gehostet wird, der von Amazon CloudFront vertrieben wird. Um die Latenz zu reduzieren und die Sicherheit zu verbessern, umfasst diese Lösung eine CloudFront Distribution mit einer Ursprungszugriffsidentität. Dabei handelt es sich um einen CloudFront Benutzer, der öffentlichen Zugriff auf die Bucket-Inhalte der Lösungswebsite gewährt. Standardmäßig unterstützt die CloudFront Distribution Verbindungen bis zu TLS 1.3, wobei die höchste Protokollversion verwendet wird, die der Viewer unterstützt. Weitere Informationen finden Sie unter Beschränken des Zugriffs auf einen Amazon S3 S3-Ursprung im Amazon CloudFront Developer Guide.

CloudFront aktiviert zusätzliche Sicherheitsmaßnahmen, um HTTP-Sicherheitsheader an jede Zuschauerantwort anzuhängen. Weitere Informationen finden Sie unter Hinzufügen oder Entfernen von HTTP-Headern in Antworten. CloudFront

Diese Lösung verwendet das CloudFront Standardzertifikat, für das mindestens das Sicherheitsprotokoll TLS v1.0 unterstützt wird. Um die Verwendung von TLS v1.2 oder TLS v1.3 zu erzwingen, müssen Sie ein benutzerdefiniertes SSL-Zertifikat anstelle des Standardzertifikats verwenden. CloudFront Weitere Informationen finden Sie unter Wie konfiguriere ich meine CloudFront Distribution für die Verwendung eines SSL/TLS Zertifikats?

Amazon API Gateway

Diese Lösung stellt Edge-optimierte Amazon API Gateway Gateway-Endpunkte bereit, um RESTful-APIs für die Lasttestfunktion bereitzustellen. Dabei wird der standardmäßige API-Gateway-Endpunkt anstelle einer benutzerdefinierten Domain verwendet. Für Edge-optimierte APIs, die den Standardendpunkt verwenden, verwendet API Gateway die TLS-1-0 Sicherheitsrichtlinie. Weitere Informationen finden Sie unter Arbeiten mit REST-APIs im Amazon API Gateway Developer Guide.

Diese Lösung verwendet das Standard-API-Gateway-Zertifikat, für das mindestens das Sicherheitsprotokoll TLS v1.0 unterstützt wird. Um die Verwendung von TLS v1.2 oder TLS

v1.3 zu erzwingen, müssen Sie anstelle des standardmäßigen API-Gateway-Zertifikats eine benutzerdefinierte Domain mit einem benutzerdefinierten SSL-Zertifikat verwenden. Weitere Informationen finden Sie unter Benutzerdefinierte Domainnamen für REST-APIs einrichten.

AWS Fargate-Sicherheitsgruppe

Standardmäßig öffnet diese Lösung die ausgehende Regel der AWS Fargate-Sicherheitsgruppe für die Öffentlichkeit. Wenn Sie verhindern möchten, dass AWS Fargate überall Datenverkehr sendet, ändern Sie die ausgehende Regel in eine bestimmte Classless Inter-Domain Routing (CIDR) -Regel.

Diese Sicherheitsgruppe umfasst auch eine Regel für eingehenden Datenverkehr, die lokalen Datenverkehr auf Port 50.000 zu jeder Quelle zulässt, die zu derselben Sicherheitsgruppe gehört. Dies wird verwendet, damit die Container miteinander kommunizieren können.

Amazon VPC

VPC: Eine virtuelle private Cloud (VPC), die auf dem Amazon VPC-Service basiert, bietet Ihnen ein privates, logisch isoliertes Netzwerk in der AWS-Cloud.

Sie können während der Bereitstellung Ihre eigene VPC in CloudFormation AWS-Parametern angeben. Die VPC wird ausschließlich von den ECS-Aufgaben verwendet, die Last erzeugen. Die Webkonsole und die API werden nicht in dieser VPC bereitgestellt. Wenn Sie keine vorhandene VPC angeben, erstellt die Lösung eine neue VPC mit der erforderlichen Netzwerkkonfiguration. Wenn Sie sich für die Verwendung einer vorhandenen VPC entscheiden, muss diese die folgenden Anforderungen erfüllen, um Lasttestaufgaben erfolgreich ausführen zu können.

VPC-Anforderungen

Die Mindestanforderungen für eine VPC zur Verwendung mit Distributed Load Testing auf AWS sind unten aufgeführt.

- Die VPC muss mindestens zwei AZs enthalten
- Die VPC muss mindestens zwei Subnetze enthalten, jedes in einer separaten AZ
- VPC-Subnetze können entweder öffentlich oder privat sein, aber sie müssen dieselbe Konfiguration verwenden (beide öffentlich ODER beide privat)
- Die VPC muss Zugriff auf Endpunkte für ECR, CloudWatch Logs, S3 und AWS IoT Core bieten.
- Die VPC muss Zugriff auf die Dienste gewähren, auf die Lasttests abzielen.

> **Note**
>
> Wenn Sie keine VPC haben, die diese Kriterien erfüllt, können Sie mit dem VPC-Assistenten schnell eine VPC erstellen. Weitere Informationen finden Sie unter Erstellen einer VPC.

Öffentliche Subnetze können diese Anforderungen erfüllen, indem sie Folgendes beinhalten:

- Ein an die VPC angeschlossenes Internet-Gateway
- Eine Route zum Internet-Gateway (0.0.0. 0/0)

Private Subnetze können diese Anforderungen durch die Verwendung von NAT-Gateways oder VPC-Endpunkten erfüllen, wie unten beschrieben.

Option 1: NAT-Gateway

- Stellen Sie in jeder AZ mit privaten Subnetzen ein NAT-Gateway bereit
- Konfigurieren Sie Routentabellen für die Weiterleitung von internetgebundenem Datenverkehr (0.0.0. 0/0) über das NAT-Gateway

Option 2: VPC-Endpunkte

Erstellen Sie die folgenden VPC-Endpoints in Ihrer VPC:

- Amazon ECR API-Endpunkt: `com.amazonaws.<region>.ecr.api`
- Amazon ECR DKR-Endpunkt: `com.amazonaws.<region>.ecr.dkr`
- Amazon CloudWatch Logs-Endpunkt: `com.amazonaws.<region>.logs`
- Amazon S3 Gateway-Endpunkt: `com.amazonaws.<region>.s3`
- AWS IoT Core Core-Endpunkt (erforderlich, wenn Live-Datendiagramme verwendet werden) `com.amazonaws.<region>.iot.data`

Andere VPC-Konfigurationen funktionieren möglicherweise ebenfalls.

> **Important**
>
> Die an jede VPC-Endpunktschnittstelle angefügte Sicherheitsgruppe muss eingehenden TCP-Verkehr auf Port 443 von der ECS-Aufgabensicherheitsgruppe zulassen.

Konfiguration der Sicherheitsgruppe

Während der Bereitstellung erstellt die Lösung eine Sicherheitsgruppe innerhalb Ihrer VPC, um den folgenden Datenverkehr mit Aufgaben im ECS-Cluster zuzulassen:

- Der gesamte ausgehende Verkehr
- Eingehender Datenverkehr auf Port 50000 von anderen Aufgaben in derselben Sicherheitsgruppe, um die Koordination zwischen den Aufgaben des Mitarbeiters und des Leiters zu erleichtern.

Netzwerk-Stresstest

Sie sind dafür verantwortlich, diese Lösung gemäß der Amazon EC2-Testrichtlinie zu verwenden. Die Richtlinie deckt Netzwerktests mit hohem Volumen ab, die von Amazon EC2 EC2-Instances zu anderen Amazon EC2 EC2-Instances, AWS-Services oder externen Endpunkten ausgeführt werden. Diese Tests werden manchmal als Stresstests, Belastungstests oder Gameday-Tests bezeichnet. Lesen Sie die Richtlinie, um den Unterschied zwischen Netzwerk-Stresstests und DDoS-Simulationen zu verstehen (die in EC2 verboten sind und separat durch die Richtlinie für DDoS-Simulationstests abgedeckt werden), und beachten Sie, dass AWS bei hohem Datenverkehrsaufkommen möglicherweise Traffic Engineering oder -Shaping einsetzt. Aktuelle Schwellenwerte und Anleitungen, bevor Sie umfangreiche Tests durchführen, finden Sie auf der Seite mit den Richtlinien.

Beschränken Sie den Zugriff auf die öffentliche Benutzerschnittstelle

Der Ansatz zur Beschränkung des Zugriffs auf die Webkonsole hängt von der ausgewählten Bereitstellungsoption ab.

Standardbereitstellung (CloudFront + S3) — Um den Zugriff auf die öffentlich zugängliche Benutzeroberfläche über die von IAM und Amazon Cognito bereitgestellten Authentifizierungs- und Autorisierungsmechanismen hinaus einzuschränken, können Sie der Distribution eine AWS WAF WAF-Web-ACL zuordnen. CloudFront Erwägen Sie die Nutzung der AWS WAF Security Automations-Lösung, die eine Reihe von vorkonfigurierten AWS-WAF-Regeln einsetzt, die gängige

webbasierte Angriffe filtern. Die Standardvorlage CloudFront + S3 stellt WAF-Ressourcen nicht automatisch bereit.

Bereitstellung von ALB + ECS Fargate — Die Lösung stellt automatisch eine AWS WAF WAF-Web-ACL vor der ALB mit verwalteten Regeln bereit, die grundlegenden Schutz vor gängigen webbasierten Angriffen bieten. Sie können die WAF-Regeln an Ihre spezifischen Sicherheitsanforderungen anpassen, einschließlich des Hinzufügens von IP-based Zulassungs- oder Sperrlisten, geografischen Einschränkungen, Ratenbegrenzungen oder zusätzlichen von AWS verwalteten Regelgruppen. Anweisungen zum Ändern der WAF-Konfiguration finden Sie im Abschnitt zur WAF-Integration in den Bereitstellungsanweisungen.

MCP-Serversicherheit (optional)

Wenn Sie die optionale MCP-Serverintegration einsetzen, verwendet die Lösung Amazon Bedrock AgentCore Gateway, um KI-Agenten einen sicheren Zugriff auf Lasttestdaten zu ermöglichen. AgentCore Gateway validiert Amazon Cognito Cognito-Authentifizierungstoken für jede Anfrage, sodass nur autorisierte Benutzer auf den MCP-Server zugreifen können. Für alle Interaktionen mit dem MCP-Server gelten dieselben Berechtigungsgrenzen und Zugriffskontrollen wie für die Webkonsole.

Der Parameter MCP-Serverzugriffsmodus steuert, was KI-Agenten tun können. Im `ReadOnly` Standardmodus stellt der MCP-Server nur Lesetools zur Verfügung. Die AWS Identity and Access Management (IAM) -Richtlinie für die MCP Server Lambda-Funktion erlaubt nur `GET` Anfragen an die API der Lösung, sodass Agenten Testkonfigurationen oder Ergebnisse nicht ändern können. Im `ReadWrite` Modus können Agenten zusätzlich Testszenarien erstellen, ändern, löschen und starten. Wählen Sie diese Option `ReadWrite` nur, wenn Ihr Anwendungsfall erfordert, dass Agenten Tests ändern.

Rotation von Anmeldeinformationen — Wenn Sie einen MCP-Client mithilfe einer service-basierten Authentifizierung (Machine-to-Machine) verbinden, erstellt die Lösung einen Amazon Cognito-App-Client mit einem langlebigen Client-Geheimnis. Sie sind dafür verantwortlich, dieses Geheimnis zu wechseln und es sicher zu speichern. Wir empfehlen, es mindestens alle 90 Tage zu wechseln. Informationen zu den Verfahren finden Sie unter Rotation des MCP Server-Client-Geheimnisses unter Die Lösung verwenden.

Sicherheit für gehostete Webkonsolen mit ALB + ECS Fargate (optional)

Wenn Sie sich für die Bereitstellungsoption ALB + ECS Fargate entscheiden, gelten die folgenden Sicherheitsüberlegungen:

- Kompatibilität mit VPC Block Public Access — Die Option ALB + ECS Fargate wurde für Umgebungen entwickelt, in denen VPC Block Public Access (BPA) -Richtlinien den Datenverkehr von öffentlichen Distributionen blockieren. CloudFront Der ALB kann als interner Load Balancer in Ihrer VPC eingesetzt werden, auf den nur über Ihr Unternehmensnetzwerk, VPN oder AWS PrivateLink zugegriffen werden kann, sodass keine Anforderungen an die Gefährdung durch das öffentliche Internet erfüllt werden.
- ACM-Zertifikatsverwaltung — Die ALB verwendet ein ACM-Zertifikat für die HTTPS-Terminierung. Sie sind dafür verantwortlich, dass das Zertifikat gültig bleibt und vor Ablauf erneuert wird. ACM erneuert automatisch die von ihm verwalteten Zertifikate, importierte Zertifikate müssen jedoch manuell erneuert werden. Weitere Informationen finden Sie unter Verwaltete Zertifikatserneuerung im AWS Certificate Manager Manager-Benutzerhandbuch.
- AWS-WAF-Schutz — WAF wird standardmäßig mit der ALB + ECS Fargate-Vorlage bereitgestellt. Einzelheiten finden Sie unter Beschränken des Zugriffs auf die öffentliche Benutzeroberfläche.

Kopflose Sicherheit (bringen Sie Ihren eigenen Webserver mit) (optional)

Wenn Sie sich für die Headless-Bereitstellungsoption entscheiden und die Webkonsole auf Ihrem eigenen Webserver hosten, sind Sie für die folgenden Sicherheitsaspekte verantwortlich:

- HTTPS-Konfiguration — Wir empfehlen dringend, HTTPS auf Ihrem Webserver zu konfigurieren.
- Zugriffskontrollen — Sie sind für die Implementierung von Zugriffskontrollen, Firewallregeln und Netzwerksicherheit auf Ihrem Webserver verantwortlich.
- Sicherheitshärtung — Wenden Sie die Sicherheitsstandards Ihres Unternehmens auf den Webserver an, einschließlich Patching, Überwachung und Erkennung von Eindringlingen.

Third-party Test-Frameworks

Distributed Load Testing auf AWS bündelt drei Test-Frameworks von Drittanbietern — Apache JMeter, Grafana k6 und Locust. Im Rahmen des AWS-Modells der gemeinsamen Verantwortung sind Sie dafür verantwortlich, vor der Durchführung von Lasttests zu bewerten, ob diese Frameworks und ihre gebündelten Versionen die Sicherheitsanforderungen Ihres Unternehmens erfüllen. Die Lösung verteilt jedes Framework unverändert und verifiziert die gebündelten Binärdateien mithilfe von SHA512-Prüfsummen während der Erstellung und Laufzeit.

Einzelheiten dazu, wann jedes Framework installiert ist und wie es bereitgestellt wird, finden Sie unter Testen der Framework-Bereitstellung.

Apache JMeter

Die gebündelte Version von Apache JMeter weist bekannte Sicherheitslücken auf, die extern nicht vollständig gepatcht werden können, ohne die Kompatibilität mit dem Taurus-Testautomatisierungs-Framework und dem JMeter-Plugin-Ökosystem, von dem die Lösung abhängt, zu beeinträchtigen. Bevor Sie Lasttests durchführen, lesen Sie die Sicherheitsempfehlungen von Apache JMeter und prüfen Sie, ob sie Sicherheitslücken für Sie verursachen können.

> **Note**
>
> Apache JMeter läuft auch unter der Haube für den Testtyp Simple HTTP Endpoint. Wenn Sie eine URL, Methode, Header und Body-Payload in der Webkonsole konfigurieren, generiert die Lösung einen JMeter-Testplan und führt ihn mit der mitgelieferten JMeter-Binärdatei aus. Die in diesem Abschnitt beschriebenen Sicherheitsüberlegungen für JMeter gelten daher auch für einfache HTTP-Endpoint-Tests.

Wenn Sie eine gepatchte Version von JMeter benötigen, haben Sie zwei Möglichkeiten. Beide Optionen erfordern ein Testarchiv und sind nur für den JMeter-Testtyp verfügbar:

- Stellen Sie eine gepatchte JMeter-Binärdatei bereit — Fügen Sie eine gepatchte JMeter-Binärdatei in Ihr Testarchiv ein. Die Lösung verwendet Ihre Binärdatei anstelle der gebündelten Version.
- Einzelne Plugin-JARs überschreiben — Verwenden Sie den Plugin-Override-Mechanismus, um bestimmte anfällige Plugin-JARs durch gepatchte Versionen zu ersetzen. Weitere Informationen finden Sie unter JMeter-Tests.

Der Testtyp Simple HTTP Endpoint akzeptiert kein Testarchiv und kann daher die mitgelieferte JMeter-Binärdatei oder -Plugins nicht überschreiben. Wenn Sie HTTP-Endpunkttests mit einem gepatchten JMeter ausführen müssen, verwenden Sie den JMeter-Testtyp und stellen Sie ein JMeter-Skript (.jmx) oder ein .zip-Archiv bereit, das Ihre gepatchten JMeter-Binärdateien oder Plugin-JARs enthält.

Grafana K6

k6 wird unter der Lizenz veröffentlicht. AGPL-3.0 Die Webkonsole zeigt eine Lizenzbestätigungsmeldung an, wenn Sie einen neuen k6-Test erstellen. Zum Zeitpunkt der Veröffentlichung dieser Lösung wurden in der gebündelten Version von k6 keine bekannten Sicherheitslücken festgestellt. Die Lösung überwacht k6 nicht kontinuierlich auf neue

Sicherheitslücken. Sie sind dafür verantwortlich, k6 während der gesamten Nutzung anhand Ihrer Sicherheitsanforderungen zu bewerten.

Heuschrecke

Zum Zeitpunkt der Veröffentlichung dieser Lösung wurden in der gebündelten Version von Locust keine bekannten Sicherheitslücken festgestellt. Die Lösung überwacht Locust nicht kontinuierlich auf neue Sicherheitslücken. Sie sind dafür verantwortlich, Locust während der gesamten Nutzung anhand Ihrer Sicherheitsanforderungen zu bewerten.

Unterstützte AWS Regionen

Distributed Load Testing auf AWS ist in den folgenden AWS-Regionen verfügbar:

> **Note**
>
> Die aktuelle Verfügbarkeit der in dieser Lösung verwendeten AWS-Services nach Regionen finden Sie in der Liste der regionalen AWS-Dienste.

Name der Region	Regionscode
USA Ost (Nord-Virginia)	us-east-1
USA Ost (Ohio)	us-east-2
USA West (Nordkalifornien)	us-west-1
USA West (Oregon)	us-west-2
Kanada (Zentral)	ca-central-1
Europa (Frankfurt)	eu-central-1
Europa (Irland)	eu-west-1
Europa (London)	eu-west-2
Europa (Paris)	eu-west-3

Name der Region	Regionscode
Europa (Stockholm)	eu-north-1
Asien-Pazifik (Mumbai)	ap-south-1
Asien-Pazifik (Tokio)	ap-northeast-1
Asien-Pazifik (Seoul)	ap-northeast-2
Asien-Pazifik (Singapur)	ap-southeast-1
Asien-Pazifik (Sydney)	ap-southeast-2
Südamerika (São Paulo)	sa-east-1
AWS GovCloud (US-East)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1

> **Note**
>
> In AWS-Regionen GovCloud (USA) ist die standardmäßige Hosting-Option CloudFront + S3-Webkonsole nicht verfügbar, da Amazon in der AWS-Partition GovCloud (USA) nicht verfügbar CloudFront ist. Verwenden Sie die ALB + ECS Fargate-Vorlage oder die Headless-Vorlage und folgen Sie den Bereitstellungsschritten unter In AWS-Regionen GovCloud (USA) bereitstellen.

Von MCP Server unterstützte AWS-Regionen (optional)

Wenn Sie die optionale MCP-Serverintegration bereitstellen möchten, müssen Sie die Lösung in einer AWS-Region bereitstellen, in der AgentCore Gateway verfügbar ist. Die MCP-Serverfunktion ist nur in den folgenden AWS-Regionen verfügbar:

Name der Region	Regionscode
USA Ost (Nord-Virginia)	us-east-1

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

Die aktuelle Verfügbarkeit von AgentCore Gateway nach Regionen finden Sie unter Amazon Bedrock AgentCore Endpoints and Quotas in der Allgemeinen AWS-Referenz.

> **Tip**
>
> Wenn Sie den MCP-Server verwenden möchten, aber Lasttest-Traffic aus einer AWS-Region senden müssen, in der AgentCore Gateway nicht unterstützt wird, können Sie den Haupt-Hub-Stack (mit aktiviertem MCP-Server) in einer unterstützten AgentCore Gateway-Region bereitstellen und dann den regionalen Stack (siehe Multi-Region Bereitstellung) in der AWS-Region bereitstellen, in der Sie Testdatenverkehr generieren möchten. Auf diese Weise

können Sie den MCP-Server für AI-assisted Analysen verwenden und gleichzeitig Lasttests in der gewünschten Region ausführen.

Kontingente

Service Quotas, auch als Limits bezeichnet, sind die maximale Anzahl von Serviceressourcen oder -vorgängen für Ihr AWS-Konto.

Kontingente für AWS-Services in dieser Lösung

Stellen Sie sicher, dass Sie über ein ausreichendes Kontingent für jeden der in dieser Lösung implementierten Services verfügen. Weitere Informationen finden Sie unter AWS-Servicekontingente.

Verwenden Sie die folgenden Links, um zur Seite für diesen Dienst zu gelangen. Um die Service-Kontingente für alle AWS-Services in der Dokumentation anzuzeigen, ohne zwischen den Seiten zu wechseln, sehen Sie sich stattdessen die Informationen auf der Seite Service-Endpunkte und Kontingente in der PDF-Datei an.

CloudFormation AWS-Kontingente

Ihr AWS-Konto verfügt über CloudFormation AWS-Kontingente, die Sie beachten sollten, wenn Sie den Stack in dieser Lösung starten. Wenn Sie diese Kontingente verstehen, können Sie Limitationsfehler vermeiden, die Sie daran hindern würden, diese Lösung erfolgreich einzusetzen. Weitere Informationen finden Sie unter CloudFormation AWS-Kontingente im CloudFormation AWS-Benutzerhandbuch.

Kontingente für Lasttests

Die maximale Anzahl von Aufgaben, die in Amazon ECS mit dem Starttyp AWS Fargate ausgeführt werden können, basiert auf der vCPU-Größe der Aufgaben. Die Standardaufgabengröße bei Distributed Load Testing auf AWS beträgt 2 vCPU. Die aktuellen Standardkontingente finden Sie unter Amazon ECS-Servicekontingente. Die Kontingente für aktuelle Konten können von den aufgeführten Kontingenten abweichen. Um die für ein Konto spezifischen Kontingente zu überprüfen, überprüfen Sie das Service-Kontingent für die Anzahl der On-Demand-vCPU-Ressourcen von Fargate in der AWS-Managementkonsole. Anweisungen, wie Sie eine Erhöhung beantragen können, finden Sie unter AWS-Servicekontingente im AWS General Reference Guide.

Die Container-Images von Amazon Linux 2023 beschränken die gleichzeitigen Verbindungen pro Aufgabe nicht, was jedoch nicht bedeutet, dass eine Aufgabe eine unbegrenzte Anzahl von Benutzern unterstützen kann. Informationen zur Bestimmung der Anzahl gleichzeitiger Benutzer, die die Container für einen Test generieren können, finden Sie im Abschnitt „Anzahl der Benutzer ermitteln“ dieses Handbuchs. Das Limit pro Aufgabe ist eine Eigenschaft des Containers und gilt daher in beiden Traffic-Shape-Modi. Im Standardmodus halten Sie sich daran, indem Sie den Wert für Parallelität festlegen. Im systemeigenen Modus bleiben Sie innerhalb dieses Bereichs, indem Sie die Anzahl der virtuellen Benutzer einschränken, die Ihr Skript deklariert.

> **Note**
>
> Das empfohlene Limit für gleichzeitige Benutzer liegt auf der Grundlage der Standardeinstellungen bei 200 Benutzern.

Gleichzeitige Tests

Jedes Mal, wenn ein Auslastungstest ausgeführt wird, erstellt die Task-Runner-AWS-Lambda-Funktion ein CloudWatch Amazon-Dashboard, das `EcsLoadTesting-<testId>-<region>` in jeder Region benannt ist, in der der Test ausgeführt wird. Das CloudWatch Dashboard zeigt die kombinierte Ausgabe aller Aufgaben, die im Amazon ECS-Cluster ausgeführt werden, in Echtzeit an: durchschnittliche Antwortzeit, Anzahl gleichzeitiger Benutzer, Anzahl erfolgreicher Anfragen und Anzahl fehlgeschlagener Anfragen. Die Lösung aggregiert jede Metrik sekundengenau und aktualisiert das Dashboard jede Minute.

Bei nachfolgenden Durchläufen desselben Testszenarios wird dasselbe Dashboard aktualisiert, sodass Ihr Konto ein Dashboard für jedes Testszenario in jeder Region enthält. Diese Dashboards verbleiben nach Abschluss der Tests in Ihrem Konto. Für sie fällt eine monatliche Gebühr an, bis Sie sie löschen. Einzelheiten finden Sie unter CloudWatch Dashboard-Kosten pro Testszenario. Die Lösung löscht die Dashboards eines Szenarios, wenn Sie das Testszenario löschen (z. B. über die Webkonsole). Die Dashboards werden nicht gelöscht, wenn Sie die Stacks der Lösung löschen. CloudFormation Anweisungen zum manuellen Löschen finden Sie im Abschnitt Manuelles Löschen einbehaltener Ressourcen in diesem Handbuch.

Amazon EC2-Testrichtlinie

Sie sind dafür verantwortlich, Belastungstests gemäß der Amazon EC2-Testrichtlinie durchzuführen. In der Richtlinie finden Sie aktuelle Hinweise dazu, was als Netzwerk-Stresstest gilt, welches

Datenverkehrsvolumen zulässig ist und wann Sie sich vor der Durchführung von Tests mit hohem Datenvolumen an AWS wenden sollten. Beachten Sie, dass AWS Traffic Engineering oder Shaping bei hohem Gesamtdatenverkehrsvolumen anwenden kann.

Amazon-Richtlinie für CloudFront Auslastungstests

Wenn Sie planen, einen CloudFront Endpunkt unter Last zu testen, lesen Sie die Richtlinien für Auslastungstests im Amazon CloudFront Developer Guide. Wir empfehlen außerdem, den Traffic auf mehrere Aufgaben und Regionen zu verteilen. Stellen Sie mindestens 30 Minuten Anlaufzeit für den Auslastungstest bereit. Für Auslastungstests, bei denen mehr als 500.000 Anfragen pro Sekunde gesendet werden oder mehr als 300 Gbit/s Daten benötigt werden, empfehlen wir, zunächst eine Vorabgenehmigung für das Senden des Datenverkehrs einzuholen. CloudFront kann nicht genehmigten Lasttestverkehr drosseln, der sich auf die Verfügbarkeit der Dienste auswirkt. CloudFront

Überwachung der Lösung nach der Bereitstellung

Nach der Bereitstellung der Lösung empfehlen wir, die Ressourcen der Lösung mithilfe von CloudWatch Amazon-Alarmen und -Metriken kontinuierlich zu überwachen.

CloudWatch Alarme einrichten

Sie können CloudWatch Alarme einrichten, um wichtige Kennzahlen zu überwachen und Benachrichtigungen zu erhalten, wenn Schwellenwerte überschritten werden. Erwägen Sie die Einrichtung von Alarmen für die folgenden Ressourcen:

CloudFront Amazon-Vertriebsmetriken

Überwachen Sie die CloudFront Vertriebsleistung und die Fehler. Weitere Informationen finden Sie unter CloudFront Vertriebsmetriken im Amazon CloudFront Developer Guide.

Metriken für Amazon API Gateway

Überwachen Sie API-Anforderungsraten, Latenz und Fehler. Weitere Informationen finden Sie unter Amazon API Gateway-Dimensionen und -Metriken im Amazon API Gateway Developer Guide.

AWS Lambda Lambda-Funktionsmetriken

Überwachen Sie Lambda-Funktionsaufrufe, Dauer, Fehler und Drosselungen für die Microservices der Lösung.

Amazon ECS- und AWS Fargate-Metriken

Überwachen Sie die CPU- und Speicherauslastung von Aufgaben während der Lasttests, um sicherzustellen, dass die Ressourcen ausreichend sind.

Amazon DynamoDB-Metriken

Überwachen Sie den Lese- und Schreibkapazitätsverbrauch, gedrosselte Anfragen und die Latenz.

Beauftragen Sie einen Experten

Kurzfristige AWS Countdown Premium-Engagements für verteilte Lasttests auf AWS

Unsere AWS-Techniker bieten fachkundige Beratung zu den Grundlagen von Leistungstests, zur Skriptentwicklung und zur Ergebnisanalyse. Melden Sie sich jetzt an.

Übersicht

AWS Countdown Premium (CDP) Short Term Engagements bieten Expertenberatung für Unternehmen, die Leistungstests in großem Umfang durchführen. Durch ein kollaboratives „Do-it-yourself“ -Modell bieten AWS-Techniker strategische Übersicht und technisches Fachwissen, während Ihr Team die Verantwortung für die Ausführung behält. Erfahrene AWS-Techniker stehen innerhalb einer Woche nach der Registrierung zur Verfügung, ohne dass langfristige Verträge erforderlich sind.

Servicemodell

CDP-Techniker arbeiten mit Ihrem Team zusammen, um Sie während der gesamten Implementierung Ihrer Leistungstests zu beraten und zu beaufsichtigen. Dieser praxisorientierte Ansatz stellt sicher, dass Sie fachkundige Anleitung erhalten und gleichzeitig interne Fähigkeiten aufbauen. Der Service ist ideal für Unternehmen mit bestehenden Testkapazitäten, die spezielles AWS-Fachwissen benötigen, um Distributed Load Testing auf AWS effektiv zu implementieren.

Was bieten CDP-Techniker

CDP-Techniker führen Sie durch die Grundlagen von Leistungstests und Distributed Load Testing auf der AWS-Architektur. Sie bieten Anleitungen zur JMeter-, k6- und Locust-Skriptstruktur und zur Entwicklung von Testskripten, unterstützen bei der Bereitstellung von CloudFormation Vorlagen und

bewerten die Testergebnisse mit Empfehlungen zur Leistungsoptimierung. Der Support umfasst eine Analyse der Ressourcennutzung, die Abstimmung von Best Practices und eine umfassende Beratung von der ersten Einrichtung bis zur Ergebnisanalyse, sodass der Wissenstransfer an Ihr Team ermöglicht wird.

Pflichten des Kunden

Ihr Team kümmert sich um Konfigurationen auf Anwendungsebene, die Entwicklung von Testskripten und die Überprüfung von Testszenarien. Sie behalten die Verantwortung für die tatsächliche Ausführung und den Betrieb der Tests, einschließlich aller Testaktivitäten vor, während und nach Leistungstests.

Wichtigste Vorteile

Kurzfristige CDP-Engagements reduzieren das Risiko durch fachkundige Aufsicht, kontextbezogene Beratung speziell für Ihre Arbeitslast, Empfehlungen zur Leistungsoptimierung, schnellere Problemlösung, Abstimmung bewährter Verfahren und umfassenden Support, während gleichzeitig die Eigenverantwortung und die Weiterentwicklung der Fähigkeiten Ihres Teams gewahrt bleiben.

Unterstützte Architekturen

Distributed Load Testing auf AWS unterstützt Tests für Webanwendungen, APIs, Microservices und serverlose Architekturen in großem Maßstab und nutzt dabei die Distributed Load Testing on AWS-Lösung. Die Testmöglichkeiten gehen weit über diese üblichen Anwendungsfälle hinaus und umfassen Datenbanken, TCP/UDP Protokolle, LDAP-Verzeichnisse, SMTP-Mailserver und viele andere Systeme und Protokolle, die eine Leistungsvalidierung unter Last erfordern.

Erste Schritte

Organizations, die an CDP Short Term Engagements for Distributed Load Testing auf AWS interessiert sind, können sich hier direkt über die AWS-Website anmelden und „Use Case Implementation“ für Ihren Schwerpunktbereich auswählen.

Außerhalb des Geltungsbereichs

CDP bietet keine Entwicklung von benutzerdefinierten Testskripten (nur Anleitungen), verwaltet keine Testausführungen und organisiert auch keine kundenspezifischen praktischen Übungen oder Workshops. On-site Support liegt ebenfalls außerhalb des Geltungsbereichs.

Bereitstellen der Lösung

Diese Lösung bietet drei Hosting-Optionen für Webkonsolen. Die Backend-Architektur und das Amazon Cognito Cognito-Authentifizierungsmodell sind für alle Optionen identisch.

Hosting-Option	Description	Wann sollte dies verwendet werden?
CloudFront + S3 (Standard)	Die Webkonsole wird in Amazon S3 gehostet und über Amazon bereitgestellt CloudFront.	Für die meisten Bereitstellungen empfohlen. Bietet globales Edge-Caching und erfordert keine zusätzliche Infrastruktur.
ALB + ECS Fargate	Die Webkonsole läuft auf einem ECS Fargate-Dienst hinter einem Application Load Balancer mit einem vom Kunden bereitgestellten ACM-Zertifikat und einer benutzerdefinierten Domain. Eine AWS WAF WAF-Web-ACL wird vor der ALB bereitgestellt, um gängige webbasierte Angriffe zu filtern.	Verwenden Sie diese Option, wenn VPC Block Public Access (BPA) - Richtlinien den Datenverkehr von öffentlichen CloudFront Verteilungen verhindern, wenn Ihr Unternehmen keine Gefahr für das öffentliche Internet benötigt, wenn Sie einen benutzerdefinierten Domainnamen für die Webkonsole benötigen oder wenn Sie Firewall-Schutz für Webanwendungen mit anpassbaren Regeln benötigen. Häufig in regulierten Branchen (Finanzen, Gesundheitswesen, Regierung) und Umgebungen mit strengen Compliance-Anforderungen.
Headless (bringen Sie Ihren eigenen Webserver mit)	Die Lösung stellt nur das Backend bereit und bietet eine herunterladbare ZIP-Datei der statischen Ressourcen der Webkonsole zum Hosten auf Ihrem eigenen Webserver.	Verwenden Sie diese Option, wenn Sie die Webkonsole auf einer bestehenden Infrastruktur außerhalb von AWS hosten möchten, wenn Unternehmensrichtlinien das Hosting auf verwalteten Webservern vorschreiben oder wenn Ihr Unternehmen unabhängig von den

Hosting-Option	Description	Wann sollte dies verwendet werden?
		Authentifizierungsmechanismen eine vollständige Netzwerkisolierung ohne öffentlich zugängliche Endpunkte vorschreibt.

AWS Launch Wizard ist die empfohlene Bereitstellungsmethode für diese Lösung. Es bietet:

- Ein geführtes Konfigurationserlebnis mit detaillierten Hilfefenstern bei jedem Schritt
- Eine zentrale Seite zur Überwachung des Zustands all Ihrer Bereitstellungen
- Angabe, wann eine neuere Version der Lösung für die Bereitstellung oder Aktualisierung verfügbar ist

Alternativ können Sie die Lösung direkt mithilfe einer CloudFormation AWS-Vorlage bereitstellen.

Überblick über den Bereitstellungsprozess

Bevor Sie die Lösung bereitstellen, sollten Sie sich mit den Kosten, der Architektur, der Sicherheit und anderen Überlegungen befassen, die weiter oben in diesem Handbuch erörtert wurden.

Bereitstellungszeit: Ungefähr 15 Minuten für den Haupt-Stack plus 5 Minuten für jede weitere Region

> **Note**
>
> Diese Lösung beinhaltet Datenerfassungsmetriken für AWS. Wir verwenden diese Daten, um besser zu verstehen, wie Kunden diese Lösung und die damit verbundenen Services und Produkte nutzen. AWS ist Eigentümer der im Rahmen dieser Umfrage gesammelten Daten. Die Datenerfassung unterliegt dem AWS-Datenschutzhinweis.

> **Note**
>
> Sie sind für die Kosten der AWS-Services verantwortlich, die Sie beim Betrieb dieser Lösung in Anspruch nehmen. Weitere Informationen zu den Kosten finden Sie im Abschnitt Kosten in

diesem Leitfaden. Weitere Informationen finden Sie auf der Preisseite für jeden AWS-Service, der in dieser Lösung verwendet wird.

Bereitstellung mit dem AWS Launch Wizard

Diese Lösung umfasst einen geführten Bereitstellungsprozess mithilfe des AWS Launch Wizard. Gehen Sie wie folgt vor, um Distributed Load Testing auf AWS in Ihrem Konto bereitzustellen.

1. Melden Sie sich bei der AWS-Managementkonsole an und klicken Sie auf die folgende Schaltfläche, um den Bereitstellungsprozess zu starten.

2. Wenn für die Lösung mehr als ein Bereitstellungsmuster verfügbar ist, wählen Sie das aus, das für Ihren Anwendungsfall am besten geeignet ist.
3. Wählen Sie eine Version für die Bereitstellung aus. Die neueste Version wird empfohlen.
4. Wählen Sie die Schaltfläche Deployment Wizard starten.

Anschließend folgen Sie einer Reihe von Schritten, um die Informationen zu sammeln, die für die Bereitstellung der Lösung erforderlich sind. Die Bereitstellung der erforderlichen Ressourcen dauert ungefähr 15 Minuten.

Wählen Sie Ihre Bereitstellung aus der Bereitstellungsliste aus, um deren Status einzusehen.

Mit AWS bereitstellen CloudFormation

Diese Lösung verwendet CloudFormation AWS-Vorlagen und -Stacks, um ihre Bereitstellung zu automatisieren. Die CloudFormation Vorlagen spezifizieren die in dieser Lösung enthaltenen AWS-Ressourcen und ihre Eigenschaften. Der CloudFormation Stack stellt die Ressourcen bereit, die in den Vorlagen beschrieben sind.

CloudFormation AWS-Vorlage

Sie können die CloudFormation Vorlagen für diese Lösung herunterladen, bevor Sie sie bereitstellen. Diese Lösung verwendet AWS CloudFormation , um die Bereitstellung von Distributed Load Testing

auf AWS zu automatisieren. Es enthält die folgenden CloudFormation AWS-Vorlagen, die Sie vor der Bereitstellung herunterladen können:

distribu

load-testing-on-aws.template — Verwenden Sie diese Vorlage, um die Lösung mit dem standardmäßigen + S3-Webkonsolen-Hosting zu starten. CloudFront In der Standardkonfiguration werden die Kern- und Unterstützungsdienste bereitgestellt, die in den AWS-Services in diesem Lösungsabschnitt enthalten sind. Sie können die Vorlage jedoch an Ihre spezifischen Anforderungen anpassen.

distribu

load-testing-on-aws-alb-ecs.template — Verwenden Sie diese Vorlage, um die Lösung mit der auf ECS Fargate gehosteten Webkonsole hinter einem ALB zu starten. Anweisungen zur Bereitstellung finden Sie unter Deploy using ALB + ECS Fargate.

distribu

load-testing-on-aws-headless.template — Verwenden Sie diese Vorlage nur, um das Lösungs-Backend zu starten, wobei die Web-Konsolen-Assets zum Herunterladen zur Verfügung stehen. Anweisungen zur Bereitstellung finden Sie unter Bereitstellen mithilfe einer Headless-Vorlage (bringen Sie Ihren eigenen Webserver mit).

> **Note**
>
> CloudFormation AWS-Ressourcen werden aus Konstrukten des AWS Cloud Development Kit (AWS CDK) erstellt. Wenn Sie diese Lösung bereits bereitgestellt haben, finden Sie Anweisungen zum Update unter Lösung aktualisieren.

Starten Sie den Stack (Standard, CloudFront + S3-gehostete Webkonsole)

Gehen Sie wie folgt vor, um die Lösung Distributed Load Testing on AWS in Ihrem Konto bereitzustellen. Diese automatisierte CloudFormation AWS-Vorlage stellt Distributed Load Testing auf AWS bereit.

1. Melden Sie sich bei der AWS-Managementkonsole an und klicken Sie auf die Schaltfläche, um die CloudFormation Vorlage zu starten.

   Launch solution

   Alternativ können Sie die Vorlage als Ausgangspunkt für Ihre eigene Implementierung herunterladen.
2. Die Vorlage wird standardmäßig in der Region USA Ost (Nord-Virginia) gestartet. Um diese Lösung in einer anderen AWS-Region zu starten, verwenden Sie die Regionsauswahl in der Navigationsleiste der Konsole.

   > **Note**
   >
   > Diese Lösung verwendet Amazon Cognito, das derzeit nur in bestimmten AWS-Regionen verfügbar ist. Daher müssen Sie diese Lösung in einer AWS-Region starten, in der Amazon Cognito verfügbar ist. Die aktuelle Serviceverfügbarkeit nach Regionen finden Sie in der regionalen AWS-Serviceliste.
3. Vergewissern Sie sich auf der Seite Stack erstellen, dass die richtige Vorlagen-URL im Textfeld Amazon S3 S3-URL angezeigt wird, und wählen Sie Weiter.
4. Weisen Sie Ihrem Lösungsstapel auf der Seite „Stack-Details angeben“ einen Namen zu.
5. Überprüfen Sie unter Parameter die Parameter für die Vorlage und ändern Sie sie nach Bedarf. Diese Lösung verwendet die folgenden Standardwerte.

Parameter	Standard	Description
Name des Administrators	<Erfordert Eingabe>	Benutzername für den ursprünglichen Lösungsadministrator.
E-Mail-Adresse des Administrators	*<Requires input>*	E-Mail-Adresse des Administratorbenutzers. Nach dem Start wird eine E-Mail mit Anweisungen zur Anmeldung auf der Konsole an diese Adresse gesendet.

Parameter	Standard	Description
Bestehende VPC-ID	<Optional input>	Wenn Sie über eine VPC verfügen, die Sie verwenden möchten und die bereits erstellt wurde, geben Sie die ID einer vorhandenen VPC in derselben Region ein, in der der Stack bereitgestellt wurde. Zum Beispiel vpc-1a2b3c4d5e6f.
Erstes vorhandenes Subnetz	<Optional input>	Die ID des ersten Subnetzes innerhalb Ihrer vorhandenen VPC. Dieses Subnetz benötigt eine Route zum Internet, um das Container-Image für die Ausführung von Tests abzurufen. Zum Beispiel subnet-7h8i9j0k.
Zweites vorhandenes Subnetz	<Optional input>	Die ID des zweiten Subnetzes innerhalb der vorhandenen VPC. Dieses Subnetz benötigt eine Route zum Internet, um das Container-Image für die Ausführung von Tests abzurufen. Zum Beispiel subnet-1x2y3z.
Geben Sie einen gültigen CIDR-Block für die Lösung an, um eine VPC zu erstellen	192.168.0. 0/16	Sie können diesen Parameter leer lassen, wenn Sie eine bestehende VPC verwenden

Parameter	Standard	Description
Geben Sie einen gültigen CIDR-Block für Subnetz A an, damit die Lösung eine VPC erstellen kann	192.168.0. 0/20	CIDR-Block für Subnetz A der AWS Fargate VPC
Geben Sie einen gültigen CIDR-Block für Subnetz B an, damit die Lösung eine VPC erstellen kann	192.168.16. 0/20	CIDR-Block für Subnetz B der AWS Fargate VPC
Stellen Sie einen CIDR-Bloc k bereit, um den ausgehend en Verkehr von Fargate-A ufgaben zuzulassen	0.0.0. 0/0	CIDR-Block, der den ausgehenden Zugriff auf Amazon ECS-Container einschränkt.
Auto-update Container-Bild	No	Verwenden Sie bis zur nächsten Nebenversion automatisch das aktuellst e und sicherste Image. Bei Auswahl dieser Option No wird das Image in der ursprünglich veröffentlichten Version ohne Sicherhei tsupdates abgerufen.
Stellen Sie den optionalen MCP-Server bereit	No	Stellen Sie den optionale n Remote-MCP-Server bereit und verwenden Sie AgentCore Gateway, um KI-Anwendungen mit Distribut ed Load Testing auf AWS zu verbinden.

Parameter	Standard	Description
Bild-URI des Load Testers	<Optional input>	URI eines benutzerdefinierten Loadtester-Container-Images aus der privaten Registrierung von Amazon Elastic Container Registry (Amazon ECR) (z. B. `123456789012.dkr.ecr.us-gov-west-1.amazonaws.com/dlt-load-tester:v4.0.0`). Erforderlich für AWS-Bereitstellungen GovCloud (USA), auf die ECS-Aufgaben nicht zugreifen `public.ecr.aws` können. Wenn das Feld leer ist, wird das standardmäßige öffentliche Image verwendet. Weitere Informationen finden Sie unter Tester-Image-URI laden.

6. Wählen Sie Weiter aus.
7. Wählen Sie auf der Seite Configure stack options (Stack-Optionen konfigurieren) Next (Weiter) aus.
8. Überprüfen und bestätigen Sie die Einstellungen auf der Seite Review. Markieren Sie das Kästchen, um zu bestätigen, dass die Vorlage AWS Identity and Access Management (IAM) - Ressourcen erstellt.
9. Wählen Sie Stack erstellen aus, um den Stack bereitzustellen.

Sie können den Status des Stacks in der CloudFormation AWS-Konsole in der Spalte Status anzeigen. Sie sollten in etwa 15 Minuten den Status CREATE_COMPLETE erhalten.

> **Note**
>
> Zusätzlich zur primären AWS-Lambda-Funktion umfasst diese Lösung die Lambda-Funktion für benutzerdefinierte Ressourcen, die nur während der Erstkonfiguration oder wenn Ressourcen aktualisiert oder gelöscht werden, ausgeführt wird.

Beim Ausführen dieser Lösung ist die Lambda-Funktion für benutzerdefinierte Ressourcen inaktiv. Löschen Sie diese Funktion jedoch nicht, da sie für die Verwaltung der zugehörigen Ressourcen erforderlich ist.

Starten Sie den Stack (von ALB + von ECS Fargate gehostete Webkonsole)

Voraussetzungen

Bevor Sie die ALB + ECS Fargate-Vorlage bereitstellen, müssen Sie die folgenden Schritte ausführen:

1. ACM-Zertifikat — Fordern Sie ein SSL/TLS Zertifikat an oder importieren Sie es in AWS Certificate Manager (ACM) in derselben Region, in der Sie den Stack bereitstellen werden. Das Zertifikat muss den Domainnamen abdecken, den Sie für die Webkonsole verwenden möchten.
2. Domainbesitz — Sie müssen den Domainnamen, den Sie im ConsoleDomainNameParameter angeben, besitzen oder kontrollieren. Nach der Bereitstellung müssen Sie in der Lage sein, DNS-Einträge für diese Domain zu erstellen.

Starten des -Stacks

1. Melden Sie sich bei der AWS-Managementkonsole an und klicken Sie auf die Schaltfläche, um die CloudFormation Vorlage zu starten.

   Launch solution

   Alternativ können Sie die Vorlage als Ausgangspunkt für Ihre eigene Implementierung herunterladen.
2. Die Vorlage wird standardmäßig in der Region USA Ost (Nord-Virginia) gestartet. Um in einer anderen AWS-Region zu starten, verwenden Sie die Regionsauswahl in der Navigationsleiste der Konsole.

> **Note**
>
> Diese Lösung verwendet Amazon Cognito, das derzeit nur in bestimmten AWS-Regionen verfügbar ist. Daher müssen Sie diese Lösung in einer AWS-Region starten, in der Amazon Cognito verfügbar ist. Die aktuelle Serviceverfügbarkeit nach Regionen finden Sie in der regionalen AWS-Serviceliste.

3. Vergewissern Sie sich auf der Seite Stack erstellen, dass die richtige Vorlagen-URL im Textfeld Amazon S3 S3-URL angezeigt wird, und wählen Sie Weiter.
4. Weisen Sie Ihrem Lösungsstapel auf der Seite „Stack-Details angeben“ einen Namen zu.
5. Überprüfen Sie unter Parameter die Parameter für die Vorlage und ändern Sie sie nach Bedarf. Diese Vorlage verwendet zusätzlich zu den Standardparametern der Standardvorlage die folgenden Parameter.

Parameter	Standard	Description
Domänenname der Konsole	<Erfordert Eingabe>	Der benutzerdefinierte Domainname für die Webkonsole (z. B.`dlt.example.com`). Diese Domain muss mit dem ACM-Zertifikat übereinst immen, das Sie in den Voraussetzungen erstellt haben.
ACM-Zertifikat ARN	<Erfordert Eingabe>	Der ARN des ACM-Zerti fikats für die Domain (z. B.`arn:aws:acm:us-eas t-1:123456789012:c ertificate/abcd123 4-ef56-gh78-ij90-k lmnopqrstuv`). Muss sich in derselben Region wie der Stack befinden.

Parameter	Standard	Description
Name des Administrators	<Erfordert Eingabe>	Benutzername für den ursprünglichen Lösungsad ministrator.
E-Mail-Adresse des Administr ators	<Erfordert Eingabe>	E-Mail-Adresse des Administr atorbenutzers. Nach dem Start wird eine E-Mail mit Anweisungen zur Anmeldung auf der Konsole an diese Adresse gesendet.
Bestehende VPC-ID	<Optional input>	Wenn Sie über eine VPC verfügen, die Sie verwenden möchten und die bereits erstellt wurde, geben Sie die ID einer vorhandenen VPC in derselben Region ein, in der der Stack bereitges tellt wurde. Zum Beispiel vpc-1a2b3c4d5e6f.
Erstes vorhandenes Subnetz	<Optional input>	Die ID des ersten Subnetzes innerhalb Ihrer vorhanden en VPC. Dieses Subnetz benötigt eine Route zum Internet, um das Container -Image für die Ausführung von Tests abzurufen. Zum Beispiel subnet-7h8i9j0k.

Parameter	Standard	Description
Zweites vorhandenes Subnetz	<Optional input>	Die ID des zweiten Subnetzes innerhalb der vorhandenen VPC. Dieses Subnetz benötigt eine Route zum Internet, um das Container-Image für die Ausführung von Tests abzurufen. Zum Beispiel subnet-1x2y3z.
Geben Sie einen gültigen CIDR-Block für die Lösung an, um eine VPC zu erstellen	192.168.0. 0/16	Sie können diesen Parameter leer lassen, wenn Sie eine bestehende VPC verwenden.
Geben Sie einen gültigen CIDR-Block für Subnetz A an, damit die Lösung eine VPC erstellen kann	192.168.0. 0/20	CIDR-Block für Subnetz A der AWS Fargate VPC.
Geben Sie einen gültigen CIDR-Block für Subnetz B an, damit die Lösung eine VPC erstellen kann	192.168.16. 0/20	CIDR-Block für Subnetz B der AWS Fargate VPC.
Stellen Sie einen CIDR-Block bereit, um den ausgehenden Verkehr von Fargate-Aufgaben zuzulassen	0.0.0. 0/0	CIDR-Block, der den ausgehenden Zugriff auf Amazon ECS-Container einschränkt.

Parameter	Standard	Description
Auto-update Container-Bild	No	Verwenden Sie bis zur nächsten Nebenversion automatisch das aktuellste und sicherste Image. Bei Auswahl dieser Option No wird das Image in der ursprünglich veröffentlichten Version ohne Sicherheitsupdates abgerufen.
Image-URI der Webkonsole	<Optional input>	URI eines benutzerdefinierten Container-Images für die Webkonsole aus der privaten Amazon ECR-Registrierung (z. B. `123456789012.dkr.ecr.us-east-1.amazonaws.com/my-web-console:latest`). Wenn das Feld leer ist, wird das öffentliche Standardbild verwendet. Weitere Informationen finden Sie unter Bild der Webkonsole.
WAF bereitstellen	Yes	Stellen Sie vor der ALB eine AWS-WAF-Web-ACL mit von AWS verwalteten Regeln bereit, um gängige webbasierte Angriffe zu filtern. Auf No einstellen, um die WAF-Bereitstellung zu deaktivieren.

Parameter	Standard	Description
Stellen Sie den optionalen MCP-Server bereit	No	Stellen Sie den optionalen Remote-MCP-Server bereit und verwenden Sie AgentCore Gateway, um KI-Anwendungen mit Distributed Load Testing auf AWS zu verbinden.

6. Wählen Sie Weiter aus.
7. Wählen Sie auf der Seite Configure stack options (Stack-Optionen konfigurieren) Next (Weiter) aus.
8. Überprüfen und bestätigen Sie die Einstellungen auf der Seite Review. Markieren Sie das Kästchen, um zu bestätigen, dass die Vorlage AWS Identity and Access Management (IAM) - Ressourcen erstellt.
9. Wählen Sie Stack erstellen aus, um den Stack bereitzustellen.

Sie können den Status des Stacks in der CloudFormation AWS-Konsole in der Spalte Status anzeigen. Sie sollten in etwa 15 Minuten den Status CREATE_COMPLETE erhalten.

Post-deployment Konfiguration

Nach Abschluss der Stack-Erstellung müssen Sie DNS so konfigurieren, dass Ihre benutzerdefinierte Domain auf die ALB verweist.

1. Navigieren Sie zur Registerkarte Ausgaben des CloudFormation Stacks und kopieren Sie den ALBDnsNameWert.
2. Erstellen Sie in Ihrem DNS-Anbieter einen der folgenden Einträge:
   - CNAME-Eintrag — Ordnet Ihre Domain dem ALB-DNS-Namen zu. Geeignet für Subdomains (z. B.). `dlt.example.com`
   - Alias-Datensatz (Route 53) — Wenn Sie Amazon Route 53 verwenden, können Sie einen Alias-Datensatz erstellen, der auf die ALB verweist. Dies ist beispielsweise für Zone-Apex-Domains erforderlich und vermeidet CNAME-Suchgebühren. `example.com`

     Erstellen Sie in der Route 53 53-Konsole einen A-Datensatz mit aktiviertem Alias, wählen Sie Alias to Application und Classic Load Balancer, wählen Sie die Region und dann ALB aus.

3. Warten Sie, bis die DNS-Propagierung abgeschlossen ist. Sie können dies überprüfen mit:

```
$ dig dlt.example.com
```

4. Greifen Sie auf die Webkonsole zu `https://<your-domain>` (z. B.`https://dlt.example.com`).

Die URL der Webkonsole ist auch auf der Registerkarte CloudFormation Outputs als consoleUrl verfügbar.

> **Note**
>
> Die DNS-Übertragung kann je nach Ihrem DNS-Anbieter und den TTL-Einstellungen zwischen einigen Minuten und 48 Stunden dauern.

WAF-Integration (optional)

Die standardmäßig verwalteten Regelgruppen (Core Rule Set, Amazon IP Reputation List und Anonymous IP List) bieten Basisschutz für die meisten Bereitstellungen. Sie können die WAF-Bereitstellung deaktivieren, indem Sie den Parameter Deploy WAF CloudFormation auf setzen. No Es ist keine zusätzliche WAF-Konfiguration erforderlich, es sei denn, Sie haben spezielle Sicherheitsanforderungen.

Wenn Sie die WAF-Regeln anpassen müssen, können Sie die Konfiguration wie folgt ändern:

1. Öffnen Sie die AWS WAF WAF-Konsole in derselben Region wie Ihre Bereitstellung.
2. Wählen Sie die von der Lösung erstellte Web-ACL aus. Der Web-ACL-Name befindet sich in den CloudFormation Stack-Ressourcen.
3. Wählen Sie Regeln, um die Regeln in der Web-ACL anzuzeigen, hinzuzufügen, zu entfernen oder zu ändern.
4. Sie können je nach Ihren Sicherheitsanforderungen weitere von AWS verwaltete Regelgruppen, benutzerdefinierte Regeln oder ratenbasierte Regeln hinzufügen.

Weitere Informationen finden Sie unter AWS WAF im AWS WAF Developer Guide.

Starten Sie den Stack (Headless)

Starten des -Stacks

1. Melden Sie sich bei der AWS-Managementkonsole an und klicken Sie auf die Schaltfläche, um die CloudFormation Vorlage zu starten.

   Launch solution

   Alternativ können Sie die Vorlage als Ausgangspunkt für Ihre eigene Implementierung herunterladen.
2. Die Vorlage wird standardmäßig in der Region USA Ost (Nord-Virginia) gestartet. Um in einer anderen AWS-Region zu starten, verwenden Sie die Regionsauswahl in der Navigationsleiste der Konsole.

   > **Note**
   >
   > Diese Lösung verwendet Amazon Cognito, das derzeit nur in bestimmten AWS-Regionen verfügbar ist. Daher müssen Sie diese Lösung in einer AWS-Region starten, in der Amazon Cognito verfügbar ist. Die aktuelle Serviceverfügbarkeit nach Regionen finden Sie in der regionalen AWS-Serviceliste.
3. Vergewissern Sie sich auf der Seite Stack erstellen, dass die richtige Vorlagen-URL im Textfeld Amazon S3 S3-URL angezeigt wird, und wählen Sie Weiter.
4. Weisen Sie Ihrem Lösungsstapel auf der Seite „Stack-Details angeben“ einen Namen zu.
5. Überprüfen Sie unter Parameter die Parameter für die Vorlage und ändern Sie sie nach Bedarf. Diese Vorlage verwendet dieselben Parameter wie die Standardvorlage (Administratorname, Administrator-E-Mail, VPC/subnet Konfiguration, Egress CIDR, Auto-update Container-Image und Deploy Optional MCP Server). Es sind keine zusätzlichen Parameter erforderlich.
6. Wählen Sie Weiter aus.
7. Wählen Sie auf der Seite Configure stack options (Stack-Optionen konfigurieren) Next (Weiter) aus.
8. Überprüfen und bestätigen Sie die Einstellungen auf der Seite Review. Markieren Sie das Kästchen, um zu bestätigen, dass die Vorlage AWS Identity and Access Management (IAM) -Ressourcen erstellt.

9. Wählen Sie Stack erstellen aus, um den Stack bereitzustellen.

Sie können den Status des Stacks in der CloudFormation AWS-Konsole in der Spalte Status anzeigen. Sie sollten in etwa 15 Minuten den Status CREATE_COMPLETE erhalten.

Post-deployment Konfiguration

Nach Abschluss der Stack-Erstellung können Sie mithilfe der DLT-CLI mit der Lösung interagieren, um direkt von Ihrem Terminal aus auf die REST-API zuzugreifen oder die Webkonsole auf Ihrem eigenen Webserver zu hosten.

Um die Webkonsole zu hosten:

1. Navigieren Sie zur Registerkarte Ausgaben des CloudFormation Stacks.
2. Wählen Sie den ConsoleAssetsBucketAusgabelink, um den S3-Bucket in der Konsole zu öffnen.
3. Laden Sie die `dlt-web-console.zip` Datei aus dem Konsolen-Bucket herunter.
4. Extrahieren Sie den ZIP-Inhalt und hosten Sie die Website auf Ihrem Webserver.
5. Konfigurieren Sie Amazon Cognito so, dass die URL Ihres Webservers zulässig ist:
   a. Öffnen Sie die Amazon Cognito Cognito-Konsole und wählen Sie den Benutzerpool aus, der durch den Stack erstellt wurde.
   b. Wählen Sie App-Clients und wählen Sie den App-Client aus. Die App-Client-ID finden Sie auf der Registerkarte Outputs im CloudFormation Stack unter CognitoAppClientID.
   c. Wählen Sie im Abschnitt Anmeldeseiten die Option Bearbeiten aus.
   d. Fügen Sie unter Zulässige Callback-URLs die URL Ihres Webservers und die URL mit einem abschließenden Schrägstrich (z. B. `https://dlt.example.com` und) hinzu. `https://dlt.example.com/`
   e. Fügen Sie unter Zulässige Abmelde-URLs dieselben URLs hinzu.
   f. Wählen Sie Änderungen speichern aus.

      Weitere Informationen finden Sie unter Application-specific Einstellungen mit App-Clients im Amazon Cognito Developer Guide.

> **Important**
>
> Wenn Sie die Webkonsole hosten, sind Sie für die Konfiguration von HTTPS, Zugriffskontrollen und Sicherheitsverstärkung auf Ihrem Webserver verantwortlich. Wir empfehlen dringend HTTPS für den produktiven Einsatz.

> **Note**
>
> Die Ressourcen der Webkonsole enthalten eine Konfigurationsdatei mit der API-Endpunkt-URL und den Cognito-Einstellungen. Diese Werte werden bei der Stack-Erstellung vorkonfiguriert. Wenn Sie die Assets auf einen anderen Server verschieben, stellen Sie sicher, dass die Konfigurationsdatei intakt bleibt.

In AWS-Regionen GovCloud (USA) bereitstellen

Die Lösung unterstützt die Bereitstellung in den Regionen AWS GovCloud (US-West) und AWS GovCloud (US-East). Sie müssen jedoch die Startschaltflächen in diesem Abschnitt anstelle der Startschaltflächen in anderen Abschnitten dieses Handbuchs verwenden. Die Startschaltflächen in anderen Abschnitten verweisen auf Vorlagen, die für die AWS-Standardpartition bereitgestellt wurden und auf die von AWS-Regionen GovCloud (USA) aus nicht zugegriffen werden kann.

> **Note**
>
> Die standardmäßige Hosting-Option CloudFront + S3-Webkonsole ist in AWS-Regionen GovCloud (USA) nicht verfügbar, da Amazon in der AWS-Partition GovCloud (USA) nicht verfügbar CloudFront ist. Verwenden Sie stattdessen die ALB + ECS Fargate-Vorlage oder die Headless-Vorlage.

Vor der Bereitstellung müssen Sie die Container-Images der Lösung in ein privates Amazon ECR-Repository in Ihrer AWS-Region GovCloud (USA) spiegeln, da ECS-Aufgaben in AWS GovCloud (USA) nicht auf die öffentliche ECR-Registrierung zugreifen können (`public.ecr.aws`). Anweisungen zum Spiegeln der Container-Images finden Sie im Abschnitt Load Tester-Image-URI dieses Handbuchs.

Launch solution distribu

load-testing-on-aws-alb-ecs.template — Startet die Lösung mit der auf ECS Fargate gehosteten Webkonsole hinter einem ALB. Folgen Sie den Anweisungen unter Stack starten (von ALB + ECS Fargate gehostete Webkonsole) und geben Sie zusätzlich die Parameter Load Tester Image URI und Web Console Image URI an.

Launch solution distribu

load-testing-on-aws-headless.template — Startet nur das Lösungs-Backend. Folgen Sie den Anweisungen unter Stack starten (Headless) und geben Sie zusätzlich den Parameter Load Tester Image URI an.

Alternativ können Sie die Vorlagen als Ausgangspunkt für Ihre eigene Implementierung herunterladen. Wählen Sie oben eine Startschaltfläche und kopieren Sie dann die Vorlagen-URL aus dem Feld Amazon S3 S3-URL auf der Seite Stack erstellen.

Multi-Region Bereitstellung

Zeit für die Bereitstellung: Ungefähr 5 Minuten pro Region

Sie können Tests in mehreren Regionen durchführen.

Wenn Sie die Distributed Load Testing-Lösung bereitstellen, erstellt sie eine regionale CloudFormation Vorlage im S3-Bucket für Szenarien. Die URL für diese Vorlage ist in den CloudFormation Ausgaben Ihres Haupt-Stacks unter dem Schlüssel „RegionalCFTemplate“ aufgeführt.

Um einen Test mit mehreren Regionen durchzuführen, müssen Sie die regionale CloudFormation Vorlage in jeder Region bereitstellen, in der Sie den Test ausführen möchten.

> **Note**
>
> Jedes AWS-Konto kann nur einen regionalen Stack pro Region verwenden. Außerdem kann der regionale Stack nicht in derselben Region wie der Haupt-Stack verwendet werden.

Sie können die regionale Vorlage wie folgt installieren:

1. Navigieren Sie in der Webkonsole der Lösung im linken Menü zu Dashboard.
2. Verwenden Sie das Zwischenablage-Symbol, um den CloudFormation Vorlagenlink in Amazon S3 zu kopieren.
3. Melden Sie sich bei der CloudFormation AWS-Konsole an und wählen Sie die richtige Region aus.
4. Vergewissern Sie sich auf der Seite Stack erstellen, dass die richtige Vorlagen-URL im Textfeld Amazon S3 S3-URL angezeigt wird, und wählen Sie Weiter.
5. Weisen Sie Ihrem Lösungsstapel auf der Seite „Stack-Details angeben“ einen Namen zu.
6. Überprüfen Sie unter Parameter die Parameter für die Vorlage und ändern Sie sie nach Bedarf. Diese Lösung verwendet die folgenden Standardwerte.

Parameter	Standard	Description
Bestehende VPC-ID	<Optional input>	Wenn Sie über eine VPC verfügen, die Sie verwenden möchten und die bereits erstellt wurde, geben Sie die ID einer vorhandenen VPC in derselben Region ein, in der der Stack bereitgestellt wurde. Zum Beispiel vpc-1a2b3c4d5e6f.
Erstes vorhandenes Subnetz	<Optional input>	Die ID des ersten Subnetzes innerhalb Ihrer vorhandenen VPC. Dieses Subnetz benötigt eine Route zum Internet, um das Container-Image für die Ausführung von Tests abzurufen. Zum Beispiel subnet-7h8i9j0k.
Zweites vorhandenes Subnetz	<Optional input>	Die ID des zweiten Subnetzes innerhalb der vorhandenen VPC. Dieses Subnetz benötigt eine Route zum Internet, um

Parameter	Standard	Description
		das Container-Image für die Ausführung von Tests abzurufen. Zum Beispiel subnet-1x2y3z.
Geben Sie einen gültigen CIDR-Block für die Lösung an, um eine VPC zu erstellen	192.168.0. 0/16	Wenn Sie keine Werte für eine bestehende VPC angeben, enthält der CIDR-Block für die von der Lösung erstellte Amazon-VPC die IP-Adresse für AWS Fargate.
Stellen Sie einen CIDR-Bloc k bereit, um den ausgehend en Verkehr von Fargate-A ufgaben zuzulassen	0.0.0. 0/0	CIDR-Block, der den ausgehenden Zugriff auf Amazon ECS-Container einschränkt.

7. Wählen Sie Weiter aus.
8. Wählen Sie auf der Seite Configure stack options (Stack-Optionen konfigurieren) Next (Weiter) aus.
9. Überprüfen und bestätigen Sie die Einstellungen auf der Seite Review. Achten Sie darauf, das Kästchen zu aktivieren, das bestätigt, dass die Vorlage AWS Identity and Access Management (IAM) -Ressourcen erstellt.
10. Wählen Sie Stack erstellen aus, um den Stack bereitzustellen.

    Sie können den Status des Stacks in der CloudFormation AWS-Konsole in der Spalte Status anzeigen. Sie sollten in etwa fünf Minuten den Status CREATE_COMPLETE erhalten.

Wenn die Regionen erfolgreich bereitgestellt wurden, werden sie in der Webkonsole angezeigt. Wenn Sie einen Test erstellen, werden alle verfügbaren Regionen im Dashboard und unter Szenarioerstellung aufgelistet. Sie können einem Test im Schritt Traffic Shape der Szenarioerstellung eine Region hinzufügen.

Die Lösung erstellt ein DynamoDB-Element für jede bereitgestellte Region in der Szenariotabelle, das die erforderlichen Informationen zu den Testressourcen in dieser Region enthält. Sie können die Testergebnisse in der Webkonsole nach Region sortieren. Verwenden Sie CloudWatch Amazon-

Metriken, um aggregierte Ergebnisse für alle Regionen in einem Test mit mehreren Regionen anzuzeigen. Den Quellcode für das Diagramm finden Sie nach Abschluss des Tests in den Testergebnissen.

> **Note**
>
> Sie können den regionalen Stack ohne die Webkonsole starten. Besorgen Sie sich einen Link zur regionalen Vorlage im Amazon S3 S3-Szenario-Bucket und geben Sie ihn als Quelle an, wenn Sie den regionalen Stack in der erforderlichen Region starten. Alternativ können Sie die Vorlage herunterladen und als Quelle für die gewünschte Region hochladen.

Benutze die Lösung

Dieser Abschnitt enthält eine umfassende Anleitung zur Verwendung der Distributed Load Testing on AWS-Lösung, von der Erstellung Ihres ersten Testszenarios bis hin zur Analyse detaillierter Ergebnisse. Der Workflow umfasst die Erstellung eines Testszenarios, die Ausführung eines Tests und die Untersuchung der Testergebnisse.

Erstellen Sie ein Testszenario

Die Erstellung eines Testszenarios umfasst vier Hauptschritte: Konfiguration der allgemeinen Einstellungen, Definition des Szenarios, Gestaltung von Verkehrsmustern und Überprüfung Ihrer Konfiguration.

Schritt 1: Allgemeine Einstellungen

Konfigurieren Sie die grundlegenden Parameter für Ihren Lasttest, einschließlich des Testnamens, der Beschreibung und der allgemeinen Konfigurationsoptionen.

Identifizierung des Tests

- Testname (erforderlich) — Ein beschreibender Name für Ihr Testszenario
- Testbeschreibung (erforderlich) — Zusätzliche Details zum Testzweck und zur Konfiguration
- Tags (optional) — Fügen Sie bis zu 5 Tags hinzu, um Ihre Testszenarien zu kategorisieren und zu organisieren

Optionen für die Terminplanung

Konfigurieren Sie, wann der Test ausgeführt werden soll:

- Jetzt ausführen — Führen Sie den Test unmittelbar nach der Erstellung aus.
- Einmal ausführen — Planen Sie den Test so, dass er an einem bestimmten Datum und zu einer bestimmten Uhrzeit ausgeführt wird.
- Nach einem Zeitplan ausführen — Verwenden Sie die cron-basierte Planung, um Tests automatisch in regelmäßigen Intervallen auszuführen. Sie können aus gängigen Mustern (stündlich, täglich, wöchentlich) wählen oder einen benutzerdefinierten Cron-Ausdruck definieren.

Einzelheiten zum akzeptierten Cron-Format, den unterstützten Mustern und Einschränkungen finden Sie in der Cron-Ausdrucksreferenz im Entwicklerhandbuch.

Arbeitsablauf planen

Wenn Sie einen Test planen, findet der folgende Workflow statt:

- Die Zeitplanparameter werden über Amazon API Gateway an die API der Lösung gesendet.
- Die API übergibt die Parameter an eine Lambda-Funktion, die einen Amazon EventBridge Scheduler-Zeitplan erstellt, der so konfiguriert ist, dass er am angegebenen Datum ausgeführt wird.
- Bei einmaligen Tests (Run Once) ruft der EventBridge Scheduler-Zeitplan die `api-services` Lambda-Funktion zum angegebenen Datum und zur angegebenen Uhrzeit auf, wodurch der Test ausgeführt wird.
- Bei wiederkehrenden Tests (Run on a Schedule) ruft der EventBridge Scheduler-Zeitplan die `api-services` Lambda-Funktion sofort und in der durch den Cron- oder Rate-Ausdruck definierten Schrittfrequenz bis zum Ablaufdatum auf.

Live-Daten

Aktivieren Sie das Kontrollkästchen Live-Daten einbeziehen, um Echtzeitmetriken anzuzeigen, während Ihr Test ausgeführt wird. Wenn diese Option aktiviert ist, können Sie Folgendes überwachen:

- Durchschnittliche Antwortzeit.
- Anzahl virtueller Benutzer.
- Erfolgreiche Anfragen zählen.
- Anzahl fehlgeschlagener Anfragen.

Die Live-Datenfunktion ermöglicht Echtzeitdiagramme mit Daten, die in Intervallen von einer Sekunde aggregiert werden. Weitere Informationen finden Sie unter Überwachung mit Live-Daten.

Schritt 2: Szenariokonfiguration

Definieren Sie das spezifische Testszenario und wählen Sie Ihr bevorzugtes Test-Framework aus.

Auswahl des Testtyps

Wählen Sie die Art des Belastungstests, den Sie durchführen möchten:

- Einfacher HTTP-Endpunkt — Testen Sie einen einzelnen API-Endpunkt oder eine Webseite mit einfacher Konfiguration.
- JMeter - Laden Sie JMeter-Testskripte (.jmx-Dateien oder .zip-Archive) hoch.
- k6 - Laden Sie k6-Testskripte (.js-Dateien oder .zip-Archive) hoch.
- Locust - Laden Sie Locust-Testskripte hoch (.py-Dateien oder .zip-Archive).

> **Note**
>
> Alle vier Testtypen basieren auf Komponenten von Drittanbietern. Die Lösung führt Tests über das Taurus Test Automation Framework durch, das je nach Testtyp JMeter, k6 oder Locust ausführt. Einfache HTTP-Endpoint-Tests werden in einen JMeter-Testplan umgewandelt und vom mitgelieferten Apache JMeter ausgeführt. Bevor Sie einen Test erstellen, sollten Sie die Test-Frameworks auf Sicherheitsaspekte, Lizenzinformationen und Third-party Patching-Optionen überprüfen.

Verkehrs-Shape-Modus

Wählen Sie aus, welche Seite die vom Test generierte Last steuert. Der von Ihnen gewählte Modus ändert die Felder, die die Konsole in Schritt 3: Traffic Shape anzeigt.

- Standard — Die Lösung steuert die Last. Sie legen die virtuellen Benutzer, die Anlaufzeit und die Haltedauer fest. Standard ist die Standardeinstellung, und es ist der einzige Modus, der einfache HTTP-Endpunkttests unterstützt.
- Nativ — Ihr Skript steuert das Laden. Die Lösung führt Ihr Skript unter der eigenen Befehlszeile des Test-Frameworks aus und legt nur die Anzahl der Aufgaben pro Region und eine Sicherheitsdauer fest.

Native erfordert ein hochgeladenes Skript, sodass Sie es nicht auswählen können, wenn Sie den Testtyp Simple HTTP Endpoint wählen. Die vollständigen Definitionen und Anleitungen dazu, welchen Modus Sie wählen sollten, finden Sie unter Traffic Shape Modes.

Konfiguration des HTTP-Endpunkts

Wenn „Simple HTTP Endpoint“ ausgewählt ist, generiert die Lösung anhand Ihrer Konfiguration einen JMeter-Testplan und führt ihn mit der mitgelieferten Apache JMeter-Binärdatei aus. Konfigurieren Sie diese Einstellungen:

HTTP-Endpunkt (erforderlich)

Geben Sie die vollständige URL des Endpunkts ein, den Sie testen möchten. Beispiel, `https://api.example.com/users`. Stellen Sie sicher, dass der Endpunkt von der AWS-Infrastruktur aus zugänglich ist.

HTTP-Methode (erforderlich)

Wählen Sie die HTTP-Methode für Ihre Anfragen aus. Der Standardwert ist `GET`. Zu den weiteren Optionen gehören `POST`,`PUT`,`DELETE`,`PATCH`,`HEAD`, und`OPTIONS`.

Header der Anfrage (optional)

Fügen Sie Ihren Anfragen benutzerdefinierte HTTP-Header hinzu. Allgemeine Beispiele sind:

- `Content-Type: application/json`
- `Authorization: Bearer <token>`
- `User-Agent: LoadTest/1.0`

Wählen Sie Header hinzufügen, um mehrere Header einzubeziehen.

Nutzlast des Körpers (optional)

Fügen Sie den Inhalt des Anforderungstexts für POST- oder PUT-Anfragen hinzu. Unterstützt JSON-, XML- oder Nur-Text-Formate. Beispiel: `{"userId": 123, "action": "test"}`.

Testen Sie Framework-Skripte

Wenn Sie JMeter, k6 oder Locust verwenden, laden Sie Ihre Testskriptdatei oder ein ZIP-Archiv hoch, das Ihr Testskript und unterstützende Dateien enthält.

Für JMeter können Sie benutzerdefinierte Plugins in einen `/plugins` Ordner innerhalb Ihres ZIP-Archivs aufnehmen.

Für Locust muss ein Testskript in einem ZIP-Archiv benannt werden. `locustfile.py` Um Python-Pakete von Drittanbietern zur Laufzeit in den Container zu installieren, fügen Sie eine `requirements.txt` Datei in das Archiv und optional ein `packages` Unterverzeichnis mit

Raddateien hinzu, um sie ohne Internetzugang zu installieren. Weitere Informationen finden Sie unter Locust-Tests.

> **Important**
>
> Im Standardmodus steuert die Lösung das Laden und überschreibt, was Ihr Skript deklariert. Ihr Testskript (JMeter, k6 oder Locust) kann Parallelität (virtuelle Benutzer), Transaktionsraten (TPS), Anlaufzeiten und andere Ladeparameter definieren. Die Lösung wendet stattdessen die Werte an, die Sie im Traffic Shape-Bildschirm angeben. Diese Konfiguration steuert die Anzahl der Aufgaben, die Parallelität (virtuelle Benutzer pro Aufgabe), die Dauer des Hochlaufs und die Haltedauer für die Testausführung.
> Im systemeigenen Modus steuert Ihr Skript die Last und die Lösung übergibt keine Ladeparameter an das Framework. Informationen zum Unterschied zwischen den beiden Modi finden Sie unter Traffic Shape Modes.

Sicherheitsdauer (systemeigener Modus)

Wenn Sie den systemeigenen Modus auswählen, zeigt die Konsole neben dem Skript-Upload ein Feld für die Sicherheitsdauer an. Die Standardeinstellung ist 4 Stunden und das Maximum ist 24 Stunden.

Im systemeigenen Modus beendet die Sicherheitsdauer einen Test, der länger als beabsichtigt ausgeführt wird, da Ihr Skript entscheidet, wann der Testlauf beendet wird. Es ist eine Wache, kein Zeitplan. Wenn der Test nach Ablauf der Dauer noch läuft, stoppt die Lösung das Test-Framework. Es behält die Ergebnisse für den Teil bei, der ausgeführt wurde, und zeichnet den Testlauf als abgeschlossen auf und nicht als fehlgeschlagen. Setzen Sie den Wert über die längste Ausführung, die das Skript voraussichtlich benötigen wird.

Schritt 3: Verkehrsform

Konfigurieren Sie, wie der Verkehr während Ihres Tests verteilt werden soll, einschließlich Unterstützung für mehrere Regionen.

Modi der Verkehrsform

Die Lösung bietet zwei Traffic-Shape-Modi: Standard und Nativ. Sie unterscheiden sich darin, welche Seite den Ladevorgang steuert: die Lösung oder das Skript, das Sie hochladen. Sie wählen

den Modus in Schritt 2: Szenariokonfiguration aus und er bestimmt, welche der folgenden Felder zutreffen.

> **Note**
>
> Der native Modus ist eine Vorschaufunktion in Version 4.3.0. Der Standardmodus ist die Standardeinstellung und das Verhalten, das die Lösung immer verwendet hat.

Standard

Im Standardmodus hat die Lösung die Kontrolle über die Last. Sie legen die Anzahl der Fargate-Aufgaben pro Region, die gleichzeitigen virtuellen Benutzer pro Aufgabe, eine Anlaufzeit und eine Haltedauer fest. Die Lösung führt Ihren Test über das Taurus Automation Framework aus, das diese Werte in die eigenen Laststeuerungen des zugrunde liegenden Frameworks übersetzt. Taurus hat Vorrang vor dem Ladevorgang, den Ihr Skript deklariert, sodass ein k6-Optionsblock, eine Locust- oder eine JMeter-Thread-Gruppe `LoadTestShape` neu geschrieben oder ignoriert wird. Die virtuellen Benutzer einer Region sind die Anzahl der Aufgaben multipliziert mit der Parallelität für jede Aufgabe, und die Form ist dieselbe, unabhängig davon, welches Framework Sie wählen. So liefen alle Tests vor Version 4.3.0 ab, sodass sich zuvor erstellte Szenarien weiterhin genauso verhalten wie zuvor und müssen nicht geändert werden.

Wählen Sie Standard, wenn die Ladeform außerhalb des Skripts liegt, das über die Konsole, die CLI oder einen Agenten festgelegt wurde. Standard ist der einzige Modus, in dem Sie eine exakte Anzahl virtueller Benutzer festlegen und die Ramp-up- und Hold-Form ändern können, ohne das Skript zu berühren. Nur Standard unterstützt den Typ Simple HTTP Endpoint, bei dem die Lösung den Testplan für Sie generiert. Typische Anwendungen: eine Kapazitätsprüfung mit 500 bis 5.000 virtuellen Benutzern, eine nächtliche Regression mit 1.000 Benutzern für zehn Minuten oder ein beliebiger Vergleich, für den eine identische Rampe erforderlich ist. Ihre Grenze liegt in der Aussagekraft. Alles, was Taurus nicht abbilden kann, ist hier nicht verfügbar, einschließlich mehrerer gewichteter Szenarien, Schwellenwerte pro Stufe und Ausführungspersonen mit ausführender Ankunftszeit.

Einheimisch

Im systemeigenen Modus hat Ihr Skript die Kontrolle über den Ladevorgang. Die Lösung führt die Datei aus, die Sie über die eigene Befehlszeile des Frameworks hochgeladen haben: `jmeter -n -tk6 run`,, oder`locust --headless`. Es gibt keine Lade-Flags weiter, sodass Ihr Skript die alleinige Autorität für den generierten Datenverkehr hat und die Lösung ihn niemals neu schreibt.

Zwei Kontrollen bleiben bestehen: die Anzahl der Fargate-Aufgaben, die pro Region gestartet werden sollen, und die erforderliche Sicherheitsdauer von bis zu 24 Stunden. Die Sicherheitsdauer dient als Schutz vor einem Skript, das niemals beendet wird, und ist kein Zeitplan. Wenn ein Test nach Ablauf der Dauer noch läuft, stoppt die Lösung das Framework, sammelt die Ergebnisse für den Teil, der ausgeführt wurde, und zeichnet den Testlauf als abgeschlossen auf und nicht als fehlgeschlagen. Jede Aufgabe wird als unabhängiger Framework-Prozess ohne Koordination zwischen den Aufgaben ausgeführt, sodass eine Region für jede Aufgabe eine vollständige Kopie der in Ihrem Skript deklarierten Last generiert. Beispiel: Ein k6-Skript mit 200 virtuellen Benutzern, das auf fünf Aufgaben ausgeführt wird, platziert 1.000 virtuelle Benutzer auf dem Ziel. Die Anzahl der Aufgaben ist daher die einzige Laststeuerung, die Native Ihnen bietet, und sie verschiebt ganze Vielfache dessen, was das Skript deklariert. Wenn Sie die Rampe, die Haltezeit oder die Anzahl der virtuellen Benutzer selbst ändern, müssen Sie das Skript bearbeiten.

Wählen Sie Nativ, wenn Sie ein Skript genau so ausführen möchten, wie es geschrieben wurde. Ein Skript, das bereits lokal oder in CI ausgeführt wird, wird unverändert auf der Lösung ausgeführt. Dies ist der Hauptgrund, sich für Native zu entscheiden. Native behält alles bei, was das Framework ausdrücken kann: k6-Szenarien, Stufen und Schwellenwerte; `LoadTestShape` Locust-Klassen und gewichtete Aufgabensätze; JMeter-Timer und Thread-Gruppen. Wählen Sie es aus, wenn die Lastform Teil der Testaussage ist und es wichtiger ist, sie originalgetreu zu reproduzieren, als sie von außen zu steuern. Typische Anwendungen: Wiederverwendung eines K6-Skripts aus einer Pipeline, ohne es neu zu schreiben, ein Profil, bei dem es nach oben geht, oder gewichtete Szenarien, die eine einzelne Parallelitätszahl nicht ausdrücken kann. Zwei Einschränkungen ergeben sich aus der Ausführung des Frameworks, wie es geschrieben wurde. Native erfordert ein hochgeladenes Skript, daher ist Simple HTTP Endpoint nicht verfügbar. Locust-Skripte dürfen nicht gesetzt werden`processes`, da die Lösung Anfragen nur zählt, wenn Locust als ein einziger Prozess ausgeführt wird.

Einen Modus wählen

Wenn du brauchst	Klicken Sie auf
Eine genaue Anzahl virtueller Benutzer, die von außerhalb des Skripts festgelegt wurde	Standard
Um die Hochlauf- oder Haltezeit zu ändern, ohne das Skript zu bearbeiten	Standard
Eine einzelne URL ohne jegliches Skript	Standard

Wenn du brauchst	Klicken Sie auf
Unabhängig vom Framework dieselbe Ladeform	Standard
Um ein CI oder ein lokales Skript unverändert wiederzuv erwenden	Nativ
Die eigenen Stufen, Schwellenwerte oder die Form des Skripts wurden berücksichtigt	Nativ
k6-Szenarien, Schwellenwerte oder Ausführungsprogramme mit hoher Ankunftsrate	Nativ
Ein `LoadTestShape` Locust- oder gewichteter Aufgabensatz	Nativ
JMeter-Timer und Thread-Gruppen laufen genau so, wie sie geschrieben wurden	Nativ
Um die Last nur auf ein Vielfaches der eigenen Last des Skripts zu skalieren	Nativ

Multi-Region Konfiguration des Datenverkehrs

Wählen Sie eine oder mehrere AWS-Regionen aus, um Ihren Belastungstest geografisch zu verteilen. Konfigurieren Sie für jede ausgewählte Region:

Anzahl der Aufgaben

Die Anzahl der Container (Aufgaben), die im Fargate-Cluster für das Testszenario gestartet werden. Zusätzliche Aufgaben werden nicht mehr erstellt, sobald das Konto das Limit „Fargate-Ressource wurde erreicht“ erreicht hat. Die Anzahl der Aufgaben gilt für beide Traffic Shape-Modi. Im nativen Modus ist dies die einzige Laststeuerung, die die Konsole bietet, da jede Aufgabe eine vollständige Kopie Ihres Skripts ausführt.

Concurrency (Nebenläufigkeit)

Die Anzahl gleichzeitiger virtueller Benutzer, die pro Aufgabe generiert werden. Das empfohlene Limit basiert auf den Standardeinstellungen von 2 vCPUs pro Aufgabe. Die Parallelität wird durch CPU- und Speicherressourcen begrenzt. Dieses Feld gilt nur für den Standardmodus. Im nativen

Modus zeigt die Konsole für jede Region den schreibgeschützten Wert „Definiert durch Skript“ an, da Ihr Skript die Anzahl der virtuellen Benutzer selbst festlegt.

Ermitteln Sie die Anzahl der Benutzer

Die Anzahl der Benutzer, die ein Container für einen Test unterstützen kann, kann bestimmt werden, indem die Anzahl der Benutzer schrittweise erhöht und die Leistung in Amazon überwacht wird CloudWatch. Sobald Sie feststellen, dass die CPU- und Speicherleistung an ihre Grenzen stößt, haben Sie die maximale Anzahl von Benutzern erreicht, die ein Container für diesen Test in seiner Standardkonfiguration unterstützen kann (2 vCPU und 4 GB Arbeitsspeicher).

Durch diese Kalibrierung wird der Parallelitätswert festgelegt, sodass er für den Standardmodus gilt. Die von ihr festgelegten Container-Grenzwerte gelten auch für den systemeigenen Modus. Dort erhöhen oder verringern Sie die von Ihrem Skript deklarierte Last, anstatt das Feld Concurrency festzulegen.

Der Kalibrierungsprozess

Anhand des folgenden Beispiels können Sie mit der Festlegung der Grenzwerte für gleichzeitige Benutzer für Ihren Test beginnen:

1. Erstellen Sie einen Test mit nicht mehr als 200 Benutzern.
2. Überwachen Sie während der Ausführung des Tests die CPU und den Speicher mithilfe der CloudWatch Konsole:
   a. Wählen Sie im Navigationsbereich unter Container Insights die Option Performance Monitoring aus.
   b. Wählen Sie auf der Seite zur Leistungsüberwachung im linken Dropdownmenü die Option ECS-Cluster aus.
   c. Wählen Sie im rechten Drop-down-Menü Ihren Amazon Elastic Container Service (Amazon ECS) -Cluster aus.
3. Achten Sie bei der Überwachung auf die CPU und den Arbeitsspeicher. Wenn die CPU nicht über 75% oder der Arbeitsspeicher nicht über 85% liegt (einmalige Spitzenwerte ignorieren), können Sie einen weiteren Test mit einer höheren Benutzerzahl durchführen.

Wiederholen Sie die Schritte 1 bis 3, wenn der Test die Ressourcengrenzen nicht überschritten hat. Optional können Sie die Container-Ressourcen erhöhen, um eine höhere Anzahl gleichzeitiger

Benutzer zu ermöglichen. Dies führt jedoch zu höheren Kosten. Einzelheiten finden Sie im Entwicklerhandbuch.

> **Note**
>
> Um genaue Ergebnisse zu erzielen, sollten Sie bei der Festlegung der Grenzwerte für gleichzeitige Benutzer jeweils nur einen Test durchführen. Alle Tests verwenden denselben Cluster, und CloudWatch Container Insights aggregiert die Leistungsdaten auf der Grundlage des Clusters. Dies führt dazu, dass beide Tests gleichzeitig an CloudWatch Container Insights gemeldet werden, was zu ungenauen Kennzahlen zur Ressourcennutzung für einen einzelnen Test führt.

Weitere Informationen zur Kalibrierung von Benutzern pro Engine finden Sie in der Dokumentation unter Kalibrierung eines Taurus-Tests. BlazeMeter

> **Note**
>
> Die Lösung zeigt Informationen zur verfügbaren Kapazität für jede Region an und hilft Ihnen so, Ihre Testkonfiguration innerhalb der verfügbaren Grenzen zu planen.

Tabelle der verfügbaren Aufgaben

In der Tabelle der verfügbaren Aufgaben wird die Ressourcenverfügbarkeit für jede ausgewählte Region angezeigt:

- Region — Der Name der AWS-Region.
- vCPUs pro Aufgabe — Die Anzahl der virtuellen CPUs, die jeder Aufgabe zugewiesen sind (Standard: 2).
- DLT-Aufgabenlimit — Die maximale Anzahl von Aufgaben, die auf der Grundlage des Fargate-On-Demand-vCPU-Kontingents Ihres Kontos erstellt werden können. Neue Konten haben in der Regel ein niedrigeres Kontingent. Überprüfen Sie Ihr aktuelles Limit in der Service-Kontingents-Konsole und fordern Sie bei Bedarf eine Erhöhung an.
- Verfügbare DLT-Aufgaben — Die aktuelle Anzahl der in der Region verfügbaren Aufgaben, berechnet als Ihr DLT-Aufgabenlimit abzüglich der vCPUs, die bereits durch die Ausführung von Fargate-Aufgaben verwendet werden.

Informationen zur Erhöhung der Anzahl verfügbarer Aufgaben oder vCPUs pro Aufgabe finden Sie im Entwicklerhandbuch.

Dauer des Tests

Definieren Sie, wie lange Ihr Auslastungstest laufen soll. Die Konsole zeigt diesen Abschnitt nur im Standardmodus an. Im systemeigenen Modus bestimmt Ihr Skript, wie lange der Test ausgeführt wird. Dies ist durch die Sicherheitsdauer begrenzt, die Sie in Schritt 2: Szenariokonfiguration festgelegt haben.

Hochfahren

Die Zeit bis zum Erreichen der Zielparallelität. Die Last steigt in diesem Zeitraum allmählich von 0 auf den konfigurierten Parallelitätsgrad.

Halten Sie für

Die Dauer, für die die Ziellast aufrechterhalten werden soll. Der Test wird für diesen Zeitraum mit voller Parallelität fortgesetzt.

Schritt 4: Überprüfen und Erstellen

Überprüfen Sie alle Ihre Konfigurationen, bevor Sie das Testszenario erstellen. Überprüfen:

- Allgemeine Einstellungen (Name, Beschreibung, Zeitplan).
- Szenariokonfiguration (Testtyp, Endpunkt oder Skript).
- Verkehrsform (Modus, Aufgaben, Benutzer, Dauer, Regionen).

Wählen Sie nach der Überprüfung Create aus, um Ihr Testszenario zu speichern.

Testszenarien verwalten

Nachdem Sie ein Testszenario erstellt haben, können Sie:

- Bearbeiten — Ändern Sie die Testkonfiguration. Häufige Anwendungsfälle umfassen:
  - Verfeinerung der Verkehrsform, um die gewünschte Transaktionsrate zu erreichen.
- Kopieren — Duplizieren Sie ein vorhandenes Testszenario, um Varianten zu erstellen. Häufige Anwendungsfälle umfassen:
  - Endpunkte aktualisieren oder headers/body Parameter hinzufügen.

  - Hinzufügen oder Ändern von Testskripten.
- Löschen — Entfernen Sie Testszenarien, die Sie nicht mehr benötigen.

Führen Sie ein Testszenario aus

Nachdem Sie ein Testszenario erstellt haben, können Sie es sofort ausführen oder so planen, dass es zu einem bestimmten Zeitpunkt in der future ausgeführt wird. Wenn Sie zu einem laufenden Test navigieren, zeigt die Konsole die Registerkarte Szenariodetails mit Aufgabenstatus und Messdaten in Echtzeit an.

Ansicht der Szenariodetails

Auf der Registerkarte Szenariodetails werden wichtige Informationen zu Ihrem Test angezeigt. In der Tabelle mit dem Aufgabenstatus werden Echtzeitinformationen für jede Region angezeigt.

Tabelle mit dem Aufgabenstatus

In der Tabelle mit dem Aufgabenstatus werden Echtzeitinformationen für jede Region angezeigt:

- Region — Die AWS-Region, in der Aufgaben ausgeführt werden
- Anzahl der Aufgaben — Die Gesamtzahl der für die Region konfigurierten Aufgaben
- Parallelität — Die Anzahl der virtuellen Benutzer pro Aufgabe
- Wird ausgeführt — Anzahl der Aufgaben, die den Test derzeit ausführen
- Ausstehend — Anzahl der Aufgaben, die darauf warten, gestartet zu werden
- Bereitstellung — Anzahl der Aufgaben, die bereitgestellt werden

Arbeitsablauf bei der Testausführung

Wenn ein Test gestartet wird, findet der folgende Workflow statt:

1. Aufgabenbereitstellung — Die Lösung stellt Container (Aufgaben) in den angegebenen AWS-Regionen bereit. Aufgaben werden in der Spalte „Bereitstellung“ angezeigt.
2. Start der Aufgabe — Die Lösung stellt weiterhin Aufgaben bereit, bis die Zielanzahl der Aufgaben in jeder Region erreicht ist. Aufgaben werden von „Bereitstellung“ zu „Ausstehend“ zu „Wird ausgeführt“ verschoben.

3. Generierung von Traffic — Nachdem die Lösung alle Aufgaben in einer Region bereitgestellt hat, beginnt sie, Traffic an Ihren Zielendpunkt zu senden.
4. Testausführung — Der Test wird für die konfigurierte Dauer (Hochlauf + Haltezeit) ausgeführt.
5. Analyse der Ergebnisse — Wenn der Test beendet ist, aggregiert und verarbeitet ein Hintergrund-Analysejob die Ergebnisse aus allen Regionen.

Status der Testläufe

Testläufe können den folgenden Status haben:

- Geplant — Der Test soll in der future ausgeführt werden.
- Wird ausgeführt — Der Test ist derzeit im Gange.
- Abgebrochen — Ein Benutzer hat einen laufenden Testlauf abgebrochen.
- Fehlerhaft — Beim Testlauf ist ein Fehler aufgetreten.
- Abgeschlossen — Der Testlauf wurde erfolgreich abgeschlossen und die Ergebnisse liegen vor.

Überwachung mit Live-Daten

Wenn Sie bei der Erstellung des Testszenarios Live-Daten aktiviert haben, können Sie Echtzeitmetriken anzeigen, während der Test ausgeführt wird. Im Abschnitt Echtzeit-Metriken werden vier Diagramme angezeigt, die im Verlauf des Tests kontinuierlich aktualisiert werden. Die Daten werden in Intervallen von einer Sekunde aggregiert.

Beschreibungen der Grafiken

Durchschnittliche Antwortzeit

Zeigt die durchschnittliche Antwortzeit in Sekunden für Anfragen an, die von jeder Region verarbeitet wurden. Y-axis Zeigt die Antwortzeit in Sekunden und die X-axis Tageszeit an. Jede Region wird in der Legende durch eine andere Farbe dargestellt.

Virtuelle Benutzer

Zeigt die Anzahl der gleichzeitigen virtuellen Benutzer an, die in jeder Region aktiv Last erzeugen. Das Diagramm zeigt, wie virtuelle Benutzer während des Tests an Wert gewinnen und das angestrebte Parallelitätsniveau beibehalten wird.

Erfolgreiche Anfragen

Zeigt die Gesamtanzahl erfolgreicher Anfragen im Laufe der Zeit für jede Region an. Das Diagramm zeigt die Geschwindigkeit, mit der erfolgreiche Anfragen bearbeitet werden.

Fehlgeschlagene Anfragen

Zeigt die Gesamtzahl der fehlgeschlagenen Anfragen im Laufe der Zeit für jede Region an. Eine niedrige Anzahl oder Null weist auf eine fehlerfreie Testausführung hin.

Multi-Region Visualisierung

Wenn Tests in mehreren Regionen ausgeführt werden, zeigt jedes Diagramm Daten für alle Regionen gleichzeitig an. Die Legende am unteren Rand jedes Diagramms gibt an, welche Farbe die einzelnen Regionen repräsentiert (z. B. us-west-2 und us-east-1).

Technische Umsetzung

Die CloudWatch Protokollgruppe für die Fargate-Aufgaben enthält einen Abonnementfilter, der Testergebnisse erfasst. Wenn das Muster erkannt wird, strukturiert eine Lambda-Funktion die Daten und veröffentlicht sie in einem AWS IoT Core Core-Thema. Die Webkonsole abonniert dieses Thema und zeigt die Metriken in Echtzeit an.

> **Note**
>
> Live-Daten sind kurzlebig und nur verfügbar, während der Test läuft. Die Webkonsole speichert maximal 5.000 Datenpunkte. Danach werden die ältesten Daten durch die neuesten ersetzt. Wenn die Seite aktualisiert wird, sind die Grafiken leer und beginnen mit dem nächsten verfügbaren Datenpunkt. Sobald ein Test abgeschlossen ist, speichert die Lösung die Ergebnisdaten in DynamoDB und Amazon S3. Wenn noch keine Daten verfügbar sind, wird in den Diagrammen „Es sind keine Daten verfügbar“ angezeigt.

Einen Test stornieren

Sie können einen laufenden Test über die Webkonsole abbrechen. Wenn Sie einen Test abbrechen, läuft der folgende Arbeitsablauf ab:

1. Die Stornierungsanfrage wird an die `microservices` API gesendet
2. Die `microservices` API ruft die `task-canceler` Lambda-Funktion auf, die alle aktuell gestarteten Aufgaben stoppt
3. Wenn die `task-runner` Lambda-Funktion nach dem ersten Abbruchaufruf weiter ausgeführt wird, werden Aufgaben möglicherweise weiterhin kurz gestartet.
4. Sobald die `task-runner` Lambda-Funktion abgeschlossen ist, fährt AWS Step Functions mit dem `Cancel Test` Schritt fort, in dem die `task-canceler` Lambda-Funktion erneut ausgeführt wird, um alle verbleibenden Aufgaben zu beenden

> **Note**
>
> Bei abgebrochenen Tests dauert es einige Zeit, bis der Shutdown-Vorgang abgeschlossen ist, da die Lösung alle Container beendet. Der Teststatus ändert sich auf „Storniert“, sobald alle Ressourcen bereinigt sind.

Erkunden Sie die Testergebnisse

Sobald der Parsing-Job abgeschlossen ist, stehen die Testergebnisse zur Analyse zur Verfügung. Die Lösung bietet umfassende Metriken und Tools, die Ihnen helfen, die Leistung Ihrer Anwendung unter Last zu verstehen.

Übersichtsmetriken für den Testlauf

Wenn ein Test abgeschlossen ist, generiert die Lösung eine Zusammenfassung, die die folgenden Kennzahlen enthält:

- Durchschnittliche Antwortzeit — Die durchschnittliche Antwortzeit in Sekunden für alle durch den Test generierten Anfragen.
- Durchschnittliche Latenz — Die durchschnittliche Latenz in Sekunden für alle durch den Test generierten Anfragen.
- Durchschnittliche Verbindungszeit — Die durchschnittliche Zeit in Sekunden, die für alle Anfragen benötigt wird, um eine Verbindung zum Host herzustellen.
- Durchschnittliche Bandbreite — Die durchschnittliche Bandbreite für alle durch den Test generierten Anfragen.
- Gesamtzahl — Die Gesamtzahl der Anfragen.
- Anzahl erfolgreicher Anfragen — Die Gesamtzahl der erfolgreichen Anfragen.
- Anzahl der Fehler — Die Gesamtzahl der Fehler.
- Anfragen pro Sekunde — Die durchschnittlichen Anfragen pro Sekunde für alle vom Test generierten Anfragen.
- Perzentile — Perzentile für die Antwortzeit, einschließlich p50 (Median), p90, p95 und p99, die die Verteilung der Antwortzeiten auf alle Anfragen zeigen.

Tabelle mit Testläufen

In der Tabelle mit den Testläufen werden alle historischen Testläufe für ein Szenario angezeigt. Sie können:

- Übersichtsmetriken für jeden Testlauf anzeigen.
- Legen Sie einen Basistestlauf für den Leistungsvergleich fest.
- Laden Sie die Tabelle als CSV-Datei herunter.
- Wechseln Sie zwischen den Spalten, um Ihre Ansicht anzupassen.
- Wählen Sie einen Testlauf aus, um detaillierte Ergebnisse anzuzeigen.

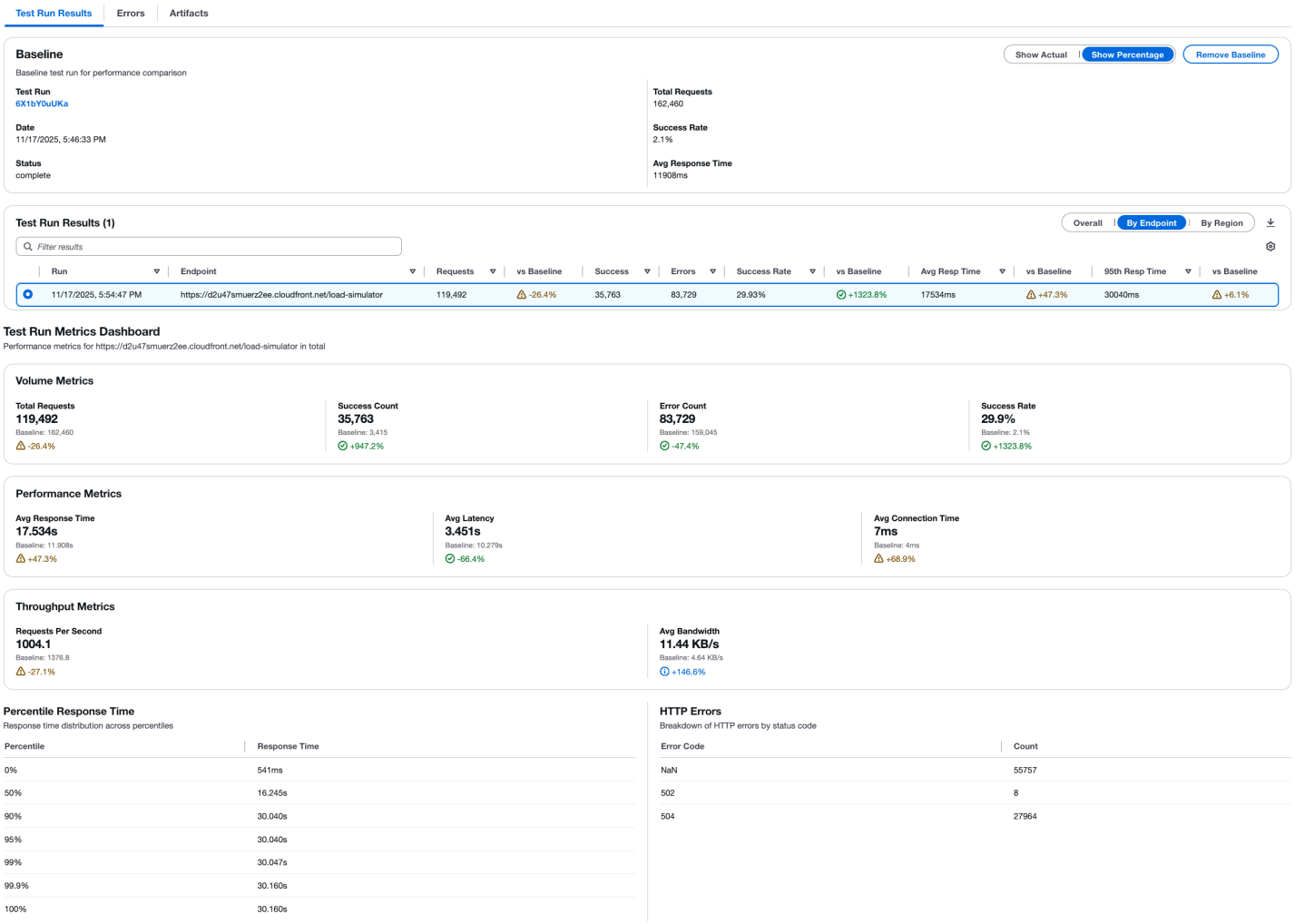
Vergleich der Ausgangswerte

Sie können einen Testlauf als Ausgangswert festlegen, um future Testläufe damit zu vergleichen. Wenn ein Basiswert festgelegt ist:

- Die Tabelle mit den Testläufen zeigt für jede Metrik prozentuale Unterschiede (+/%) im Vergleich zum Ausgangswert.
- Der Basisindikator hilft Ihnen dabei, Leistungsverbesserungen oder Regressionen schnell zu erkennen.
- Sie können den Basiswert jederzeit ändern oder löschen.

Detaillierte Testergebnisse

Wenn Sie einen Testlauf auswählen, wird die detaillierte Ergebnisansicht mit drei Registerkarten geöffnet: Testlaufergebnisse, Fehler und Artefakte.



Basisinformationen

Wenn ein Basistestlauf festgelegt ist, wird dieser oben auf der Seite angezeigt. Sie können „Aktuellen Wert anzeigen“, „Prozentsatz anzeigen“ oder „Basiswert entfernen“ wählen, um zu steuern, wie Basisvergleiche angezeigt werden.

Tabelle mit den Ergebnissen des Testlaufs

Die Ergebnistabelle enthält detaillierte Messwerte mit den folgenden Funktionen:

Dimensionsansichten

Wechseln Sie mit den Bemaßungsschaltflächen zwischen drei Ansichten:

- Insgesamt — Aggregierte Ergebnisse für alle Endpunkte und Regionen
- Nach Endpunkten — Die Ergebnisse sind nach einzelnen Endpunkten aufgeschlüsselt

- Nach Regionen — Ergebnisse aufgeschlüsselt nach AWS-Regionen

Aktionsschaltflächen

- Aktuelle Werte anzeigen — Zeigt die tatsächlichen Metrikwerte an
- Prozentsatz anzeigen — Zeigt prozentuale Abweichungen vom Ausgangswert an
- Basisplan entfernen — Löscht den Basisvergleich

Datenexport und Anpassung

- Laden Sie die Ergebnistabelle als CSV-Datei herunter
- Wechseln Sie zwischen den Spalten, um Ihre Ansicht anzupassen
- Filtern und sortieren Sie Daten, um sich auf bestimmte Kennzahlen zu konzentrieren.

Registerkarte „Fehler“

Die Registerkarte Fehler bietet eine detaillierte Fehleranalyse:

- Zeigt die Anzahl der Fehler nach Typ an.
- Lassen Sie sich die Fehler nach Gesamttest oder Endpunkt aggregiert anzeigen.
- Identifizieren Sie Muster bei fehlgeschlagenen Anfragen.
- Beheben Sie Probleme mit bestimmten Endpunkten oder Regionen.

Registerkarte „Artefakte“

Über die Registerkarte Artefakte können Sie auf alle Dateien zugreifen, die während des Testlaufs generiert wurden:

- Einzelne Artefakte (Protokolle, Ergebnisdateien) anzeigen.
- Laden Sie bestimmte Artefakte für die Offline-Analyse herunter.
- Laden Sie alle Testlauf-Artefakte als ein einziges Archiv herunter.

Struktur der S3-Ergebnisse

In Version 4.0 wurde die S3-Ergebnisstruktur geändert, um die Organisation zu verbessern:

- Neue Struktur -`scenario-id/test-run-id/results-files`.

- Legacy-Struktur — Tests, die vor Version 4.0 ausgeführt wurden, zeigen alle Ergebnisdateien auf Szenario-ID-Ebene.

> **Note**
>
> Die Testergebnisse werden in der Konsole angezeigt. Sie können auch direkt im Amazon S3 S3-Bucket unter dem `Results` Ordner auf die Rohtestergebnisse zugreifen. Weitere Informationen zu den Taurus-Testergebnissen finden Sie unter Generieren von Testberichten im Taurus-Benutzerhandbuch.

Überwachen Sie mit Logs Insights CloudWatch

Diese Lösung stellt vier in CloudWatch Logs Insights gespeicherte Abfragen bereit, die in der CloudWatch Logs Insights-Konsole unter Gespeicherte Abfragen angezeigt werden. Diese Abfragen bieten Einblick in Testorchestrierung, Fehler, Aufgabenfehler und Orphan-Cleanup, ohne dass Sie benutzerdefinierte Abfragen schreiben müssen.

Jeder Abfragename enthält den Stacknamen und die Region zur Identifizierung in mehreren Bereitstellungen. Beispiel: `DLT - Test Timeline [my-stack us-east-1]`.

Verwenden von gespeicherten Abfragen

So führen Sie eine gespeicherte Abfrage aus:

1. Öffnen Sie die CloudWatch Logs Insights-Konsole.
2. Erweitern Sie im linken Bereich den Bereich Gespeicherte Abfragen.
3. Wählen Sie eine Abfrage mit dem `DLT -` Präfix aus.
4. Wenn die Abfrage einen Platzhalterwert wie enthält`REPLACE_WITH_TEST_RUN_ID`, ersetzen Sie ihn durch die tatsächliche Testlauf-ID aus Ihren Testergebnissen.
5. Wählen Sie Abfrage ausführen.

DLT — Zeitplan für den Test

Zeigt den gesamten Lebenszyklus für einen einzigen Testlauf für alle Orchestrierungs-Lambda-Funktionen. Verwenden Sie diese Abfrage, um die Reihenfolge der Ereignisse von der Erstellung bis zum Abschluss des Tests nachzuverfolgen.

Attribut	Wert
Felder	`@timestamp` , `logEvent`, `message`, `region`, `error`
Filter	`testRunId = "REPLACE_WITH_TEST_RUN_ID"`
Sortierung	`@timestamp asc`
Limit	500

`REPLACE_WITH_TEST_RUN_ID`Ersetzen Sie es durch die Testlauf-ID, die Sie untersuchen möchten.

DLT — Testfehler

Zeigt alle ERROR-level Einträge für einen einzelnen Testlauf mit Lambda-Funktionen und ECS-Aufgaben an. Verwenden Sie diese Abfrage, um die Hauptursache zu ermitteln, wenn ein Test fehlschlägt oder zu unerwarteten Ergebnissen führt.

Attribut	Wert
Felder	`@timestamp` , `logEvent`, `message`, `region`, `taskId`, `error`
Filter	`testRunId = "REPLACE_WITH_TEST_RUN_ID" and level = "ERROR"`
Sortierung	`@timestamp asc`

`REPLACE_WITH_TEST_RUN_ID`Ersetzen Sie es durch die Testlauf-ID, die Sie untersuchen möchten.

DLT — Aufgabenfehler

Zeigt einzelne Fehler bei ECS-Aufgaben mit Stoppcodes und Fehlerklassifizierung an. Verwenden Sie diese Abfrage, um zu verstehen, warum bestimmte Fargate-Aufgaben während eines Tests gestoppt wurden.

Attribut	Wert
Felder	`@timestamp` , `testId`, `testRunId` , `region`, `taskArn`, `stopCode`, `exitCode`, `stopCategory` , `stoppedReason`
Filter	`logEvent = "TASK_FAILURE_DETECTED"`
Sortierung	`@timestamp desc`
Limit	50

Für diese Abfrage ist keine Testlauf-ID erforderlich. Sie zeigt alle kürzlich ausgefallenen Aufgaben in allen Testläufen an.

DLT — Orphan Cleanup

Zeigt den Verlauf der Erkennung verwaister ECS-Dienste an. Verwenden Sie diese Abfrage, um zu überprüfen, ob im Rahmen des stündlichen Säuberungsvorgangs für verwaiste Dienste die Suche und Entfernung aufgegebener Dienste erfolgt.

Attribut	Wert
Felder	`@timestamp` , `logEvent`, `message`, `region`, `cluster`, `orphanCount` , `orphanTestIds`
Filter	`logEvent = "ORPHAN_DETECTED"`
Sortierung	`@timestamp desc`
Limit	50

CloudWatch Alarme

Bei dieser Lösung werden zwei CloudWatch Alarme eingesetzt, die die Betriebsbedingungen überwachen, die besondere Aufmerksamkeit erfordern. Standardmäßig sind für diese Alarme keine Benachrichtigungsaktionen konfiguriert. Wir empfehlen, für jeden Alarm ein Amazon SNS SNS-Thema zu abonnieren, damit die Betreiber sofort benachrichtigt werden, wenn Probleme auftreten.

Abonnieren Sie Alarmbenachrichtigungen

Um Benachrichtigungen zu erhalten, wenn ein Alarm ausgelöst wird:

1. Öffnen Sie die CloudWatch Alarm-Konsole.
2. Suchen Sie nach Alarmen, denen Ihr Stack-Name vorangestellt ist (z. B.`my-stack-OrphanCleanupFailure`).
3. Wählen Sie den Alarm aus und wählen Sie Bearbeiten.
4. Wählen Sie unter Benachrichtigung die Option Benachrichtigung hinzufügen aus.
5. Wählen oder erstellen Sie ein SNS-Thema mit Ihren bevorzugten Benachrichtigungsendpunkten (E-Mail, SMS oder Lambda).
6. Wählen Sie Update Alarm (Alarm bearbeiten) aus.

Wiederholen Sie den Vorgang für jeden Alarm.

OrphanCleanupFailure

Attribut	Wert
Name des Alarms	`{StackName}-OrphanCleanupFailure`
Metrik	`OrphanCleanupFailures` im `distributed-load-testing` Namespace
Threshold	>= 1 Fehler innerhalb von 5 Minuten
Behandeln Sie fehlende Daten	Verstoß

Was dieser Alarm überwacht: Die Lösung nutzt drei Schutzebenen, um außer Kontrolle geratene ECS-Services zu verhindern:

- Ebene 1: Automatisierte Fehlerbehandlung — Der Workflow zur Testorchestrierung umfasst die Fehlerbehandlung bei jedem Schritt. Wenn während der Bereitstellung, Stabilisierung oder Ausführung etwas fehlschlägt, löst der Workflow automatisch eine Bereinigung aus, um die ECS-Services zu leeren und zu löschen.

- Schicht 2: Erkennung von Ausführungsfehlern — Wenn der Orchestrierungs-Workflow selbst unerwartet beendet wird (z. B. aufgrund eines Timeouts oder eines internen Fehlers, der die normale Fehlerbehandlung umgeht), erkennt eine EventBridge Regel den Fehler und löst unabhängig eine Bereinigung für jede am Test beteiligte Region aus.
- Schicht 3: Stündliche Orphan-Bereinigung — Ein geplanter Prozess wird stündlich ausgeführt, sucht nach ECS-Services, die keinem aktiven Test zugeordnet sind, und erzwingt deren Löschung. Dies ist das letzte Sicherheitsnetz — wenn sowohl Layer 1 als auch Layer 2 ausfallen, werden durchgesickerte Dienste trotzdem innerhalb einer Stunde entfernt. Wenn der Orphan-Bereinigungsprozess selbst fehlschlägt, wird dieser Alarm ausgelöst.

Warum das wichtig ist: Verwaiste ECS Fargate-Dienste werden weiterhin ausgeführt und es fallen Gebühren an, ohne dass sie in der DLT-Konsole angezeigt werden. Ohne ein Benachrichtigungsabonnement werden die Betreiber das Problem erst entdecken, wenn unerwartete Kosten auf der Rechnung erscheinen.

Empfohlene Reaktion: Wenn dieser Alarm ausgelöst wird, navigieren Sie zur Amazon ECS-Konsole, identifizieren Sie Dienste im DLT-Cluster, die keinem laufenden Test entsprechen, und löschen Sie sie manuell.

MetricFilterCount

Attribut	Wert
Name des Alarms	`{StackName}-MetricFilterCount-Alarm`
Metrik	`MetricFilterCount` im Namespace `distributed-load-testing`
Threshold	>= 90
Behandeln Sie fehlende Daten	Kein Verstoß

Was dieser Alarm überwacht: Die Lösung erstellt dynamisch CloudWatch Metrikfilter in der ECS-Protokollgruppe, um Live-Metriken während der Testausführung zu unterstützen. AWS begrenzt jede Protokollgruppe auf 100 Metrikfilter. Dieser Alarm wird ausgelöst, wenn die Nutzung 90% dieses Grenzwerts erreicht.

Warum das wichtig ist: Wenn das Limit erreicht wird, schlagen neue Lasttestläufe fehl.

Empfohlene Antwort: Löschen Sie Testszenarien, die nicht mehr benötigt werden. Wenn ein Testszenario gelöscht wird, entfernt die Lösung die zugehörigen Metrikfilter und gibt Kapazität für neue Tests frei.

MCP-Serverintegration

Wenn Sie die optionale MCP-Serverkomponente während der Lösungsbereitstellung bereitgestellt haben, können Sie die Distributed Load Testing-Lösung in KI-Entwicklungstools integrieren, die das Model Context Protocol unterstützen. Der MCP-Server bietet programmatischen Zugriff zum Abrufen, Verwalten und Analysieren von Lasttests mithilfe von KI-Assistenten.

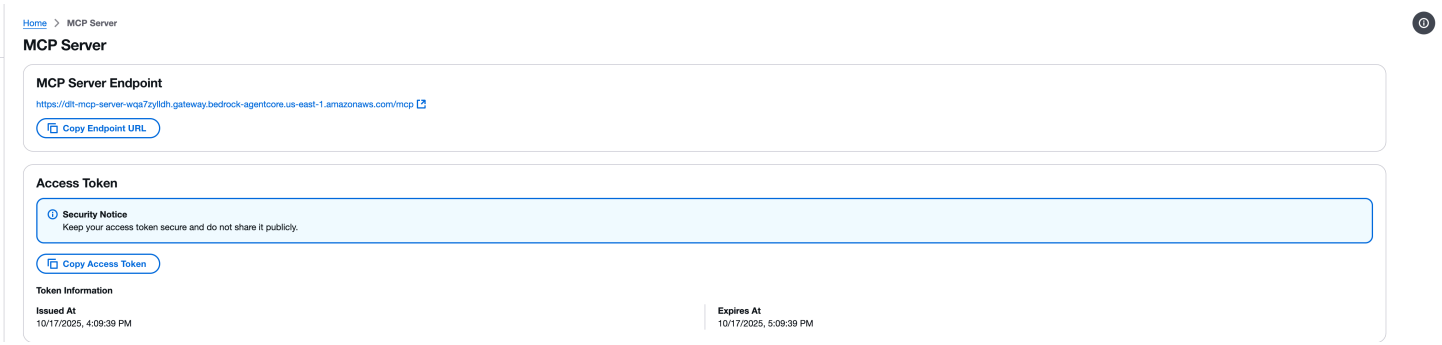
Sie können sich mit dem Client Ihrer Wahl (Kiro CLI, Claude usw.) mit jeweils leicht unterschiedlichen Konfigurationsanweisungen mit dem DLT MCP-Server verbinden. Dieser Abschnitt enthält Anweisungen zur Einrichtung von MCP Inspector, Kiro CLI, Cline und Amazon Quick.

Schritt 1: Holen Sie sich den MCP-Endpunkt und das Zugriffstoken

Bevor Sie einen MCP-Client konfigurieren, müssen Sie Ihren MCP-Server-Endpunkt und Ihr Zugriffstoken von der DLT-Webkonsole abrufen.

1. Navigieren Sie in der Distributed Load Testing-Webkonsole zur MCP-Serverseite.
2. Suchen Sie den Abschnitt MCP Server Endpoint.
3. Kopieren Sie die Endpunkt-URL mithilfe der Schaltfläche Endpunkt-URL kopieren. Die Endpunkt-URL folgt dem Format: `https://{gateway-id}.gateway.bedrock-agentcore.{region}.amazonaws.com/mcp`
4. Suchen Sie den Abschnitt Access Token.
5. Kopieren Sie das Zugriffstoken mit der Schaltfläche Zugriffstoken kopieren.

> **Important**
>
> Bewahren Sie Ihr Zugriffstoken sicher auf. Teilen Sie es nicht öffentlich. Standardmäßig bietet das Token über die MCP-Schnittstelle schreibgeschützten Zugriff auf Ihre Distributed Load Testing-Lösung. Wenn der MCP-Server im ReadWrite Zugriffsmodus bereitgestellt wird, ermöglicht das Token auch Erstellungs-, Aktualisierungs- und Löschvorgänge. Weitere Informationen finden Sie in der MCP-Tools-Spezifikation im Developer Guide.



Schritt 2: Testen Sie mit MCP Inspector

Das Model Context Protocol bietet MCP Inspector, ein Tool zum direkten Herstellen einer Verbindung zu MCP-Servern und zum Aufrufen von Tools. Dies bietet eine praktische Benutzeroberfläche und Beispielnetzwerkanforderungen zum Testen Ihrer MCP-Serververbindung vor der Konfiguration von AI-Clients.

> **Note**
>
> MCP Inspector benötigt Version 0.17 oder höher. Alle Anfragen können auch direkt mit JSON RPC gestellt werden, MCP Inspector bietet jedoch eine benutzerfreundlichere Oberfläche.

Installieren und starten Sie MCP Inspector

1. Installieren Sie npm, falls erforderlich.
2. Führen Sie den folgenden Befehl aus, um MCP Inspector zu starten:

```
npx @modelcontextprotocol/inspector
```

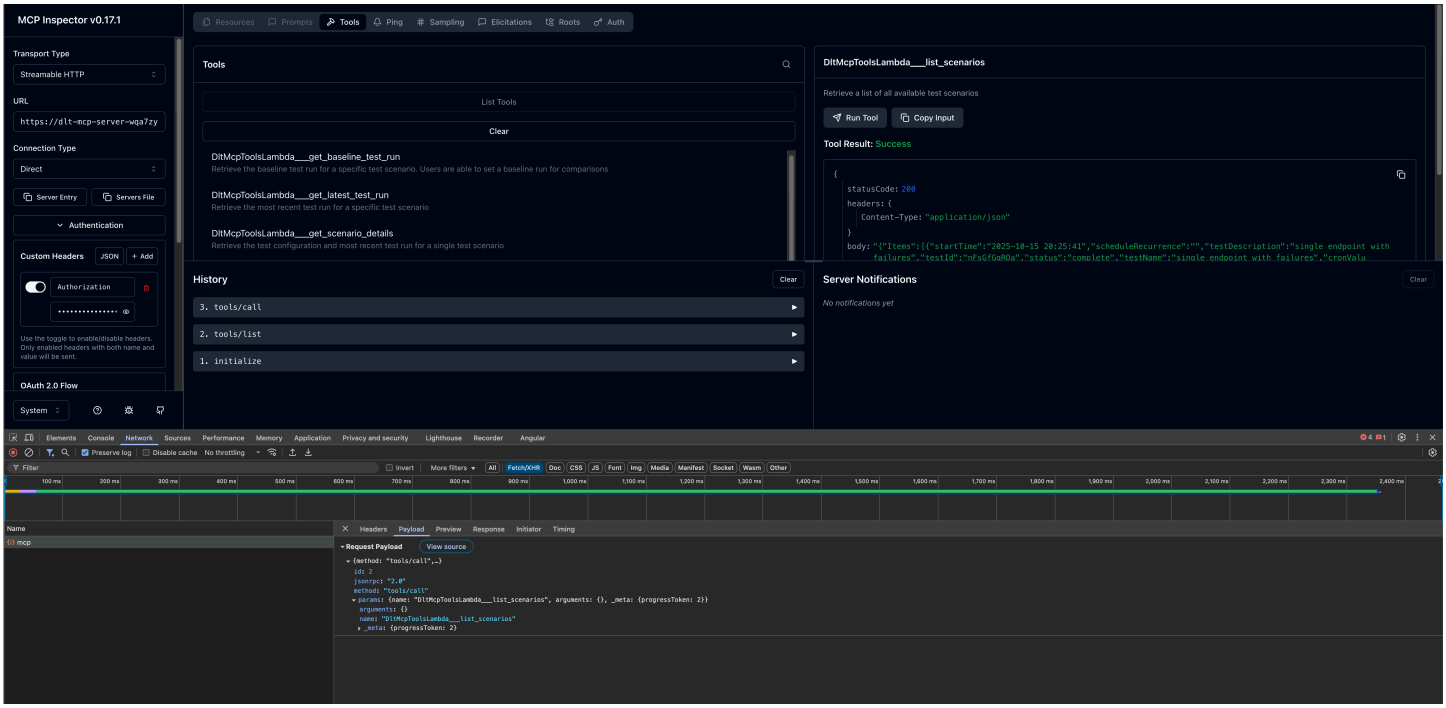
Konfigurieren Sie die Verbindung

1. Geben Sie in der MCP Inspector-Oberfläche Ihre MCP Server-Endpunkt-URL ein.
2. Fügen Sie einen Autorisierungsheader mit Ihrem Zugriffstoken hinzu.
3. Wählen Sie Connect, um die Verbindung herzustellen.

Rufen Sie Tools auf

Sobald die Verbindung hergestellt ist, können Sie die verfügbaren MCP-Tools testen:

1. Durchsuchen Sie die Liste der verfügbaren Tools im linken Bereich.
2. Wählen Sie ein Werkzeug aus (z. B.`list_scenarios`).
3. Geben Sie alle erforderlichen Parameter an.
4. Wählen Sie Invoke, um das Tool auszuführen und die Antwort anzuzeigen.



Schritt 3: Konfigurieren Sie KI-Entwicklungsclients

Nachdem Sie Ihre MCP-Serververbindung mit MCP Inspector überprüft haben, können Sie Ihren bevorzugten AI-Entwicklungsclient konfigurieren.

Kiro CLI

Kiro CLI (ehemals Amazon Q Developer CLI) bietet Befehlszeilenzugriff auf die AI-assisted Entwicklung mit MCP-Serverintegration.

Schritte zur Konfiguration

1. Bearbeiten Sie die `mcp.json` Konfigurationsdatei. Weitere Informationen zum Speicherort der Konfigurationsdateien finden Sie unter Model Context Protocol (MCP) in der Kiro CLI-Dokumentation.
2. Fügen Sie Ihre DLT MCP-Serverkonfiguration hinzu:

```
{
  "mcpServers": {
    "dlt-mcp": {
      "type": "http",
      "url": "https://<gateway-id>.gateway.bedrock-agentcore.<region>.amazonaws.com/
mcp",
```

```
      "headers": {
        "Authorization": "Bearer <access-token>"
      }
    }
  }
}
```

Ersetzen Sie <gateway-id> und <region> durch die Werte aus Ihrer MCP Server-Endpunkt-URL und durch den Wert, <access-token> den Sie in Schritt 1 kopiert haben.

Überprüfen Sie die Konfiguration

1. Geben Sie in einem Terminal ein, `kiro-cli` um Kiro CLI zu starten.
2. Geben Sie `/mcp` ein, um alle verfügbaren MCP-Server zu sehen.
3. Geben Sie `/tools` ein, um die verfügbaren Tools von `dlt-mcp` und anderen konfigurierten MCP-Servern anzuzeigen.
4. Stellen Sie sicher, dass die Initialisierung `dlt-mcp` erfolgreich war.

Cline

Cline ist ein KI-Codierungsassistent, der die MCP-Serverintegration unterstützt.

Schritte zur Konfiguration

1. Navigieren Sie in Cline zu MCP-Server verwalten > Konfigurieren > MCP-Server konfigurieren.
2. Aktualisieren Sie die Datei: `cline_mcp_settings.json`

```
{
  "mcpServers": {
    "dlt-mcp": {
      "type": "streamableHttp",
      "url": "https://<gateway-id>.gateway.bedrock-agentcore.<region>.amazonaws.com/
mcp",
      "headers": {
        "Authorization": "Bearer <access-token>"
      }
    }
  }
}
```

Ersetzen Sie `<gateway-id>` und `<region>` durch die Werte aus Ihrer MCP Server-Endpunkt-URL und `<access-token>` durch den Wert, den Sie in Schritt 1 kopiert haben.

3. Speichern Sie die Konfigurationsdatei.
4. Starten Sie Cline neu, um die Änderungen zu übernehmen.

Amazon Quick

Amazon Quick (ehemals Amazon Quick Suite) bietet eine umfassende KI-Assistentenplattform mit Unterstützung für MCP-Serveraktionen.

Voraussetzungen

Bevor Sie den MCP-Server in Amazon Quick konfigurieren, müssen Sie die OAuth-Anmeldeinformationen aus dem Cognito-Benutzerpool Ihrer DLT-Bereitstellung abrufen:

1. Navigieren Sie zur CloudFormation AWS-Konsole.
2. Wählen Sie den Distributed Load Testing-Stack aus.
3. Suchen Sie auf der Registerkarte Ausgaben die Cognito-Benutzerpool-ID, die Ihrer DLT-Bereitstellung zugeordnet ist, und kopieren Sie sie.
4. Navigieren Sie zur Amazon-Cognito-Konsole.
5. Wählen Sie den Benutzerpool anhand der Benutzerpool-ID aus den CloudFormation Ausgaben aus.
6. Wählen Sie in der linken Navigationsleiste App-Integration > App-Clients aus.
7. Suchen Sie den App-Client, dessen Name auf `m2m` (Machine-to-Machine) endet, und wählen Sie ihn aus.
8. Kopieren Sie auf der Detailseite des App-Clients die Client-ID. Um das Client-Geheimnis aufzudecken, wählen Sie Client-Geheimnis anzeigen und kopieren Sie dann den Wert.
9. Kehren Sie zum Benutzerpool zurück und rufen Sie die Benutzerpool-Domäne auf der Registerkarte Domäne ab.
10. Konstruieren Sie die Token-Endpunkt-URL, indem `/oauth2/token` Sie sie an das Ende der Domäne anhängen.

Schritte zur Konfiguration

1. Erstellen Sie in Amazon Quick einen neuen Agenten oder wählen Sie einen vorhandenen Agenten aus.
2. Fügen Sie eine Agentenaufforderung hinzu, die beschreibt, wie Sie mit dem DLT MCP-Server interagieren.
3. Fügen Sie eine neue Aktion hinzu und wählen Sie MCP-Serveraktion aus.
4. Konfigurieren Sie die MCP-Serverdetails:
   - MCP-Server-URL: Ihr DLT-MCP-Endpunkt
   - Authentifizierungstyp: Authentifizierung Service-based
   - Token-Endpunkt: Ihre Cognito-Token-Endpunkt-URL
   - Client-ID: Die Client-ID des M2M-App-Clients
   - Geheimer Client-Schlüssel: Der geheime Client-Schlüssel des M2M-App-Clients
5. Speichern Sie die Aktionskonfiguration des MCP-Servers.
6. Fügen Sie Ihrem Agenten die neue MCP-Serveraktion hinzu.

Starten und testen Sie den Agenten

1. Starten Sie den Agenten in Amazon Quick.
2. Starten Sie mithilfe von Eingabeaufforderungen in natürlicher Sprache eine Konversation mit dem Agenten.
3. Der Agent verwendet die MCP-Tools, um Ihre Lasttestdaten abzurufen und zu analysieren.

Rotation des geheimen MCP-Server-Clients

Wenn Sie die dienstbasierte Authentifizierung (von Maschine zu Maschine) verwenden, um einen MCP-Client mit der Lösung zu verbinden, sind Sie dafür verantwortlich, den vom Client verwendeten geheimen Client-Schlüssel zu rotieren.

Für welche Anmeldeinformationen gilt das

Die Lösung gibt zwei verschiedene MCP-Anmeldeinformationen aus. Nur einer von ihnen erfordert eine manuelle Rotation.

Credential	Verwendet von	Rotation (Drehung)
Benutzerz ugriffstoken — wurde von der MCP-Serverseite der Webkonsole kopiert	MCP Inspector, Kiro CLI, Cline und andere Clients, die einen Header senden `Authorization: Bearer`	Keine erforderlich. Dies ist ein kurzlebiges Amazon Cognito Cognito-Zugriffstoken, das ungefähr eine Stunde nach seiner Ausgabe abläuft. Um ein neues Token zu erhalten, kehren Sie zur MCP-Serve rseite zurück und kopieren Sie es erneut.
Machine-to-machine Client-ID und Client-Geheimnis — abgerufen aus dem Amazon Cognito Cognito-Benutzerpool	Amazon Quick und alle anderen Clients, die für die servicebasierte Authentifizierung konfiguriert sind	Manuelle Rotation erforderlich. Das Kundengeheimnis ist langlebig und läuft nicht von selbst ab.

Der Rest dieses Abschnitts bezieht sich auf das Computer-zu-Maschine-Client-Geheimnis. Die Lösung erstellt diese Anmeldeinformationen als Amazon Cognito-App-Client, der benannt wird, `<stack-name>-userpool-client-m2m` wenn Sie die Bereitstellung mit Deploy Optional MCP Server auf einrichten. `Yes` Anweisungen zum Abrufen finden Sie in den Voraussetzungen in Amazon Quick.

Important

Behandeln Sie das Kundengeheimnis wie alle anderen langlebigen Anmeldeinformationen:

- Übergeben Sie es nicht der Quellcodeverwaltung, betten Sie es nicht in den Anwendungscode ein und fügen Sie es nicht in Issue Tracker, Chats oder Dokumentationen ein.
- Schreiben Sie es nicht in Logs, Build-Ausgaben oder CI/CD Job-Ausgaben.

- Speichern Sie es in einem Secrets Manager wie AWS Secrets Manager oder im verschlüsselten Anmeldeinformationsspeicher des MCP-Clients, der es verwendet. Bewahren Sie es nicht in einer Klartextdatei auf.
- Gewähren Sie nur denjenigen Zugriff auf das Geheimnis, die es benötigen, z. B. Ihnen und den Diensten, die das Geheimnis verwenden.

Empfohlene Rotationskadenz

Wechseln Sie den geheimen Client-Schlüssel von Maschine zu Maschine mindestens alle 90 Tage. Wechseln Sie sofort, außerhalb des regulären Zeitplans, wenn eines der folgenden Ereignisse eintritt:

- Sie vermuten oder bestätigen, dass das Geheimnis gelüftet wurde.
- Ein Operator mit Zugriff auf das Geheimnis verlässt das Team oder wechselt die Rollen.
- Sie stellen einen MCP-Client, der mit dem Secret konfiguriert wurde, außer Betrieb.

Rotation des Secrets

Ein Amazon Cognito-App-Client unterstützt bis zu zwei aktive Client-Geheimnisse gleichzeitig. Wechseln Sie, indem Sie ein zweites Geheimnis hinzufügen, Ihre MCP-Clients dorthin migrieren und dann das Original löschen — ohne Unterbrechung des MCP-Zugriffs.

Die Client-ID ändert sich während der Rotation nicht, sodass Sie den CloudFormation AWS-Stack der Lösung, den MCP-Serverendpunkt oder den Token-Endpunkt nicht aktualisieren müssen. Nur der geheime Wert, der in Ihrem MCP-Client gespeichert ist, ändert sich.

1. Fügen Sie einen zweiten geheimen Client-Schlüssel hinzu. Amazon Cognito generiert den Wert und gibt ihn in der Antwort zurück.

```
aws cognito-idp add-user-pool-client-secret \
  --user-pool-id <user-pool-id> \
  --client-id <m2m-client-id> \
  --region <region>
```

> **Important**
>
> Kopieren Sie den `ClientSecretValue` aus der Antwort und speichern Sie ihn sicher, bevor Sie fortfahren. Amazon Cognito gibt den generierten geheimen Wert nur in dieser

Antwort zurück und gibt ihn nie wieder preis — `list-user-pool-client-secrets` weder die Amazon Cognito Cognito-Konsole wird ihn anzeigen. Wenn Sie den Wert verlieren, löschen Sie das Geheimnis und fügen Sie ein neues hinzu.

Das ursprüngliche Geheimnis bleibt zu diesem Zeitpunkt gültig, sodass jeder MCP-Client, der noch damit konfiguriert ist, weiterhin funktioniert.

2. Aktualisieren Sie jeden MCP-Client, sodass er das neue Geheimnis verwendet. Bearbeiten Sie für Amazon Quick die MCP-Serveraktion und ersetzen Sie den Wert Client Secret, sodass die MCP-Server-URL, der Token-Endpunkt und die Client-ID unverändert bleiben. Speichern Sie die Aktion.
3. Stellen Sie sicher, dass das neue Geheimnis Tokens ausgibt. Beantragen Sie vom Token-Endpunkt Ihres Benutzerpools eine Erteilung von Kundenanmeldedaten.

```
curl -X POST https://<user-pool-domain>/oauth2/token \
  -H 'Content-Type: application/x-www-form-urlencoded' \
  -d 'grant_type=client_credentials' \
  -d 'client_id=<m2m-client-id>' \
  -d 'client_secret=<new-client-secret>' \
  -d 'scope=dlt-mcp-gateway/read'
```

Eine erfolgreiche Antwort enthält ein `access_token` Feld. Bestätigen Sie anschließend den End-to-End-Zugriff, indem Sie ein MCP-Tool von dem Client aus aufrufen, den Sie neu konfiguriert haben. Bitten Sie beispielsweise den Agenten, Ihre Testszenarien aufzulisten.

4. Führen Sie die Geheimnisse des Kunden auf, um das ursprüngliche Geheimnis zu identifizieren. Jedes Geheimnis wird `ClientSecretId` im Format mit einem identifiziert *`<client-id>--<epoch-create-time>`* . Verwenden Sie das `ClientSecretCreateDate` Feld, um das ursprüngliche Geheimnis von dem zu unterscheiden, das Sie gerade hinzugefügt haben.

```
aws cognito-idp list-user-pool-client-secrets \
  --user-pool-id <user-pool-id> \
  --client-id <m2m-client-id> \
  --region <region>
```

5. Macht das ursprüngliche Geheimnis ungültig. Nach diesem Anruf gibt Amazon Cognito keine Token mehr an Kunden aus, die das alte Geheimnis präsentieren.

```
aws cognito-idp delete-user-pool-client-secret \
  --user-pool-id <user-pool-id> \
```

```
--client-id <m2m-client-id> \
--client-secret-id <old-client-secret-id> \
--region <region>
```

> **Note**
>
> Beim Rotieren gelten zwei Einschränkungen:
>
> - Ein App-Client kann maximal zwei Geheimnisse haben. Wenn bereits zwei vorhanden sind, löschen Sie den, den Sie nicht mehr benötigen, bevor Sie einen weiteren hinzufügen.
> - Sie können das letzte verbleibende Geheimnis auf einem App-Client nicht löschen.
>
> Der `add-user-pool-client-secret` Befehl akzeptiert auch einen optionalen `--client-secret` Parameter für die Angabe Ihres eigenen Werts. Wenn Sie einen Wert angeben, gibt Amazon Cognito ihn in der Antwort nicht zurück. Sie müssen ihn also speichern, bevor Sie den Anruf tätigen. Wir empfehlen, Amazon Cognito das Geheimnis generieren zu lassen.

Wenn das Geheimnis aufgedeckt wurde

Gehen Sie wie zuvor beschrieben vor und schließen Sie Schritt 5 (Löschen des alten Geheimnisses) so schnell wie möglich ab, anstatt auf ein Wartungsfenster zu warten. Ergreifen Sie anschließend die folgenden zusätzlichen Maßnahmen.

- Konto für bereits ausgegebene Tokens. Durch das Löschen eines Geheimnisses kann Amazon Cognito keine neuen Token ausgeben, aber Zugriffstoken, die mit dem offengelegten Geheimnis erworben wurden, bleiben gültig, bis sie ablaufen — bis zu etwa einer Stunde. Durch das Löschen des Geheimnisses werden sie nicht ungültig.
- Schränkt ein, was diese Token bewirken können. Wenn Ihre Bereitstellung `ReadWrite` den Zugriffsmodus verwendet, führen Sie ein CloudFormation AWS-Stack-Update durch, wobei der MCP-Serverzugriffsmodus auf `ReadOnly` eingestellt ist. Dadurch werden die Schreibtools entfernt und die IAM-Berechtigungen der Lambda-Funktion des MCP Servers auf `GET` Anfragen eingeschränkt, sodass ausstehende Token keine Testszenarien erstellen, ändern, löschen oder starten können. Informationen zum Verhalten der einzelnen Zugriffsmodi finden Sie in der Spezifikation der MCP-Tools im Entwicklerhandbuch.

- Überprüfen Sie, wofür die Anmeldeinformationen verwendet wurden. Überprüfen Sie den Testlaufverlauf der Lösung und die CloudWatch Amazon-Logs für die Lambda-Funktion des MCP Servers auf unerwartete Aktivitäten. Wenn für die Bereitstellung der `ReadWrite` Zugriffsmodus verwendet wurde, überprüfen Sie auch Ihre Testszenarien und den Inhalt des `public/test-scenarios/` Präfixes des Szenario-Buckets auf nicht autorisierte Änderungen.

> **Note**
>
> Um alle MCP-Anfragen unabhängig von der Gültigkeit des Tokens sofort abzulehnen, führen Sie ein Stack-Update durch, bei dem die Option Optionalen MCP-Server bereitstellen auf eingestellt ist. No Dadurch wird das Gateway gelöscht. AgentCore
> Verwenden Sie dies nur, wenn Sie sicherstellen müssen, dass kein ausstehendes Token den MCP-Server erreichen kann. Wenn Sie den Parameter auf einstellen, wird No auch der Maschine-zu-Maschine-App-Client gelöscht. Wenn Sie ihn wieder auf setzen, `Yes` wird ein neuer App-Client mit einer neuen Client-ID und einem neuen geheimen Schlüssel erstellt. Anschließend müssen Sie jeden MCP-Client sowohl mit der neuen Client-ID als auch mit dem neuen Secret neu konfigurieren.

Beispiele für Prompts

Die folgenden Beispiele zeigen, wie Sie mit Ihrem KI-Assistenten interagieren können, um Lasttestdaten über die MCP-Schnittstelle zu analysieren. Passen Sie die Test-IDs, Datumsbereiche und Kriterien an Ihre spezifischen Testanforderungen an.

Ausführliche Informationen zu den verfügbaren MCP-Tools und ihren Parametern finden Sie in der MCP-Tool-Spezifikation im Developer Guide.

Einfache Abfrage der Testergebnisse

Die Interaktion mit dem MCP-Server in natürlicher Sprache kann so einfach wie `Show me the load tests that have completed in the last 24 hours with their associated completion status` oder aussagekräftiger sein, z. B.

```
Use list_scenarios to find my load tests. Then use get_latest_test_run to show me the
 basic execution data and performance metrics for the most recent test. If the results
 look concerning, also get the detailed performance metrics using get_test_run.
```

Interaktive Leistungsanalyse mit schrittweiser Offenlegung

```
I need to analyze my load test performance, but I'm not sure which specific tests to
 focus on. Please help me by:

1. First, use list_scenarios to show me available test scenarios
2. Ask me which tests I want to analyze based on the list you show me
3. For my selected tests, use list_test_runs to get the test run history
4. Then use get_test_run with the test_run_id to get detailed response times,
 throughput, and error rates
5. If I want to compare tests, use get_baseline_test_run to compare against the
 baseline
6. If there are any issues, use get_test_run_artifacts to help me understand what went
 wrong

Please guide me through this step by step, asking for clarification whenever you need
 more specific information.
```

Validierung der Produktionsreife

```
Help me validate if my API is ready for production deployment:

1. Use list_scenarios to find recent test scenarios
2. For the most recent test scenario, use get_latest_test_run to get basic execution
 data
3. Use get_test_run with that test_run_id to get detailed response times, error rates,
 and throughput
4. Use get_scenario_details with the test_id to show me what load patterns and
 endpoints were tested
5. If I have a baseline, use get_baseline_test_run to compare current results with the
 baseline
6. Provide a clear go/no-go recommendation based on the performance data
7. If there are any concerns, use get_test_run_artifacts to help identify potential
 issues

My SLA requirements are: response time under [X]ms, error rate under [Y]%.
```

Analyse der Leistungstrends

```
Analyze the performance trend for my load tests over the past [TIME_PERIOD]:

1. Use list_scenarios to get all test scenarios
```

```
2. For each scenario, use list_test_runs with start_date and end_date to get tests from
 that period
3. Use get_test_run for the key test runs to get detailed metrics
4. Use get_baseline_test_run to compare against the baseline
5. Identify any significant changes in response times, error rates, or throughput
6. If you detect performance degradation, use get_test_run_artifacts on the problematic
 tests to help identify causes
7. Present the trend analysis in a clear format showing whether performance is
 improving, stable, or degrading

Focus on completed tests and limit results to [N] tests if there are too many.
```

Behebung fehlgeschlagener Tests

```
Help me troubleshoot my failed load tests:

1. Use list_scenarios to find test scenarios
2. For each scenario, use list_test_runs to find recent test runs
3. Use get_test_run with the test_run_id to get the basic execution data and failure
 information
4. Use get_test_run_artifacts to get detailed error messages and logs
5. Use get_scenario_details to understand what was being tested when it failed
6. If I have a similar test that passed, use get_baseline_test_run to identify
 differences
7. Summarize the causes of failure and suggest next steps for resolution

Show me the most recent [N] failed tests from the past [TIME_PERIOD].
```

Integration mit DevOps AWS-Agenten

Mit AWS DevOps Agent können Sie die Untersuchung Ihrer Lasttestergebnisse direkt aus der Lösung Distributed Load Testing on AWS heraus automatisieren. Wenn Sie einen Agent Space verbinden, können Sie abgeschlossene Testläufe zur Analyse an den Agenten senden. Der Agent identifiziert proaktiv Leistungsengpässe, ermittelt die Ursachen und schlägt Schritte zur Behebung vor, ohne dass Sie die CloudWatch Amazon-Statistiken, -Protokolle und -Traces manuell korrelieren müssen.

Voraussetzungen

Bevor Sie einen Agent Space mit Distributed Load Testing on AWS (DLT) verbinden, gehen Sie wie folgt vor:

- Der Agent Space muss sich in demselben AWS-Konto wie Ihre DLT-Bereitstellung befinden.
- Der Agent Space muss mit dem Schlüssel `dlt-integration` und dem Wert `allowed` gekennzeichnet sein. Dieses Tag stellt eine doppelte Zustimmung dar: Sowohl der DLT-Benutzer als auch der Agent Space-Administrator müssen explizite Maßnahmen ergreifen, bevor DLT auf den Agent Space zugreifen kann.
- Das `dlt-integration: allowed` Tag muss vorhanden sein, bevor DLT Testverbindungs- oder Ermittlungsanfragen annehmen kann, auch wenn der Agent Space-ARN gültig ist.

Um das Tag hinzuzufügen, wählen Sie die DevOps AWS-Agent-Konsole und dann Ihren Agent Space aus. Wählen Sie die Registerkarte Konfiguration aus. Scrollen Sie nach unten zum Abschnitt Tags, wählen Sie Neues Tag hinzufügen aus, geben Sie es `dlt-integration` als Schlüssel und `allowed` als Wert ein und wählen Sie dann Speichern.

Registrieren Sie einen Agentenbereich

Nachdem Ihr Agent Space markiert wurde, führen Sie die folgenden Schritte durch, um ihn bei DLT zu registrieren:

1. Wählen Sie in der linken Seitenleiste der DLT-Webkonsole die Option Agentenintegration aus.
2. Wählen Sie im Bereich DevOps Agentenverbindungen die Option Agent Space registrieren aus.
3. Geben Sie einen Anzeigenamen ein und fügen Sie den Agent Space-ARN aus der DevOps AWS-Agent-Konsole ein.
4. Wählen Sie Verbindung testen, um zu überprüfen, ob der ARN erreichbar ist und ob das `dlt-integration: allowed` Tag vorhanden ist.
5. Nachdem die Verbindung bestätigt wurde, wählen Sie Speichern.

Sie können mehrere Agent Spaces registrieren und zwischen ihnen wählen, wenn Sie eine Untersuchung auslösen.

Untersuchen Sie einen Testlauf

Nachdem Sie mindestens einen Agent Space registriert haben, können Sie von jedem abgeschlossenen Testlauf aus eine Untersuchung starten. Die Schaltfläche DevOps Investigate with Agent wird auf jeder Detailseite des Testlaufs angezeigt. Führen Sie folgende Schritte aus:

1. Wählen Sie DevOps Investigate with Agent nach Abschluss eines Testlaufs — erfolgreich, fehlgeschlagen oder unvollständig — aus.
2. Wenn Sie mehrere Agent Spaces registriert haben, wählen Sie einen aus der Liste aus.
3. Geben Sie optional zusätzlichen Kontext an, z. B. eine Beschreibung der letzten Infrastrukturänderungen oder die erwartete Ausgangsleistung.
4. Wählen Sie Absenden aus.

Die Untersuchung wird auf der Registerkarte Untersuchungen auf der Detailseite des Testlaufs angezeigt. Dort können Sie ihren Status verfolgen und den Link auswählen, um die Ergebnisse in der DevOps AWS-Agentenkonsole anzuzeigen.

> **Note**
>
> AWS DevOps Agent erzwingt eine Obergrenze von 10.000 Zeichen für die Beschreibung der Untersuchung. Die von der Lösung gesendete Beschreibung kombiniert Ihren optionalen Kontext mit einer DLT-generated Zusammenfassung des Testlaufs (Testkonfiguration, Vergleich der Ausgangswerte, sortiert nach Anzahl der häufigsten Fehler und Aufschlüsselung pro Endpunkt). Um das Limit einzuhalten, priorisiert die Lösung die Testkonfiguration und den vom Benutzer bereitgestellten Kontext und kürzt Abschnitte mit niedrigerer Priorität — beginnend mit der Aufschlüsselung pro Endpunkt —, wenn die kombinierte Länge andernfalls 10.000 Zeichen überschreiten würde.

Überprüfen der Erkenntnisse

DevOps AWS-Agent-Untersuchungen werden in der Regel innerhalb weniger Minuten abgeschlossen, können jedoch je nach Agent Space-Konfiguration unterschiedlich sein. Wenn Sie fertig sind, wählen Sie auf der Registerkarte Untersuchungen die Option Ergebnisse anzeigen, um die Ergebnisse in der DevOps AWS-Agentenkonsole zu öffnen.

Zu den Ergebnissen gehören die folgenden:

- Eine strukturierte Ursachenanalyse, die ermittelt, welche Komponenten zu Latenz oder Fehlern beigetragen haben.
- Eine Liste der Symptome, die während des Testfensters beobachtet wurden.
- Konkrete Vorschläge für Abhilfemaßnahmen.

Untersuchungen verbleiben für die gesamte Lebensdauer des Testlaufdatensatzes auf der Registerkarte DLT-Untersuchungen, sodass Sie bei der Weiterentwicklung Ihrer Anwendung auf frühere Analysen zurückgreifen können.

Vorgeschlagener Workflow

Um das Beste aus der DevOps AWS-Agenten-Integration herauszuholen, verwenden Sie sie als Teil einer iterativen Leistungsoptimierungsschleife: bereitstellen, laden, testen, untersuchen, korrigieren und wiederholen. Bei jedem Durchlauf der Schleife erhält der Agent mehr Historie, mit der er vergleichen kann, und Sie erhalten messbare Beweise dafür, dass eine Änderung die Leistung verbessert hat.

1. Bereitstellung — Stellen Sie Ihre Anwendung in der Umgebung bereit, die Sie testen möchten, und führen Sie die unter Voraussetzungen und Registrierung eines Agentenbereichs beschriebene einmalige Einrichtung durch.
2. Erstellen Sie eine Baseline — Führen Sie Ihr Testszenario mit einem zweifelsfrei funktionierenden Build aus und legen Sie den Testlauf als Baseline fest (siehe Baseline-Vergleich). Wenn ein Basiswert festgelegt ist, beinhaltet die Lösung einen Vergleich mit diesem mit Ihrer Untersuchungsanfrage, sodass der Agent eine Regression von einer normalen Variation unterscheiden kann.
3. Auslastungstest — Führen Sie das Testszenario nach jeder Anwendungs- oder Infrastrukturänderung aus. Halten Sie die Szenariokonfiguration zwischen den Läufen konstant, damit die Ergebnisse vergleichbar bleiben.
4. Untersuchen — Wählen Sie nach Abschluss des Testlaufs auf der Detailseite des Testlaufs die Option Investigate with DevOps Agent aus. Geben Sie einen Überblick darüber, was sich seit dem letzten Lauf geändert hat — aktuelle Bereitstellungen, Konfigurationsänderungen oder die erwartete Leistung —, damit sich der Agent auf seine Analyse konzentrieren kann.
5. Behebung — Überprüfen Sie die Grundursache und die Vorschläge zur Behebung auf der Registerkarte Untersuchungen oder öffnen Sie die Ergebnisse in der DevOps AWS-Agentenkonsole, um einen Plan zur Schadensbegrenzung zu erstellen. Wenden Sie das Update auf Ihre Anwendung oder Infrastruktur an.
6. Wiederholen Sie Re-run das Testszenario und vergleichen Sie die Ergebnisse mit Ihrer Ausgangssituation, um das Update zu überprüfen. Wenn sich die Leistung verbessert, legen Sie den neuen Testlauf als Ausgangswert fest und setzen Sie den Vorgang mit der nächsten Änderung fort.

> **Note**
>
> Jeder Testlauf unterstützt jeweils eine aktive Untersuchung. Archivieren Sie eine abgeschlossene Untersuchung auf der Registerkarte Untersuchungen, bevor Sie im selben Testlauf eine neue Untersuchung starten.

Fehlerbehebung

Die Lösung bekannter Probleme enthält Anweisungen zur Behebung bekannter Fehler. Wenn diese Anweisungen Ihr Problem nicht lösen, finden Sie unter Wenden Sie sich an den AWS-Support. Dort finden Sie Anweisungen zum Öffnen einer AWS-Supportanfrage für diese Lösung.

Lösung eines bekannten Problems

Problem: Sie verwenden eine bestehende VPC und Ihre Tests schlagen mit dem Status Fehlgeschlagen fehl, was zu der folgenden Fehlermeldung führt:

```
Test might have failed to run.
```

- Auflösung

Stellen Sie sicher, dass die Subnetze in der angegebenen VPC vorhanden sind und dass sie über eine Route zum Internet entweder mit einem Internet-Gateway oder einem NAT-Gateway verfügen. AWS Fargate benötigt Zugriff, um das Container-Image aus dem öffentlichen Repository abzurufen, um Tests erfolgreich ausführen zu können.

Problem: Die Ausführung von Tests dauert zu lange oder läuft auf unbestimmte Zeit

- Auflösung

Brechen Sie den Test ab und überprüfen Sie AWS Fargate, um sicherzustellen, dass alle Aufgaben beendet wurden. Wenn sie nicht gestoppt wurden, beenden Sie alle Fargate-Aufgaben manuell. Überprüfen Sie die Fargate-Aufgabenlimits auf Abruf in Ihrem Konto, um sicherzustellen, dass Sie die gewünschte Anzahl von Aufgaben starten können. Sie können auch die CloudWatch Protokolle der Lambda-Task-Runner-Funktion überprüfen, um mehr über Fehler beim Starten von Fargate-Aufgaben zu erfahren. In den CloudWatch ECS-Protokollen finden Sie Einzelheiten darüber, was in laufenden Fargate-Containern passiert.

Problem: Die Tests werden gestartet, können aber nicht abgeschlossen werden oder der Status der ECS-Aufgaben ist unbekannt

- Auflösung

Wenn Sie die Option zur Bereitstellung einer vorhandenen VPC in dem Konto ausgewählt haben, in dem die Lösung bereitgestellt wurde, stellen Sie sicher, dass die von den ECS-Aufgaben verwendete VPC über genügend freie IP-Adressen verfügt, um die in der Testeingabe angegebene Anzahl von Aufgaben zu starten. Die ECS-Aufgabendefinition verwendet das ECR-Image, das ein Internet-Gateway oder eine Route zum Internet benötigt, sodass der ECS-Dienst die Aufgaben bereitstellen kann, indem er das Lösungs-ECR-Image von aws- solutions/distributed -load-testing-on-aws-load-tester herunterlädt. Wenn Sie keine Route zum Internet bereitstellen können, da alle Subnetze in der VPC privat sind, können Sie das ECR-Image mithilfe des ECR-Pull-Through-Cache in Ihrem Konto hosten. Aktualisieren Sie die Aufgabendefinition mit der neuen ECR-Image-URI und erstellen Sie eine neue Revision. Sobald die Aufgabendefinition aktualisiert wurde, muss die Lösungskonfiguration in der DynamoDB-Tabelle aktualisiert werden, um die neue Version verwenden zu können. Den Namen der DynamoDB-Tabelle finden Sie auf der Registerkarte CloudFormation Stack-Ausgaben unter dem Schlüssel. ScenariosTable Aktualisieren Sie das Attribut taskDefinition für das Element mit dem Schlüssel testID und dem Wert region- []. SOLUTION-DEPLOYED-REGION

Problem: Tests müssen einen Endpunkt verwenden, der privat ist oder nicht über das Internet-Gateway verfügbar ist

- Auflösung

Wenn Sie private API-Endpunkte testen, auf die nicht über das Internet-Gateway zugegriffen werden kann, sollten Sie die folgenden Ansätze in Betracht ziehen:

1. Netzwerkkonfiguration: Stellen Sie sicher, dass die von den ECS-Aufgaben verwendeten Subnetz-Routentabellen mit einer Route zum IP-Adressbereich des privaten Endpunkts aktualisiert werden, der getestet wird. Dadurch kann der Testdatenverkehr den privaten Endpunkt innerhalb Ihrer VPC erreichen.
2. DNS-Auflösung: Konfigurieren Sie für benutzerdefinierte Domänen die DNS-Einstellungen in Ihrer VPC, um den Domainnamen des privaten Endpunkts aufzulösen. Detaillierte Anweisungen finden Sie in der Dokumentation zu VPC DNS.
3. VPC-Endpunkte: Wenn Sie AWS-Services testen, sollten Sie die Verwendung von VPC-Endpunkten (AWS PrivateLink) in Betracht ziehen, um private Konnektivität herzustellen. Um beispielsweise ein privates API Gateway zu testen, können Sie einen VPC-Endpunkt für API Gateway erstellen. Weitere Informationen finden Sie in der Dokumentation zu Private API Gateway.
4. VPC-Peering: Wenn sich der private Endpunkt in einer anderen VPC befindet, richten Sie ein VPC-Peering zwischen der VPC, auf der die Lösung bereitgestellt wird, und der VPC mit dem privaten

Endpunkt ein. Konfigurieren Sie die entsprechenden Routing-Tabellen in beiden VPCs. Weitere Informationen finden Sie in der VPC-Peering-Dokumentation.

5. Transit Gateway: Für komplexere Netzwerkszenarien mit mehreren VPCs sollten Sie die Verwendung von AWS Transit Gateway zur Weiterleitung des Datenverkehrs zwischen der VPC der Lösung und der VPC, die den privaten Endpunkt enthält, in Betracht ziehen. Siehe Transit Gateway Gateway-Dokumentation.
6. Sicherheitsgruppen: Stellen Sie sicher, dass die mit Ihren ECS-Aufgaben verknüpften Sicherheitsgruppen ausgehenden Datenverkehr zum privaten Endpunkt zulassen und dass die Sicherheitsgruppen des privaten Endpunkts eingehenden Datenverkehr von den ECS-Aufgaben zulassen.

Stellen Sie beim Testen interner Application Load Balancers oder EC2-Instances sicher, dass sich die VPC-CIDR-Bereiche nicht überschneiden und dass die erforderlichen Routen in den Routentabellen konfiguriert sind.

Problem: Die Tests werden abgeschlossen, aber die Ergebnisse sind nicht auf der Benutzeroberfläche verfügbar

- Auflösung

Wenn der Test abgeschlossen wurde, die Ergebnisse jedoch nicht in der Benutzeroberfläche verfügbar sind, sollten die Ergebnisdateien weiterhin im S3-Bucket der ECS-Aufgaben verfügbar sein, die die Tests ausgeführt haben. Dies ist eine bekannte Einschränkung der Lösung. In der aktuellen Architektur verwendet die Lösung eine Lambda-Funktion zur Ergebnisanalyse, um die Ergebnisse mehrerer ECS-Aufgaben zusammenzufassen, die dann als Element in der DynamoDB-Tabelle gespeichert werden. Die DynamoDB-Tabelle hat eine maximale Elementgröße von 400 KB. Diese Beschränkung wird abhängig von der Komplexität des Testskripts, der Parallelität und der Anzahl der verwendeten Aufgaben erreicht. Der Fehler bedeutet nicht, dass der Test fehlschlägt, sondern bedeutet, dass der Prozess zur Zusammenfassung der Ergebnisse und deren Speicherung in der DynamoDB-Tabelle für CRUD-Operationen fehlgeschlagen ist. Die Ergebnisse sind weiterhin im S3-Bucket für das Testszenario verfügbar.

Problem: Ein Test im systemeigenen Modus endet früher als vom Skript vorgesehen, und die Ausführung wird als abgeschlossen aufgezeichnet

- Auflösung

Der Test hat seine Sicherheitsdauer erreicht. Im systemeigenen Modus entscheidet Ihr Skript, wann der Testlauf beendet wird. Daher benötigt die Lösung eine Sicherheitsdauer und beendet das Test-Framework, wenn diese abgelaufen ist. Die Lösung zeichnet die Ausführung als abgeschlossen und nicht als fehlgeschlagen auf und behält die Ergebnisse für den Teil bei, der ausgeführt wurde. Erhöhen Sie die Sicherheitsdauer über die längste Ausführung, die das Skript benötigt, bis auf den Höchstwert von 24 Stunden. Weitere Informationen finden Sie unter Verkehrsshape-Modi.

Problem: Ein Locust-Test im systemeigenen Modus meldet weit weniger Anfragen als das generierte Skript

- Auflösung

Prüfen Sie, ob Ihr Locust-Skript gesetzt ist. `processes` Die Lösung zählt Anfragen nur, wenn Locust als ein einziger Prozess ausgeführt wird, sodass ein Skript, das Arbeitsprozesse erzeugt, seine Ergebnisse zu wenig meldet. Entfernen Sie die `processes` Einstellung und skalieren Sie die Last stattdessen anhand der Anzahl der Aufgaben, da für jede Aufgabe bereits eine vollständige Kopie des Skripts ausgeführt wird. Weitere Informationen finden Sie unter Locust-Tests.

Probleme bei der Bereitstellung von ALB + ECS Fargate

Problem: Die ACM-Zertifikatsvalidierung befindet sich im Status „Ausstehende Validierung“

- Auflösung

Wenn Sie mithilfe der DNS-Validierung ein öffentliches ACM-Zertifikat angefordert haben, müssen Sie den von ACM bereitgestellten CNAME-Eintrag zu Ihrer DNS-Konfiguration hinzufügen. Navigieren Sie zur ACM-Konsole, erweitern Sie die Zertifikatsdetails und fügen Sie den DNS-Validierungseintrag zum DNS-Anbieter Ihrer Domain hinzu. Wenn Sie die E-Mail-Validierung verwendet haben, überprüfen Sie die mit der Domain verknüpfte E-Mail-Adresse auf die Bestätigungs-E-Mail von AWS. Weitere Informationen finden Sie unter DNS-Validierung im AWS Certificate Manager Manager-Benutzerhandbuch.

Problem: Die Webkonsole gibt nach der Bereitstellung den Fehler „502 Bad Gateway“ oder „503 Service Temporary Unavailable“ zurück

- Auflösung

Überprüfen Sie die Integritätsprüfungen der ALB-Zielgruppe in der EC2-Konsole. Wenn die ECS Fargate-Aufgaben als fehlerhaft angezeigt werden, stellen Sie sicher, dass die Aufgaben in der ECS-Konsole ausgeführt werden, und überprüfen Sie, ob die Task-Anmeldung auf Fehler CloudWatch erfolgt. Stellen Sie sicher, dass die den ECS-Aufgaben zugeordnete Sicherheitsgruppe eingehenden Datenverkehr von der ALB-Sicherheitsgruppe zulässt.

Problem: DNS ist konfiguriert, aber die benutzerdefinierte Domain lässt sich nicht in der Webkonsole auflösen

- Auflösung

Stellen Sie sicher, dass der CNAME-Eintrag in Ihrem DNS-Anbieter korrekt konfiguriert ist, und ordnen Sie Ihre benutzerdefinierte Domain (z. B.`console.example.com`) dem ALB-DNS-Namen aus den CloudFormation Ausgaben zu. Die DNS-Weitergabe kann je nach Ihrem DNS-Anbieter und den TTL-Einstellungen bis zu 48 Stunden dauern. Sie können den DNS-Eintrag mit `dig console.example.com CNAME` oder `nslookup console.example.com` überprüfen.

Problem: Die Webkonsole gibt nach der Bereitstellung den Fehler „403 Forbidden“ zurück

- Auflösung

Die vor der ALB bereitgestellte AWS WAF WAF-Web-ACL blockiert möglicherweise legitime Anfragen. Öffnen Sie die AWS-WAF-Konsole, wählen Sie die mit der ALB verknüpfte Web-ACL aus und überprüfen Sie auf der Registerkarte Sampled Requests, welche Regel den Datenverkehr blockiert. Sie können die WAF-Regeln ändern, um die blockierten Anfragen zuzulassen. Wenn eine verwaltete Regelgruppe beispielsweise Falschmeldungen generiert, können Sie die spezifische Regelaktion auf Zählen statt auf Blockieren setzen, um den Datenverkehr zuzulassen und ihn trotzdem zu protokollieren. Weitere Informationen finden Sie unter Testen und Optimieren Ihres AWS-WAF-Schutzes im AWS WAF Developer Guide.

Probleme bei der Headless-Bereitstellung (bringen Sie Ihren eigenen Webserver mit)

Problem: Die Webkonsole zeigt CORS-Fehler an, wenn eine Verbindung zur API hergestellt wird

- Auflösung

Cross-Origin CORS-Fehler (Resource Sharing) treten auf, wenn die auf Ihrer Domain gehostete Webkonsole versucht, die API-Gateway-Endpunkte der Lösung aufzurufen. Stellen Sie sicher, dass Ihr Webserver die Konsole über HTTPS bedient, da die API-Gateway-Endpunkte HTTPS benötigen. Stellen Sie sicher, dass die Ursprungsdomäne Ihres Webservers mit den zulässigen Ursprüngen übereinstimmt, die in den CORS-Einstellungen des API Gateway konfiguriert sind. Wenn Sie eine benutzerdefinierte Domain verwenden, müssen Sie möglicherweise die CORS-Konfiguration von API Gateway aktualisieren, um Ihre Domain einzubeziehen.

Problem: Die Webkonsole wird geladen, aber die Authentifizierung schlägt fehl oder leitet falsch um

- Auflösung

Die Ressourcen der Webkonsole enthalten eine Konfigurationsdatei mit den Cognito-Benutzerpooleinstellungen und der API-Endpunkt-URL. Stellen Sie sicher, dass diese Konfigurationsdatei beim Extrahieren nicht geändert wurde. Stellen Sie sicher, dass Ihr Webserver die Konsole über HTTPS bedient, da Cognito HTTPS für Callback-URLs benötigt. Vergewissern Sie sich, dass die Callback-URL des Cognito-App-Clients die Domain Ihres Webservers enthält.

Kontaktieren Sie AWS Support.

Wenn Sie über AWS Business Support+, AWS Enterprise Support oder Unified Operations verfügen, können Sie das AWS Support Center nutzen, um fachkundige Support zu dieser Lösung zu erhalten. In den folgenden Abschnitten finden Sie entsprechende Anweisungen.

Fall erstellen

1. Melden Sie sich im Support Center an.
2. Wählen Sie Create case (Fall erstellen) aus.

Wie können wir helfen?

1. Wählen Sie Technisch
2. Wählen Sie für Service die Option Lösungen aus.
3. Wählen Sie für Kategorie die Option Distributed Load Testing on AWS aus.
4. Wählen Sie unter Schweregrad die Option aus, die Ihrem Anwendungsfall am besten entspricht.

5. Wenn Sie den Service, die Kategorie und den Schweregrad eingeben, werden in der Benutzeroberfläche Links zu häufig gestellten Fragen zur Fehlerbehebung angezeigt. Wenn Sie Ihre Fragen mit diesen Links nicht lösen können, wählen Sie Nächster Schritt: Zusätzliche Informationen.

Zusätzliche Informationen

1. Geben Sie als Betreff einen Text ein, der Ihre Frage oder Ihr Problem zusammenfasst.
2. Beschreiben Sie unter Beschreibung das Problem detailliert, einschließlich des Namens dieses Produkts und der Version, die Sie verwenden, z. B. in diesem Beispiel: Distributed Load Testing on AWS X.Y.Z v.
3. Wählen Sie Dateien anhängen.
4. Fügen Sie die Informationen bei, die der AWS-Support zur Bearbeitung der Anfrage benötigt.

Helfen Sie uns, Ihren Fall schneller zu lösen

1. Geben Sie die angeforderten Informationen ein.
2. Klicken Sie auf Next step: Solve now or contact us (()Nächster Schritt): Jetzt lösen oder Support kontaktieren).

Löse es jetzt oder kontaktiere uns

1. Sehen Sie sich die Solve Now-Lösungen an.
2. Wenn Sie Ihr Problem mit diesen Lösungen nicht lösen können, wählen Sie Kontakt, geben Sie die angeforderten Informationen ein und klicken Sie auf Absenden.

Aktualisieren Sie die Lösung

Bei der Aktualisierung der Lösung werden die neuesten Funktionen, Sicherheitspatches und Bugfixes auf Ihre Bereitstellung angewendet. Informationen zum Update auf die neueste Version finden Sie im entsprechenden Abschnitt, der auf Ihrer ursprünglichen Bereitstellungsmethode basiert: AWS Launch Wizard oder AWS CloudFormation.

> **Important**
>
> Stellen Sie vor dem Update sicher, dass derzeit keine Auslastungstests ausgeführt werden. Der Aktualisierungsprozess kann die Verfügbarkeit der Lösung vorübergehend beeinträchtigen.

Aktualisierung mit dem AWS Launch Wizard

Die Konsole zeigt automatisch die neueste verfügbare Version der Lösung in der Dropdownliste Bereitstellungsversion an. Wenn Sie die Lösung bereits bereitgestellt haben, gehen Sie wie folgt vor, um Ihre Bereitstellung auf die neueste Version zu aktualisieren.

1. Gehen Sie zu Launch Wizard Deployments.
2. Wählen Sie die Bereitstellung aus, die Sie aktualisieren möchten.
3. Wählen Sie Aktionen und dann Bereitstellungsversion aktualisieren aus.
4. Wählen Sie die neueste Version aus den verfügbaren Bereitstellungsversionen aus.
5. Überprüfen Sie die Konfiguration.
6. Nehmen Sie bei jedem Schritt die erforderlichen Änderungen vor.
7. Bestätigen Sie das Update.

Update mit AWS CloudFormation

Wenn Sie die Lösung bereits bereitgestellt haben, gehen Sie wie folgt vor, um den CloudFormation Stack auf die neueste Version zu aktualisieren.

1. Melden Sie sich bei der CloudFormation Konsole an, wählen Sie Ihren vorhandenen CloudFormation Stack aus und wählen Sie Stack aktualisieren aus.

2. Wählen Sie Direktes Update durchführen aus.
3. Wählen Sie Bestehende Vorlage ersetzen aus.
4. Gehen Sie unter Vorlage angeben wie folgt vor:
   a. Wählen Sie Amazon S3 S3-URL aus.
   b. Kopieren Sie den Link der neuesten Vorlage.
   c. Fügen Sie den Link in das Amazon S3 S3-URL-Feld ein.
   d. Vergewissern Sie sich, dass die richtige Vorlagen-URL im Textfeld Amazon S3 S3-URL angezeigt wird.
   e. Wählen Sie Weiter aus.
   f. Wählen Sie erneut Next (Weiter).
5. Überprüfen Sie unter Parameter die Parameter für die Vorlage und ändern Sie sie nach Bedarf. Einzelheiten zu den Parametern finden Sie unter Den Stack starten.
6. Wählen Sie Weiter aus.
7. Wählen Sie auf der Seite Configure stack options (Stack-Optionen konfigurieren) Next (Weiter) aus.
8. Überprüfen und bestätigen Sie die Einstellungen auf der Seite Review.
9. Markieren Sie das Kästchen, das bestätigt, dass die Vorlage möglicherweise IAM-Ressourcen erstellt.
10. Wählen Sie „Änderungssatz anzeigen“ und überprüfen Sie die Änderungen.
11. Wählen Sie Stack aktualisieren, um den Stack bereitzustellen.

Sie können den Status des Stacks in der CloudFormation AWS-Konsole in der Spalte Status anzeigen. Sie sollten in etwa 15 Minuten einen `UPDATE_COMPLETE` Status erhalten.

> **Note**
>
> Wenn bei der Anmeldung über Ihren Browser nach dem Stack-Upgrade Probleme mit der Amazon Cognito Cognito-Authentifizierung auftreten, aktualisieren Sie Ihren Browser (Strg +Shift+R auf Windows/Linux oder Cmd+Shift+R auf Mac), um die zwischengespeicherten Daten zu löschen, und versuchen Sie es erneut.

Fehlerbehebung bei Updates von Versionen vor v3.3.0

> **Note**
>
> Dieser Abschnitt gilt nur für Updates von Versionen vor v3.3.0. Wenn Sie von Version 3.3.0 oder höher aktualisieren, folgen Sie dem Standard-Aktualisierungsverfahren über den AWS Launch Wizard oder AWS. CloudFormation

1. Laden Sie die distributed-load-testing-on-aws.template herunter.
2. Öffnen Sie die Vorlage und navigieren Sie zu und suchen Sie nach. `Conditions: DLTCommonResourcesAppRegistryCondition`
3. Die Ausgabe sollte wie folgt aussehen:

```
Conditions:
DLTCommonResourcesAppRegistryConditionCCEF54F8:
Fn::Equals:
- "true"
- "true"
```

4. Ändern Sie den zweiten `true` Wert in`false`:

```
Conditions:
DLTCommonResourcesAppRegistryConditionCCEF54F8:
Fn::Equals:
- "true"
- "false"
```

5. Verwenden Sie die benutzerdefinierte Vorlage, um Ihren Stack zu aktualisieren, indem Sie die Schritte unter Update mit AWS ausführen CloudFormation.
6. Dieses Update entfernt Ressourcen, die sich auf die App-Registrierung beziehen, aus dem Stack, sodass das Update erfolgreich abgeschlossen werden kann.
7. Führen Sie ein weiteres Stack-Update mit der neuesten Vorlagen-URL durch.

Der Pfad der Zip-Datei ändert sich von Version 3 auf Version 4

DLT v4 hat die Art und Weise geändert, wie gezippte Testszenarien extrahiert werden, um sie an die Art und Weise anzupassen, wie die JMeter-GUI Dateipfade auflöst, und um ein konsistentes Verhalten in allen unterstützten Frameworks zu unterstützen.

Version	Verhalten bei der Extraktion
v3	Die Dateien wurden unter Beibehaltung der Verzeichnisstruktur entpackt. Non-JMX Dateien (CSV, Bilder) mussten über ihren Pfad innerhalb des Archivs referenziert werden (z. B.`my-test/users.csv`).
v4	Alle Dateien werden zusammen mit der JMX-Datei direkt in das Arbeitsverzeichnis des Containers extrahiert. Jede interne Verzeichnisstruktur in der Zip-Datei ist reduziert, sodass Dateiverweise in der JMX-Datei nur den Dateinamen verwenden dürfen (z. B.). `users.csv`

Wenn Ihre ZIP-Datei beispielsweise Folgendes enthält:

```
my-test.zip
### my-test/
#   ### sample.jmx
#   ### users.csv
```

In Version 3 referenzierte die CSV-Datensatzkonfiguration in der JMX die Datei anhand ihres Pfads im Archiv:`my-test/users.csv`. In Version 4 muss dieselbe Referenz „just`users.csv`, weil“ sein `sample.jmx` und `users.csv` sich zur Laufzeit im gleichen flachen Arbeitsverzeichnis befinden.

> **Note**
>
> Dies gilt für alle Dateiverweise in Ihren Testskripten, einschließlich der Konfigurationspfade für CSV-Datensätze, Datei-Uploads und aller anderen relativen Pfade.

Aktualisierung der regionalen Stacks

Wenn Sie die Lösung in mehreren Regionen eingesetzt haben, müssen Sie jeden regionalen Stack separat aktualisieren. Folgen Sie dem Standardaktualisierungsverfahren für jeden regionalen CloudFormation Stack in den Regionen, in denen Sie die Testinfrastruktur bereitgestellt haben.

AWS Systems Manager Manager-Anwendungsmanager

Nach der Aktualisierung der Lösung bietet AWS Systems Manager Application Manager einen Überblick über die Lösung und ihre Ressourcen auf Anwendungsebene. Sie können Application Manager verwenden, um:

- Überwachen Sie Ressourcen, Kosten für bereitgestellte Ressourcen in Stacks und AWS-Konten sowie Protokolle von einem zentralen Standort aus.
- Zeigen Sie Betriebsdaten für die Ressourcen der Lösung im Kontext einer Anwendung an, z. B. den Bereitstellungsstatus, CloudWatch Alarme, Ressourcenkonfigurationen und Betriebsprobleme.

Deinstalliere die Lösung

Sie können die Lösung Distributed Load Testing on AWS über die AWS-Managementkonsole oder mithilfe der AWS-Befehlszeilenschnittstelle (AWS CLI) deinstallieren. Sie müssen mehrere zurückbehaltene Ressourcen, die mit dieser Lösung erstellt wurden, manuell löschen. AWS-Lösungen löschen Speicher- und Protokollierungsressourcen nicht automatisch, falls Sie Daten aufbewahren müssen. Lesen Sie daher die folgenden Informationen zum Löschen von S3-Buckets, DynamoDB-Tabellen, CloudWatch Protokollgruppen und Dashboards. CloudWatch

> **Important**
>
> Stellen Sie vor der Deinstallation der Lösung sicher, dass zuerst alle regionalen Stacks gelöscht wurden. Wir empfehlen außerdem, alle Testszenarien über die DLT-Webkonsole zu löschen, bevor Sie den Hauptstapel löschen. Dies hilft, während der Laufzeit erstellte Ressourcen wie Dashboards zu bereinigen. CloudWatch

Verwendung der AWS-Managementkonsole

AWS CloudFormation

1. Melden Sie sich bei der CloudFormation -Konsole an.
2. Wählen Sie auf der Seite Stacks den Installations-Stack dieser Lösung aus.
3. Wählen Sie Löschen aus.

AWS-STARTASSISTENT

1. Melden Sie sich bei der AWS Launch Wizard Wizard-Konsole an.
2. Wählen Sie auf der Seite Launch Wizard Deployments die Bereitstellung dieser Lösung aus.
3. Wählen Sie Aktionen und dann Löschen aus.
4. Bestätigen Sie das Löschen.

Verwenden der AWS-Befehlszeilenschnittstelle

Stellen Sie fest, ob die AWS-Befehlszeilenschnittstelle (AWS CLI) in Ihrer Umgebung verfügbar ist. Installationsanweisungen finden Sie unter Was ist die AWS-Befehlszeilenschnittstelle im AWS-CLI-Benutzerhandbuch. Nachdem Sie bestätigt haben, dass die AWS-CLI verfügbar ist, führen Sie den folgenden Befehl aus.

```
$ aws cloudformation delete-stack --stack-name <installation-stack-name>
```

Manuelles Löschen einbehaltener Ressourcen

Nach dem Löschen des Stacks finden Sie in den folgenden Abschnitten Informationen zum Löschen der verbleibenden Ressourcen.

Löschen der Amazon S3 S3-Buckets

Diese Lösung ist so konfiguriert, dass die von der Lösung erstellten Amazon S3 S3-Buckets beibehalten werden, falls Sie den CloudFormation AWS-Stack löschen möchten, um einen versehentlichen Datenverlust zu verhindern. Nach der Deinstallation der Lösung können Sie diese S3-Buckets manuell löschen, wenn Sie die Daten nicht behalten müssen. Gehen Sie wie folgt vor, um die Amazon S3 S3-Buckets zu löschen.

1. Melden Sie sich bei der Amazon S3-Konsole an.
2. Wählen Sie im Navigationsbereich die Option Buckets aus.
3. Suchen Sie die S3-Buckets, die von der Lösung erstellt wurden.
4. Wählen Sie einen S3-Bucket aus und wählen Sie Leer.
5. Geben Sie im Bestätigungsfeld den Text Dauerhaft löschen ein und wählen Sie Leer aus.
6. Wählen Sie den S3-Bucket aus, den Sie gerade geleert haben, und wählen Sie Löschen.
7. Geben Sie den S3-Bucket-Namen in das Bestätigungsfeld ein und wählen Sie Bucket löschen.

Wiederholen Sie die Schritte, bis Sie alle S3-Lösungs-Buckets gelöscht haben.

Um die S3-Buckets mit der AWS-CLI zu löschen, führen Sie den folgenden Befehl für jeden Bucket aus:

```
$ aws s3 rb s3://<bucket-name> --force
```

Löschen der Amazon DynamoDB-Tabellen

Diese Lösung ist so konfiguriert, dass die von der Lösung erstellten DynamoDB-Tabellen beibehalten werden, falls Sie den CloudFormation AWS-Stack löschen möchten, um einen versehentlichen Datenverlust zu verhindern. Nach der Deinstallation der Lösung können Sie diese Tabellen manuell löschen, wenn Sie die Daten nicht behalten müssen. Gehen Sie wie folgt vor, um die DynamoDB-Tabellen zu löschen.

1. Melden Sie sich bei der Amazon DynamoDB-Konsole an.
2. Wählen Sie im Navigationsbereich Tables (Tabellen) aus.
3. Suchen Sie die von der Lösung erstellten Tabellen.
4. Wählen Sie eine Tabelle aus und klicken Sie auf Löschen.
5. Bestätigen Sie den Löschvorgang, indem Sie Löschen eingeben und Tabelle löschen wählen.

Wiederholen Sie die Schritte, bis Sie alle Lösungstabellen gelöscht haben.

Führen Sie den folgenden Befehl aus, um die DynamoDB-Tabellen mit der AWS-CLI zu löschen:

```
$ aws dynamodb delete-table --table-name <table-name>
```

Löschen der Protokollgruppen CloudWatch

Diese Lösung behält die CloudWatch Protokollgruppen bei, wenn Sie den CloudFormation AWS-Stack löschen möchten, um einen versehentlichen Datenverlust zu verhindern. Nach der Deinstallation der Lösung können Sie die Protokollgruppen manuell löschen, wenn Sie die Daten nicht behalten müssen. Gehen Sie wie folgt vor, um die CloudWatch Protokollgruppen zu löschen.

1. Melden Sie sich bei der CloudWatch Amazon-Konsole an.
2. Wählen Sie im Navigationsbereich Protokollgruppen aus.
3. Suchen Sie die von der Lösung erstellten Protokollgruppen.
4. Wählen Sie die Protokollgruppen aus, die Sie löschen möchten.
5. Wählen Sie Actions und dann Delete aus.

Wiederholen Sie die Schritte, bis Sie alle Lösungsprotokollgruppen gelöscht haben.

Führen Sie die folgenden Befehle aus, um Protokollgruppen mit der AWS-CLI zu löschen:

```
$ aws logs describe-log-groups --log-group-name-prefix <stack-name> --query
 'logGroups[].logGroupName' --output text
$ aws logs delete-log-group --log-group-name <log-group-name>
```

> **Note**
>
> Wenn Sie regionale Stacks bereitgestellt haben, wiederholen Sie diesen Vorgang in jeder Region, in der ein regionaler Stack bereitgestellt wurde.

Löschen der Dashboards CloudWatch

Diese Lösung erstellt ein CloudWatch Dashboard für jedes Testszenario in jeder Region, in der der Test ausgeführt wird. Diese Dashboards folgen dem Benennungsmuster `EcsLoadTesting-<testId>-<region>` und es fallen monatliche Gebühren an, solange sie existieren. Sie werden bereinigt, wenn Sie einen Test über die DLT-Konsole löschen. Sie bleiben jedoch erhalten, wenn der Stack gelöscht wird, ohne zuerst alle Tests zu löschen.

Nach der Deinstallation der Lösung können Sie diese Dashboards manuell löschen, sofern sie nicht bereits bereinigt wurden. Gehen Sie wie folgt vor, um die Dashboards zu löschen. CloudWatch

1. Melden Sie sich bei der CloudWatch Amazon-Konsole an.
2. Wählen Sie im Navigationsbereich Dashboards aus.
3. Geben Sie in das Filterfeld ein, `EcsLoadTesting` um die Dashboards der Lösung zu finden.
4. Wählen Sie die Dashboards aus, die Sie löschen möchten.
5. Wählen Sie Löschen aus.

Führen Sie die folgenden Befehle aus, um Dashboards mit der AWS-CLI zu löschen:

```
$ aws cloudwatch list-dashboards --dashboard-name-prefix EcsLoadTesting --query
 'DashboardEntries[].DashboardName' --output text
$ aws cloudwatch delete-dashboards --dashboard-names <dashboard-name-1> <dashboard-
name-2>
```

Säuberung der Ressourcen von ALB + ECS Fargate nach der Bereitstellung

Wenn Sie die Bereitstellungsoption ALB + ECS Fargate gewählt haben, löschen Sie den CNAME- oder Route 53-Aliaseintrag, der Ihre benutzerdefinierte Domain dem ALB-DNS-Namen in Ihrem DNS-Anbieter zuordnet. Wenn das ACM-Zertifikat nicht mehr benötigt wird, können Sie es auch aus der ACM-Konsole löschen. Die vor dem ALB bereitgestellte AWS WAF WAF-Web-ACL wird automatisch gelöscht, wenn der CloudFormation Stack gelöscht wird.

Wiederherstellung von DLT-Daten nach dem Löschen des Stacks

In diesem Abschnitt wird beschrieben, wie Sie Testszenarien wiederherstellen und den Verlauf in einem neuen DLT-Stack ausführen, nachdem der ursprüngliche Stack gelöscht wurde. Dies gilt, wenn die S3-Buckets und DynamoDB-Tabellen des Kunden beibehalten wurden (das Standardverhalten) und der Kunde sie erneut mit einer neuen Bereitstellung verbinden möchte.

Hintergrund

Wenn ein DLT-Stack gelöscht wird, bleiben die folgenden Ressourcen aufgrund von Datenschutzrichtlinien im AWS-Konto des Kunden erhalten:

Ressource	Enthält	Benennung
Szenarien: S3-Bucket	Testskripte (JMeter/k6/Locust), XML-Ergebnisdateien pro Aufgabe, JMeter-Framework-Assets	Auto-generated (zum Beispiel) `dltstack-scenarios bucket-abc123`
Protokolliert den S3-Bucket	S3-Zugriffsprotokolle für alle DLT-Buckets	Auto-generated
S3-Bucket für die Konsole	Statische Ressourcen der Web-Benutzeroberfläche	Auto-generated
Szenarien DynamoDB-Tabelle	Testdefinitionen, Konfigurationen, Planungsregeln	Auto-generated (zum Beispiel`distributed-load-testing-on-`

Ressource	Enthält	Benennung
		`aws-DLTTestRunner StorageDLTScenario sTableXXXXXXXX-XXX XXXXXXXXX`)
DynamoDB-Tabelle mit Historie	Ergebnisse, Dauer und Erfolgsquoten des Testlaufs	Auto-generated (zum Beispiel) `distributed-load-t esting-on-aws-DLTT estRunnerStorageDL THistoryTableXXXXX XXX-XXXXXXXXXXX`
DynamoDB-Tabelle für Agent Spaces	Registered Agent Space-Verbindungen, die von der DevOps AWS-Agent-Integrat ion verwendet werden	Auto-generated (zum Beispiel`distributed-load-testing-on-aws-DevOpsAgentDL TAgentSpacesTableX XXXXXXX-XXXXXXXXX XX`)
DynamoDB-Tabelle für Untersuchungen	Ergebnisse von KI-Unters uchungen und Erkenntni sse im Zusammenhang mit Testläufen	Auto-generated (zum Beispiel`distributed-load-testing-on-aws-DevOpsAgentDL TInvestigationsTab leXXXXXXX-XXXXXX XXXXX`)
CloudWatch Gruppen protokoll ieren	Lambda- und ECS-Ausfü hrungsprotokolle (10 Jahre Aufbewahrung)	Benannt nach Funktion

Alle Ressourcennamen werden automatisch von CloudFormation generiert. Sie sind nicht vorhersehbar und können bei der Bereitstellung eines neuen Stacks nicht als Parameter angegeben werden.

Wichtige Einschränkungen

1. Die CloudFormation DLT-Vorlage akzeptiert keine Parameter für bestehende DynamoDB-Tabellen oder S3-Buckets. Eine neue Bereitstellung erzeugt immer neue Ressourcen.
2. Das direkte Ändern von CloudFormation-managed Ressourcen außerhalb des Stacks führt zu einer Stack-Drift. Die gesamte Datenmigration muss auf Datenebene erfolgen, nicht auf Infrastrukturebene.
3. Die Lösungs-UUID wird bei jeder neuen Bereitstellung nach dem Zufallsprinzip generiert. Der neue Stack wird eine andere UUID als das Original haben.
4. In DynamoDB-Tabellen ist Point-in-Time Recovery (PITR) standardmäßig aktiviert, sodass kontinuierliche Backups für die letzten 35 Tage bereitgestellt werden.

Wiederherstellungsprozess

Schritt 1: Identifizieren Sie die zurückgehaltenen Ressourcen

Bevor Sie einen neuen Stack bereitstellen, suchen Sie die zurückgehaltenen Ressourcen aus dem gelöschten Stack.

```
# List retained DynamoDB tables (look for DLT naming patterns)
aws dynamodb list-tables --query "TableNames[?contains(@, 'ScenariosTable') ||
 contains(@, 'HistoryTable') || contains(@, 'AgentSpacesTable') || contains(@,
 'InvestigationsTable')]"

# List retained S3 buckets (look for DLT naming patterns)
aws s3api list-buckets --query "Buckets[?contains(Name, 'dlttestrunnerstoragedlts') ||
 contains(Name, 'dltconsoleresourcesdltda')]"
```

Wenn Sie sich die CloudFormation Stack-Ausgaben vor dem Löschen notiert haben, verwenden Sie genau diese Namen. Die relevanten Ausgaben waren:

- `ScenariosBucket`— der Name des S3-Buckets
- `ScenariosTable`— der Name der Tabelle für DynamoDB-Szenarien

Verwenden Sie für Ressourcen, die nicht als Stack-Ausgaben verfügbar gemacht werden (z. B. die Verlaufstabelle), `list-stack-resources` vor dem Löschen, um die physischen Ressourcen-

IDs abzurufen. Dieser Befehl listet alle Ressourcen im Stapel auf, unabhängig davon, ob sie eine entsprechende Ausgabe hat.

Schritt 2: Stellen Sie einen neuen DLT-Stack bereit

Stellen Sie einen neuen DLT-Stack mit derselben CloudFormation Vorlagenversion (oder neuer) bereit. Verwenden Sie dieselben Parameter wie bei der ursprünglichen Bereitstellung (VPC-Einstellungen, Administrator-E-Mail usw.).

```
aws cloudformation create-stack \
  --stack-name distributed-load-testing-new \
  --template-url https://s3.amazonaws.com/solutions-reference/distributed-load-testing-
on-aws/latest/distributed-load-testing-on-aws.template \
  --parameters ParameterKey=AdminName,ParameterValue=<admin-name> \
               ParameterKey=AdminEmail,ParameterValue=<admin-email> \
  --capabilities CAPABILITY_IAM CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND
```

Warten Sie, bis der Stack den `CREATE_COMPLETE` Status erreicht hat.

Schritt 3: Identifizieren Sie die Ressourcen des neuen Stacks

Rufen Sie die Ressourcennamen aus den Ausgaben des neuen Stacks ab.

```
aws cloudformation describe-stacks \
  --stack-name distributed-load-testing-new \
  --query "Stacks[0].Outputs"
```

Notieren Sie sich die Werte für:

- `ScenariosBucket`(neuer Bucket-Name)
- `ScenariosTable`(neuer Tabellenname)

Verwenden Sie für jede Ressource, die nicht als Stack-Ausgabe verfügbar gemacht wird (z. B. die Tabellen History, Agent Spaces und Investigations), `list-stack-resources` um sie anhand ihrer logischen ID zu finden:

```
aws cloudformation list-stack-resources \
  --stack-name distributed-load-testing-new \
```

```
  --query "StackResourceSummaries[?contains(LogicalResourceId, 'HistoryTable') ||
contains(LogicalResourceId, 'AgentSpacesTable') || contains(LogicalResourceId,
'InvestigationsTable')].{LogicalId:LogicalResourceId,PhysicalId:PhysicalResourceId}"
```

Dieser Befehl listet alle Ressourcen auf, die der Stack verwaltet. Sie können nach einer logischen ID-Teilzeichenfolge filtern, um jede Ressource zu finden.

Schritt 4: DynamoDB-Daten migrieren

Exportieren Sie Daten aus den beibehaltenen Tabellen und importieren Sie sie in die neuen Tabellen. Verwenden Sie DynamoDB-Datenexport- oder Scan- und Schreiboperationen.

Option A: Verwendung von DynamoDB Export/Import (empfohlen für große Datensätze)

Verwenden Sie für Tabellen mit Tausenden von Elementen einen DynamoDB-Export nach S3, gefolgt von einem Import:

```
# Export from retained table to S3
aws dynamodb export-table-to-point-in-time \
  --table-arn arn:aws:dynamodb:<region>:<account>:table/<old-scenarios-table> \
  --s3-bucket <temporary-export-bucket> \
  --s3-prefix dlt-migration/scenarios/ \
  --export-format DYNAMODB_JSON

# Wait for export to complete, then import into new table
aws dynamodb import-table \
  --s3-bucket-source S3Bucket=<temporary-export-bucket>,S3KeyPrefix=dlt-migration/
scenarios/ \
  --table-creation-parameters '{"TableName":"<new-scenarios-table>","KeySchema":
[{"AttributeName":"testId","KeyType":"HASH"}],"AttributeDefinitions":
[{"AttributeName":"testId","AttributeType":"S"}],"BillingMode":"PAY_PER_REQUEST"}' \
  --input-format DYNAMODB_JSON
```

> **Note**
>
> Die `import-table` API erstellt eine neue Tabelle. Da der neue Stack die Zieltabelle bereits erstellt hat, verwenden Sie stattdessen Option B oder löschen Sie zuerst die neue leere Tabelle und erstellen Sie sie per Import neu (dies führt zu einer Stack-Drift, siehe die Warnung unten).

Option B: Scannen und BatchWrite (empfohlen)

Dieser Ansatz liest direkt aus den beibehaltenen Tabellen und schreibt in die Tabellen des neuen Stacks, ohne dass es zu einer Stack-Drift kommt. Es erfordert Python 3 und die `boto3` Bibliothek.

```
# Install boto3 if not already available
pip install boto3
```

```
#!/usr/bin/env python3
"""Migrate DynamoDB items directly from retained tables to new stack tables."""

import boto3

dynamodb = boto3.resource("dynamodb", region_name="<region>")

def migrate_table(source_table_name, target_table_name):
    source = dynamodb.Table(source_table_name)
    target = dynamodb.Table(target_table_name)

    # Scan all items from the retained table (handles pagination automatically)
    items = []
    response = source.scan()
    items.extend(response["Items"])
    while "LastEvaluatedKey" in response:
        response = source.scan(ExclusiveStartKey=response["LastEvaluatedKey"])
        items.extend(response["Items"])

    print(f"Scanned {len(items)} items from {source_table_name}")

    # Write items to the new table using batch_writer (handles batching automatically)
    with target.batch_writer() as batch:
        for item in items:
            batch.put_item(Item=item)

    print(f"Wrote {len(items)} items to {target_table_name}")

# Migrate scenarios
migrate_table("<old-scenarios-table>", "<new-scenarios-table>")

# Migrate history
migrate_table("<old-history-table>", "<new-history-table>")

# Migrate Agent Space registrations
```

```
migrate_table("<old-agent-spaces-table>", "<new-agent-spaces-table>")

# Migrate investigation results
migrate_table("<old-investigations-table>", "<new-investigations-table>")
```

Bei großen Tabellen (Zehntausende von Elementen) sollten Sie parallel Scansegmente verwenden, um die Lesephase zu beschleunigen.

Option C: Aus PITR wiederherstellen (falls innerhalb eines Zeitfensters von 35 Tagen)

Wenn der Stapel innerhalb der letzten 35 Tage gelöscht wurde und PITR aktiviert war (dies ist standardmäßig der Fall), können Sie die Tabelle zu einem bestimmten Zeitpunkt wiederherstellen. Bei PITR-Wiederherstellungen wird jedoch eine neue Tabelle mit einem anderen Namen erstellt, sodass Sie immer noch Daten mit Option B in die vom Stack verwaltete Tabelle kopieren müssten.

```
# Restore to a new temporary table
aws dynamodb restore-table-to-point-in-time \
  --source-table-name <old-scenarios-table> \
  --target-table-name dlt-scenarios-restored \
  --restore-date-time <timestamp-before-deletion>

# Then scan from dlt-scenarios-restored and batch-write into the new stack's table
```

Schritt 5: S3-Daten migrieren

Kopieren Sie Testskripte und Ergebnisdateien aus dem beibehaltenen Bucket in den Bucket des neuen Stacks.

```
# Copy all objects from the old scenarios bucket to the new one
aws s3 sync \
  s3://<old-scenarios-bucket> \
  s3://<new-scenarios-bucket> \
  --source-region <region> \
  --region <region>
```

Dies kopiert:

- Testskripte (`.jmx`JMeter-Dateien, K6-Skripte, Locust-Dateien, ZIP-Archive)
- XML-Dateien für Testergebnisse, organisiert nach Test-ID und Region
- Ressourcen und Plugins des JMeter-Frameworks

Schritt 6: Überprüfen Sie die Migration

1. Melden Sie sich mit der URL des neuen Stacks (in der Stack-Ausgabe) bei der `WebConsole` DLT-Webkonsole an.
2. Vergewissern Sie sich, dass alle Testszenarien in der Szenarioliste angezeigt werden.
3. Öffnen Sie einzelne Szenarien und stellen Sie sicher, dass der Testlaufverlauf sichtbar ist.
4. Vergewissern Sie sich, dass Testskripte von der Szenario-Detailseite heruntergeladen werden können.
5. Führen Sie einen kleinen Test durch, um die End-to-End-Funktionalität mit dem neuen Stack zu überprüfen.

Schritt 7: Bereinigen Sie die zurückgehaltenen Ressourcen

Nachdem Sie überprüft haben, ob alle Daten erfolgreich migriert wurden, löschen Sie die alten zurückgehaltenen Ressourcen, um laufende Speicherkosten zu vermeiden.

```
# Delete old DynamoDB tables
aws dynamodb delete-table --table-name <old-scenarios-table>
aws dynamodb delete-table --table-name <old-history-table>
aws dynamodb delete-table --table-name <old-agent-spaces-table>
aws dynamodb delete-table --table-name <old-investigations-table>

# Empty and delete old S3 buckets
aws s3 rm s3://<old-scenarios-bucket> --recursive
aws s3api delete-bucket --bucket <old-scenarios-bucket>

aws s3 rm s3://<old-logs-bucket> --recursive
aws s3api delete-bucket --bucket <old-logs-bucket>

aws s3 rm s3://<old-console-bucket> --recursive
aws s3api delete-bucket --bucket <old-console-bucket>
```

Führen Sie diesen Schritt erst aus, nachdem Sie bestätigt haben, dass der neue Stack voll funktionsfähig ist und alle historischen Daten intakt sind.

Vermeidung von Stack-Drift

Stack-Drift tritt auf, wenn der tatsächliche Zustand einer Ressource von den Erwartungen CloudFormation abweicht. Gehen Sie wie folgt vor, um Abweichungen während dieses Wiederherstellungsprozesses zu vermeiden:

- Benennen Sie die DynamoDB-Tabellen oder S3-Buckets des neuen Stacks NICHT direkt um oder ändern Sie sie.
- Verwenden Sie es NICHT`import-table`, um die Tabellen des neuen Stacks zu ersetzen (dazu müsste zuerst die CloudFormation-managed Tabelle gelöscht werden).
- Ändern Sie KEINE IAM-Richtlinien, Bucket-Richtlinien oder Tabelleneinstellungen außerhalb von CloudFormation.
- Verwenden Sie nur Operationen auf Datenebene:`PutItem`,,`BatchWriteItem`,`s3 sync`. `s3 cp`
- Schreiben Sie Daten in die Tabellen und Buckets, die CloudFormation erstellt wurden. Das Hinzufügen von Elementen zu einer DynamoDB-Tabelle oder von Objekten zu einem S3-Bucket stellt keine Drift dar.

Das Schreiben von Daten in CloudFormation-managed Tabellen und Buckets ist sicher, da die Ressourcenkonfiguration (Tabellenschema, Abrechnungsmodus, Verschlüsselungseinstellungen) und nicht der Dateninhalt CloudFormation verfolgt wird.

Was kann nicht wiederhergestellt werden

Folgendes geht verloren, wenn ein Stapel gelöscht wird und kann durch diesen Vorgang nicht wiederhergestellt werden:

Item	Grund
Cognito-Benutzerkonten	Der Benutzerpool wird zusammen mit dem Stack gelöscht. Benutzer müssen sich erneut registrieren.
Aktive Zeitpläne EventBridge	Geplante Testregeln werden gelöscht. Erstellen Sie Zeitpläne manuell in der DLT-Benutzeroberfläche neu.
UUID der Lösung	Eine neue zufällige UUID wird generiert. Dies wirkt sich nur auf die Korrelation der Betriebsmetriken aus.

Item	Grund
Regionale Stack-Verbände	Regionale Stacks müssen neu verteilt und dem neuen Hub-Stack wieder zugeordnet werden.

Präventive Maßnahmen

Um future Wiederherstellungsszenarien zu vereinfachen:

1. Zeichnen Sie die Stack-Ausgaben auf, bevor Sie sie löschen. Speichern Sie die `ScenariosBucket` Namen der `ScenariosTable` Verlaufstabellen, und.
2. Aktivieren Sie DynamoDB PITR (standardmäßig in DLT aktiviert). Stellen Sie sicher, dass es aktiv bleibt.
3. Aktivieren Sie die S3-Versionierung (standardmäßig im Szenario-Bucket aktiviert). Dies schützt vor versehentlichem Löschen von Objekten.
4. Erwägen Sie, den DynamoDB-Löschschutz für die Szenarien und Verlaufstabellen als zusätzlichen Schutz zu aktivieren:

```
aws dynamodb update-table \
  --table-name <scenarios-table> \
  --deletion-protection-enabled
```

5. Verwenden Sie AWS Backup, um geplante Backups von DynamoDB-Tabellen für die langfristige Aufbewahrung über das 35-tägige PITR-Fenster hinaus zu erstellen.

Entwicklerhandbuch

Dieser Abschnitt enthält den Quellcode für die Lösung und weitere Anpassungen.

Quellcode

Besuchen Sie unser GitHub Repository, um die Vorlagen und Skripte für diese Lösung herunterzuladen und Ihre Anpassungen mit anderen zu teilen.

Wartung

Diese Lösung verwendet Docker-Images mit festen Versionen, die jeder Lösungsversion entsprechen. Standardmäßig sind automatische Updates deaktiviert, sodass Sie die vollständige Kontrolle darüber haben, wann und welche Versionsupdates auf Ihre Bereitstellung angewendet werden. Das AWS-Lösungsteam verwendet Amazon ECR Enhanced Scanning, um Common Vulnerabilities and Exposures (CVEs) im Basis-Image und den installierten Paketen zu erkennen. Wenn möglich, veröffentlicht das Team gepatchte Images mit demselben Versions-Tag, um CVEs zu beheben, ohne die Kompatibilität mit der veröffentlichten Lösungsversion zu beeinträchtigen.

Wenn Bilder auf derselben Nebenversion gepatcht werden, wird das Stable-Tag automatisch aktualisiert und ein zusätzliches Image-Tag wird in diesem Format erstellt. `<solution-version>_<date-of-fix>` Wenn eine Haupt- oder Nebenversion veröffentlicht wird, müssen Sie ein Full-Stack-Update durchführen, um die neueste Image-Version zu erhalten, da das Stable-Tag entsprechend der Lösungsversion inkrementiert wird.

Wenn Sie sich während der Bereitstellung für automatische Updates entscheiden, werden Änderungen am Image, einschließlich CVE-Patches und kleinerer Bugfixes, automatisch bis zur neuesten passenden Nebenversion übernommen.

Versionen

Standardmäßig wird diese Lösung mit deaktivierten automatischen Updates bereitgestellt. Das bedeutet, dass die Container-Image-Version an die spezifische Version gebunden ist, die Ihrer bereitgestellten Lösungsversion entspricht, sodass Sie die volle Kontrolle über Versionsupdates haben.

Wenn Sie sich dafür entscheiden, automatische Updates zu aktivieren, indem Sie während der CloudFormation Bereitstellung Ja auswählen, erhält die Lösung automatisch Sicherheitspatches und kleinere, unveränderliche Bugfixes bis zur neuesten passenden Nebenversion. Wenn Sie beispielsweise Version 4.0.0 mit aktivierten automatischen Updates bereitstellen, erhalten Sie Updates bis 4.0.x, aber nicht 4.1.0 oder höher.

Um die Container-Image-Version manuell zu steuern, können Sie die Aufgabendefinition bearbeiten, um eine bestimmte Image-Version im Tag-Versionsformat anzugeben. Auf diese Weise können Sie unabhängig von der Einstellung für automatische Updates an eine bestimmte Image-Version anheften.

Anpassung des Container-Images

Tester-Image laden

Diese Lösung verwendet ein öffentliches Amazon Elastic Container Registry (Amazon ECR) -Image-Repository, das von AWS verwaltet wird, um das Image zu speichern, das für die Ausführung der konfigurierten Tests verwendet wird. Sie können das Container-Image neu erstellen und in ein ECR-Image-Repository in Ihrem eigenen AWS-Konto übertragen.

Sie können das Standard-Container-Image an Ihre Bedürfnisse anpassen. Legen Sie diese Umgebungsvariablen fest, bevor Sie ein benutzerdefiniertes Image erstellen.

```
#!/bin/bash
export REGION=aws-region-code # the AWS region to launch the solution (e.g. us-east-1)
export BUCKET_PREFIX=my-bucket-name # prefix of the bucket name without the region code
export BUCKET_NAME=$BUCKET_PREFIX-$REGION # full bucket name where the code will reside
export SOLUTION_NAME=my-solution-name
export VERSION=my-version # version number for the customized code
export PUBLIC_ECR_REGISTRY=public.ecr.aws/awssolutions/distributed-load-testing-on-
aws-load-tester # replace with the container registry and image if you want to use a
 different container image export PUBLIC_ECR_TAG=v3.1.0 # replace with the container
 image tag if you want to use a different container image
```

Sie können ein benutzerdefiniertes Container-Image entweder in einem privaten oder öffentlichen Image-Repository in Ihrem AWS-Konto hosten. Die Image-Ressourcen befinden sich im `deployment/ecr/distributed-load-testing-on-aws-load-tester` Verzeichnis in der Codebasis.

Anschließend können Sie das Image erstellen und an das Host-Ziel übertragen.

- Informationen zu privaten Amazon ECR-Repositorys und -Images finden Sie unter Private Amazon ECR-Repositorys und private Images im Amazon ECR-Benutzerhandbuch.
- Informationen zu öffentlichen Amazon ECR-Repositorys und -Images finden Sie unter Öffentliche Amazon ECR-Repositorys und öffentliche Images im Amazon ECR Public User Guide.

Nachdem Sie das benutzerdefinierte Image erstellt haben, deklarieren Sie die folgenden Umgebungsvariablen, bevor Sie die benutzerdefinierte Lösung erstellen.

```
#!/bin/bash
export PUBLIC_ECR_REGISTRY=YOUR_ECR_REGISTRY_URI # e.g. YOUR_ACCOUNT_ID.dkr.ecr.us-
east-1.amazonaws.com/YOUR_IMAGE_NAME
export PUBLIC_ECR_TAG=YOUR_ECR_TAG # e.g. latest, v3.4.0
```

Das Dockerfile definiert das Lasttest-Image. Es baut auf einem Amazon Linux-Basisimage auf, installiert die Sprachlaufzeiten, Systempakete und Testtools, die für die Ausführung von Lasttests erforderlich sind, fügt die Metadatendateien der Framework-Version und das `load-test.sh` Entrypoint-Skript hinzu und wird als Nicht-Root-Benutzer ausgeführt. Informationen zum Anzeigen des Dockerfiles finden Sie im Repository unter Dockerfile. GitHub

Das `load-test.sh` Entrypoint-Skript wird bei jedem Start einer Aufgabe innerhalb des Containers ausgeführt. Es liest die Testkonfiguration und die Container-Metadaten. Anschließend lädt es die Testdefinition von Amazon S3 herunter und stellt das ausgewählte Test-Framework für den Lauf bereit. Das Skript wartet auf ein Startsignal, sodass alle Aufgaben in einem Test gleichzeitig beginnen. Anschließend führt es den Test für die konfigurierte Dauer aus und erstellt dabei Metriken und Protokolle. Wenn der Test abgeschlossen ist, sammelt das Skript die Ergebnisse und reichert sie mit Details über den Container an, der sie erstellt hat. Es lädt die Ergebnisse, Protokolle und andere Artefakte auf Amazon S3 hoch. Das Skript verarbeitet auch Abbruchsignale, sodass es Teilergebnisse beibehält und protokolliert, wenn Sie einen Test abbrechen, bevor er abgeschlossen ist. Das vollständige Skript finden Sie im GitHub Repository unter load-test.sh.

Bild der Webkonsole (nur ALB + ECS Fargate-Vorlage)

Die ALB + ECS Fargate-Vorlage führt die Webkonsole als Container auf ECS Fargate aus. Standardmäßig ruft die Lösung das Web-Konsolen-Image aus einem öffentlichen Amazon ECR-Repository ab, das von AWS verwaltet wird.

In Umgebungen, in denen der Zugriff auf öffentliche Container-Registries eingeschränkt ist (z. B. VPCs ohne Internetzugang oder Konten mit ECR-Richtlinien für den öffentlichen Zugriff), müssen Sie

das öffentliche Image in ein privates Amazon ECR-Repository spiegeln und den privaten Image-URI im Parameter Image URI der Web-Konsole angeben. CloudFormation Sie können auch die folgenden Schritte verwenden, um das Web-Konsolen-Image anzupassen.

Spiegeln Sie das öffentliche Image in ein privates ECR-Repository

Um das Bild der Webkonsole in ein privates ECR-Repository zu spiegeln:

1. Authentifizieren Sie sich bei der öffentlichen ECR-Registrierung:

```
$ aws ecr-public get-login-password --region us-east-1 | docker login --username AWS
 --password-stdin public.ecr.aws
```

2. Rufen Sie das Bild der öffentlichen Webkonsole ab:

```
$ docker pull public.ecr.aws/aws-solutions/distributed-load-testing-on-aws-web-
console:<version>
```

3. Erstellen Sie ein privates ECR-Repository in Ihrem Konto (falls noch keines vorhanden ist):

```
$ aws ecr create-repository --repository-name <your-repo-name> --region <region> 2>/
dev/null || true
```

4. Authentifizieren Sie sich bei Ihrer privaten ECR-Registrierung:

```
$ aws ecr get-login-password --region <region> | docker login --username AWS --
password-stdin <account-id>.dkr.ecr.<region>.amazonaws.com
```

5. Taggen Sie das Bild und übertragen Sie es in Ihr privates Repository:

```
$ docker tag public.ecr.aws/aws-solutions/distributed-load-testing-on-aws-web-
console:<version> \
  <account-id>.dkr.ecr.<region>.amazonaws.com/<your-repo-name>:<version>
$ docker push <account-id>.dkr.ecr.<region>.amazonaws.com/<your-repo-name>:<version>
```

6. Geben Sie beim Starten des ALB + ECS Fargate-Stacks den privaten Image-URI in den Image-URI-Parameter der Web-Konsole ein:

```
<account-id>.dkr.ecr.<region>.amazonaws.com/<your-repo-name>:<version>
```

Passen Sie das Bild der Webkonsole an

Sie können das Bild der Webkonsole auch anpassen. Die Bildressourcen befinden sich im `deployment/ecr/distributed-load-testing-on-aws-web-console` Verzeichnis, das sich im GitHub Repository befindet. Sie können die Containerdateien bearbeiten, ein lokales Image erstellen, es in ein privates ECR-Repository übertragen und den resultierenden Bild-URI im Image-URI-Parameter der Web-Konsole angeben.

Das Container-Image der Webkonsole baut auf einem Nginx-Basisimage auf und installiert die AWS-CLI. Sein Dockerfile kopiert die Nginx-Konfiguration und ein Einstiegspunktskript. Das Dockerfile macht den HTTP-Port verfügbar und legt fest, dass das Entrypoint-Skript beim Start des Containers ausgeführt wird. Wenn der Container gestartet wird, verwendet entrypoint.sh die AWS-CLI, um die Webkonsolen-Assets von Amazon S3 herunterzuladen. Es extrahiert sie in den Nginx-Dokumentenstamm und startet dann Nginx im Vordergrund. Die Datei nginx.conf dient der Einzelseitenanwendung (SPA) und leitet nicht übereinstimmende Pfade zurück zum Einstiegspunkt der Anwendung. Es fügt außerdem einen Health Check-Endpunkt für den Application Load Balancer hinzu, aktiviert die Gzip-Komprimierung, legt langlebiges Caching für statische Assets fest (während das Caching für den Einstiegspunkt und die Laufzeitkonfiguration deaktiviert wird) und fügt allgemeine Sicherheitsantwort-Header hinzu.

Weitere Informationen finden Sie unter Ein Docker-Image in ein privates Amazon ECR-Repository verschieben im Amazon ECR-Benutzerhandbuch.

Tester-Image-URI laden

Alle drei Bereitstellungsvorlagen (Amazon CloudFront + S3, Application Load Balancer (ALB) + ECS auf AWS Fargate und Headless) rufen zum Zeitpunkt der Bereitstellung ein Loadtester-Container-Image aus der öffentlichen Registrierung (Amazon ECR) von Amazon Elastic Container Registry (Amazon ECR) ab (`public.ecr.aws`).

Spiegeln Sie in Umgebungen, in denen der Zugriff auf öffentliche Container-Registrierungen eingeschränkt ist, das Loadtester-Image in ein privates Amazon ECR-Repository und geben Sie die private Image-URI im CloudFormation AWS-Parameter Load Tester Image URI (`LoadTesterImageUri`) an. Zu den häufigsten Beispielen gehören VPCs ohne Internetzugang, Konten mit ECR-Richtlinien für den öffentlichen Zugriff und AWS-Regionen GovCloud (USA), in denen ECS-Aufgaben nicht auf das öffentliche ECR-Register zugreifen können. Die Spiegelung in ein privates Repository in Ihrer Bereitstellungsregion ist für AWS-Bereitstellungen GovCloud (USA) und optional in anderen Ländern erforderlich.

Spiegeln Sie das Loadtester-Image in ein privates ECR-Repository

> **Note**
>
> Wenn Sie angeben`LoadTesterImageUri`, verwaltet der Parameter Auto-update Container Image (`UseStableTagging`) das Loadtester-Image nicht mehr. Um Sicherheitspatches zu erhalten, müssen Sie die aktualisierten Image-Tags erneut spiegeln.

Gehen Sie wie folgt vor, um das Loadtester-Image in ein privates ECR-Repository zu spiegeln. `<version>`Ersetzen Sie in den folgenden Befehlen durch das Image-Tag (z. B.`v4.0.0`), `<region>` durch Ihre Bereitstellungsregion (z. B.`us-gov-west-1`), `<profile>` durch Ihren AWS-Befehlszeilenschnittstellen-Profilnamen (AWS CLI) und `<account-id>` durch Ihre 12-stellige AWS-Konto-ID.

1. Authentifizieren Sie sich bei der öffentlichen ECR-Registrierung von einem Computer mit Internetzugang aus.

```
aws ecr-public get-login-password --region us-east-1 | docker login --username AWS --
password-stdin public.ecr.aws
```

2. Rufen Sie das öffentliche Loadtester-Image ab.

```
docker pull public.ecr.aws/aws-solutions/distributed-load-testing-on-aws-load-
tester:<version>
```

3. Erstellen Sie ein privates ECR-Repository in Ihrem Konto (falls noch keines vorhanden ist).

```
aws ecr create-repository --repository-name dlt-load-tester --region <region> --
profile <profile> 2>/dev/null || true
```

4. Authentifizieren Sie sich bei Ihrer privaten ECR-Registrierung.

```
aws ecr get-login-password --region <region> --profile <profile> | docker login --
username AWS --password-stdin <account-id>.dkr.ecr.<region>.amazonaws.com
```

5. Taggen Sie das Bild und übertragen Sie es in Ihr privates Repository.

```
docker tag public.ecr.aws/aws-solutions/distributed-load-testing-on-aws-load-
tester:<version> \
```

```
  <account-id>.dkr.ecr.<region>.amazonaws.com/dlt-load-tester:<version>
docker push <account-id>.dkr.ecr.<region>.amazonaws.com/dlt-load-tester:<version>
```

6. Geben Sie den privaten Image-URI im Parameter Load Tester Image URI ein, wenn Sie den Stack starten.

```
<account-id>.dkr.ecr.<region>.amazonaws.com/dlt-load-tester:<version>
```

AWS-Bereitstellungsbeispiele GovCloud (USA)

Die folgenden Beispiele zeigen die Parameter, die für jede Stack-Variante in AWS GovCloud (USA) erforderlich sind.

Für den Headless-Stack `LoadTesterImageUri` ist nur Folgendes erforderlich:

```
aws cloudformation create-stack \
  --stack-name dlt-govcloud \
  --template-url https://solutions-reference-us-gov.s3.us-gov-west-1.amazonaws.com/
distributed-load-testing-on-aws/latest/distributed-load-testing-on-aws-
headless.template \
  --parameters \
    ParameterKey=AdminName,ParameterValue=admin \
    ParameterKey=AdminEmail,ParameterValue=admin@example.com \
    ParameterKey=LoadTesterImageUri,ParameterValue=<account-id>.dkr.ecr.us-gov-
west-1.amazonaws.com/dlt-load-tester:<version> \
  --capabilities CAPABILITY_IAM \
  --region us-gov-west-1
```

Geben Sie für den Application Load Balancer (ALB) + ECS auf dem AWS Fargate-Stack sowohl als auch an. `LoadTesterImageUri` `WebConsoleImageUri` ECS-Aufgaben in AWS GovCloud (USA) können nicht auf die öffentliche ECR-Registrierung (`public.ecr.aws`) zugreifen, daher müssen Sie beide Container-Images spiegeln.

```
aws cloudformation create-stack \
  --stack-name dlt-govcloud \
  --template-url https://solutions-reference-us-gov.s3.us-gov-west-1.amazonaws.com/
distributed-load-testing-on-aws/latest/distributed-load-testing-on-aws-alb-ecs.template
 \
  --parameters \
    ParameterKey=AdminName,ParameterValue=admin \
```

```
    ParameterKey=AdminEmail,ParameterValue=admin@example.com \
    ParameterKey=ConsoleDomainName,ParameterValue=dlt.example.gov \
    ParameterKey=ACMCertificateArn,ParameterValue=arn:aws-us-gov:acm:us-gov-
west-1:123456789012:certificate/abcd1234-ef56-gh78-ij90-klmnopqrstuv \
    ParameterKey=LoadTesterImageUri,ParameterValue=<account-id>.dkr.ecr.us-gov-
west-1.amazonaws.com/dlt-load-tester:<version> \
    ParameterKey=WebConsoleImageUri,ParameterValue=<account-id>.dkr.ecr.us-gov-
west-1.amazonaws.com/dlt-web-console:<version> \
  --capabilities CAPABILITY_IAM \
  --region us-gov-west-1
```

Anweisungen zum Spiegeln des Web-Konsolen-Images finden Sie unter Web-Konsolen-Image.

API für verteilte Lasttests

Diese Lasttestlösung hilft Ihnen dabei, Testergebnisdaten auf sichere Weise verfügbar zu machen. Die API fungiert als „Eingangstür“ für den Zugriff auf Testdaten, die in Amazon DynamoDB gespeichert sind. Sie können die APIs auch verwenden, um auf alle erweiterten Funktionen zuzugreifen, die Sie in die Lösung integriert haben.

Diese Lösung verwendet einen Amazon Cognito Cognito-Benutzerpool, der in Amazon API Gateway zur Identifizierung und Autorisierung integriert ist. Wenn ein Benutzerpool mit der API verwendet wird, dürfen Clients nur Methoden aufrufen, die vom Benutzerpool aktiviert wurden, nachdem sie ein gültiges Identitätstoken bereitgestellt haben.

Weitere Informationen zur Ausführung von Tests direkt über die API finden Sie unter Signing Requests in der Amazon API Gateway REST API-Referenzdokumentation.

Die folgenden Operationen sind in der API der Lösung verfügbar.

> **Note**
>
> Weitere Informationen zu `testScenario` und anderen Parametern finden Sie in den Szenarien und Payload-Beispielen im GitHub Repository.

Informationen zum Stapel

- ERHALTE /stack-info

Szenarien

- GET /scenarios
- POST /szenarien
- OPTIONEN /Szenarien
- GET /scenarios/ {testId}
- POSTE /scenarios/ {testId}
- LÖSCHEN SIE /scenarios/ {testId}
- OPTIONEN /scenarios/ {testID}

Testläufe

- GET /scenarios/ {testId} /testruns
- GET /scenarios/ {testID} /testruns/ {test} RunId
- LÖSCHE /scenarios/ {testID} /testruns/ {test} RunId

Ausgangswert

- GET /scenarios/ {testId} /baseline
- GIB /scenarios/ {testId} /baseline ein
- LÖSCHEN SIE /scenarios/ {testId} /baseline

Aufgaben

- HOLEN SIE SICH /tasks
- OPTIONEN /Aufgaben

Regionen

- HOLEN SIE SICH /regions
- OPTIONEN /regions

ERHALTE /stack-info

Description

Der `GET /stack-info` Vorgang ruft Informationen über den bereitgestellten Stack ab, einschließlich Erstellungszeit, Region und Version. Dieser Endpunkt wird vom Frontend verwendet.

Antwort

200 — Erfolg

Name	Description
`created_time`	ISO 8601-Zeitstempel, als der Stack erstellt wurde (zum Beispiel) `2025-09-09T19:40:22Z`
`region`	AWS-Region, in der der Stack bereitgestellt wird (z. B.`us-east-1`)
`version`	Version der bereitgestellten Lösung (z. B.`v4.0.0`)

Antworten auf Fehler

- `403`- Verboten: Unzureichende Berechtigungen für den Zugriff auf Stack-Informationen
- `404`- Nicht gefunden: Stack-Informationen nicht verfügbar
- `500`- Interner Serverfehler

GET /scenarios

Description

Die `GET /scenarios` Operation ermöglicht es Ihnen, eine Liste von Testszenarien abzurufen.

Antwort

Name	Description
`data`	Eine Liste von Szenarien, einschließlich der ID, des Namens, der Beschreibung, des Status, der Laufzeit, der Tags, der Gesamtzahl der Testläufe und der letzten Ausführung für jeden Test

POST /szenarien

Description

Die `POST /scenarios` Operation ermöglicht es Ihnen, ein Testszenario zu erstellen oder zu planen.

Anforderungstext

Name	Description
`testName`	Der Name des Tests
`testDescription`	Die Beschreibung des Tests
`testTaskConfigs`	Ein Objekt, das `concurrency` (die Anzahl der parallel Läufe), `taskCount` (die Anzahl der Aufgaben, die zur Ausführung eines Tests erforderlich sind) und `region` für das Szenario angibt
`testScenario`	Die Testspezifikation, einschließlich Parallelität, Testzeit, Host und Methode für den Test
`nativeRunMode`	Ein Objekt, das den Traffic Shape-Mod us auswählt. Lassen Sie es weg, um den Standardmodus zu verwenden. Fügen Sie es mit einer `maxTestDurationSeconds`

Name	Description
	Sicherheitsdauer von bis zu 24 Stunden hinzu, um den systemeigenen Modus auszuwählen und das Laden vom hochgeladenen Skript steuern zu lassen. Nur skriptbasierte Tests unterstützen den systemeigenen Modus. Weitere Informationen finden Sie unter Traffic Shape Modes.
`testType`	Der Testtyp (zum Beispiel`simple`,`jmeter`)
`fileType`	Der Upload-Dateityp (z. B.`none`,`script`,`zip`)
`tags`	Eine Reihe von Zeichenketten zur Kategorisierung von Tests. Optionales Feld mit einer maximalen Länge von 5 (z. B.`["blue", "3.0", "critical"]`)
`scheduleDate`	Das Datum, an dem ein Test ausgeführt werden soll. Wird nur angegeben, wenn ein Test geplant wird (z. B.`2021-02-28`)
`scheduleTime`	Die Zeit, um einen Test durchzuführen. Wird nur angegeben, wenn ein Test geplant wird `21:07` (z. B.
`scheduleStep`	Der Schritt im Planungsprozess. Wird nur bereitgestellt, wenn ein wiederkehrender Test geplant wird. (Zu den verfügbaren Schritten gehören `create` und`start`)
`cronvalue`	Der Cron-Wert für die Anpassung der wiederkehrenden Terminplanung. Falls verwendet, lassen Sie ScheduleDate und ScheduleTime weg.
`cronExpiryDate`	Erforderliches Datum, damit der Cron abläuft und nicht unbegrenzt läuft.

Name	Description
`recurrence`	Die Wiederholung eines geplanten Tests. Wird nur bereitgestellt, wenn ein wiederkeh render Test geplant wird (z. B.`daily`, `weeklybiweekly`, oder`monthly`)

Antwort

Name	Description
`testId`	Die eindeutige ID des Tests
`testName`	Der Name des Tests
`status`	Der Status des Tests

OPTIONEN/Szenarien

Description

Die `OPTIONS /scenarios` Operation liefert eine Antwort auf die Anfrage mit den richtigen CORS-Antwortheadern.

Antwort

Name	Description
`testId`	Die eindeutige ID des Tests
`testName`	Der Name des Tests
`status`	Der Status des Tests

GET /scenarios/ {testId}

Description

Die `GET /scenarios/{testId}` Operation ermöglicht es Ihnen, die Details eines bestimmten Testszenarios abzurufen.

Anforderungsparameter

`testId`

- Die eindeutige ID des Tests

  Typ: Zeichenfolge

  Erforderlich: Ja

`latest`

- Abfrageparameter, um nur den letzten Testlauf zurückzugeben. Die Standardeinstellung ist `true`

  Typ: Boolesch

  Erforderlich: Nein

`history`

- Abfrageparameter, um den Testlaufverlauf in die Antwort aufzunehmen. Der Standardwert ist `true`. Auf einstellen, `false` um den Verlauf auszuschließen

  Typ: Boolesch

  Erforderlich: Nein

Antwort

Name	Description
`testId`	Die eindeutige ID des Tests
`testName`	Der Name des Tests

Name	Description
`testDescription`	Die Beschreibung des Tests
`testType`	Die Art des Tests, der ausgeführt wird (z. B.`simple`,`jmeter`)
`fileType`	Der Typ der Datei, die hochgeladen wird (z. B.`none`,`script`,`zip`)
`tags`	Eine Reihe von Zeichenketten zur Kategorisierung von Tests
`status`	Der Status des Tests
`startTime`	Die Uhrzeit und das Datum, an dem der letzte Test gestartet wurde
`endTime`	Uhrzeit und Datum, an dem der letzte Test beendet wurde
`testScenario`	Die Testspezifikation, einschließlich Parallelität, Testzeit, Host und Methode für den Test
`taskCount`	Die Anzahl der Aufgaben, die zur Ausführung des Tests erforderlich sind
`taskIds`	Eine Liste von Aufgaben-IDs für die Ausführung von Tests
`results`	Die endgültigen Ergebnisse des Tests
`history`	Eine Liste der Endergebnisse früherer Tests (ausgenommen wann`history=false`)
`totalRuns`	Die Gesamtzahl der Testläufe für dieses Szenario
`lastRun`	Der Zeitstempel des letzten Testlaufs

Name	Description
`errorReason`	Eine Fehlermeldung, die generiert wird, wenn ein Fehler auftritt
`nextRun`	Der nächste geplante Lauf (zum Beispiel`2017-04-22 17:18:00`)
`scheduleRecurrence`	Die Wiederholung des Tests (z. B.,`daily`, `weeklybiweekly`,`monthly`)

POST /scenarios/ {TestID}

Description

Der `POST /scenarios/{testId}` Vorgang ermöglicht es Ihnen, ein bestimmtes Testszenario abzubrechen.

Parameter anfordern

`testId`

- Die eindeutige ID des Tests

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`status`	Der Status des Tests

LÖSCHEN SIE /scenarios/ {testId}

Description

Die `DELETE /scenarios/{testId}` Operation ermöglicht es Ihnen, alle Daten zu löschen, die sich auf ein bestimmtes Testszenario beziehen.

Parameter anfordern

`testId`

- Die eindeutige ID des Tests

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`status`	Der Status des Tests

OPTIONEN /scenarios/ {testId}

Description

Die `OPTIONS /scenarios/{testId}` Operation liefert eine Antwort auf die Anfrage mit den richtigen CORS-Antwortheadern.

Antwort

Name	Description
`testId`	Die eindeutige ID des Tests
`testName`	Der Name des Tests
`testDescription`	Die Beschreibung des Tests

Name	Description
`testType`	Die Art des Tests, der ausgeführt wird (z. B.`simple`,`jmeter`)
`fileType`	Der Typ der Datei, die hochgeladen wird (z. B.`none`,`script`,`zip`)
`status`	Der Status des Tests
`startTime`	Die Uhrzeit und das Datum, an dem der letzte Test gestartet wurde
`endTime`	Uhrzeit und Datum, an dem der letzte Test beendet wurde
`testScenario`	Die Testspezifikation, einschließlich Parallelität, Testzeit, Host und Methode für den Test
`taskCount`	Die Anzahl der Aufgaben, die zur Ausführung des Tests erforderlich sind
`taskIds`	Eine Liste von Aufgaben-IDs für die Ausführung von Tests
`results`	Die endgültigen Ergebnisse des Tests
`history`	Eine Liste der Endergebnisse vergangener Tests
`errorReason`	Eine Fehlermeldung, die generiert wird, wenn ein Fehler auftritt

GET /scenarios/ {testId} /testruns

Description

Der `GET /scenarios/{testId}/testruns` Vorgang ruft Testlauf-IDs für ein bestimmtes Testszenario ab, optional gefiltert nach Zeitbereich. Wenn`latest=true`, gibt nur den letzten Testlauf zurück.

Anforderungsparameter

`testId`

- Die ID des Testszenarios

  Typ: Zeichenfolge

  Erforderlich: Ja

`latest`

- Gibt nur die letzte Testlauf-ID zurück

  Typ: Boolescher Wert

  Standard: `false`

  Erforderlich: Nein

`start_timestamp`

- ISO 8601-Zeitstempel zum Filtern von Testläufen (einschließlich). Beispiel: `2024-01-01T00:00:00Z`

  Typ: Zeichenfolge (Datums-/Uhrzeitformat)

  Erforderlich: Nein

`end_timestamp`

- ISO 8601-Zeitstempel zum Filtern von Testläufen bis (einschließlich). Beispiel: `2024-12-31T23:59:59Z`

  Typ: Zeichenfolge (Datums-/Uhrzeitformat)

  Erforderlich: Nein

`limit`

- Maximale Anzahl zurückzugebender Testläufe (wird ignoriert, wenn) `latest=true`

  Typ: Integer (mindestens: 1, maximal: 100)

  Standard: `20`

  Erforderlich: Nein

`next_token`

- Paginierungstoken aus der vorherigen Antwort, um die nächste Seite zu erhalten

  Typ: Zeichenfolge

  Erforderlich: Nein

Antwort

200 — Erfolg

Name	Description
`testRuns`	Array von Testlaufobjekten, die jeweils `testRunId` (Zeichenfolge) und `startTime` (ISO 8601 Datum/Uhrzeit) enthalten
`pagination`	Objekt, das `limit` (Ganzzahl) und `next_token` (Zeichenfolge oder Null) enthält. Das Token ist Null, wenn keine weiteren Ergebnisse vorliegen

Antworten auf Fehler

- `400`- Ungültiges Zeitstempelformat oder ungültige Parameter
- `404`- Das Testszenario wurde nicht gefunden
- `500`- Interner Serverfehler

Beispielverwendung

- Nur letzter Testlauf: `GET /scenarios/test123/testruns?latest=true`
- Spätester innerhalb des Zeitbereichs: `GET /scenarios/test123/testruns?latest=true&start_timestamp=2024-01-01T00:00:00Z`
- Anfrage zur nächsten Seite: `GET /scenarios/test123/testruns?limit=20&next_token=eyJ0ZXN0SWQiOiJzZVFVeTEyTEtMIiwic3RhcnRUaW1lIjoiMjAyNC0wMS`

GET /scenarios/ {testId} /testruns/ {test} RunId

Description

Der `GET /scenarios/{testId}/testruns/{testRunId}` Vorgang ruft vollständige Ergebnisse und Metriken für einen bestimmten Testlauf ab. Lassen Sie optional Verlaufsergebnisse aus, um schneller reagieren `history=false` zu können.

Anforderungsparameter

`testId`

- Die ID des Testszenarios

  Typ: Zeichenfolge

  Erforderlich: Ja

`testRunId`

- Die spezifische Testlauf-ID

  Typ: Zeichenfolge

  Erforderlich: Ja

`history`

- Schließt das Verlaufs-Array als Antwort ein. Stellen Sie auf ein`false`, um den Verlauf für eine schnellere Antwort auszulassen

  Typ: Boolescher Wert

  Standard: `true`

Erforderlich: Nein

Antwort

200 — Erfolgreich

Name	Description
`testId`	Die eindeutige ID des Tests (zum Beispiel`seQUy12LKL`)
`testRunId`	Die spezifische Testlauf-ID (z. B.`2DEwHItEn e`)
`testDescription`	Beschreibung des Belastungstests
`testType`	Die Art des Tests (z. B.`simple`,`jmeter`)
`status`	Der Status des Testlaufs: `completerunning`,`failed`, oder `cancelled`
`startTime`	Die Uhrzeit und das Datum, an dem der Test gestartet wurde (z. B.`2025-09-09 21:01:00`)
`endTime`	Uhrzeit und Datum, an dem der Test beendet wurde (z. B.`2025-09-09 21:18:29`)
`succPercent`	Erfolgsquote (zum Beispiel`100.00`)
`testTaskConfigs`	Array von Aufgabenkonfigurationsobjek tenregion, die`taskCount` , und enthalten `concurrency`
`completeTasks`	Objektzuweisung von Regionen zur Anzahl abgeschlossener Aufgaben

Name	Description
`results`	Objekt, das detaillierte Metriken wie `avg_lt` (durchschnittliche Latenz), Perzentile (`p0_0p50_0,p90_0,,p95_0,p99_0,p99_9,p100_0`), `avg_rt` (durchschnittliche Reaktionszeit), `avg_ct` (durchschnittliche Verbindungszeit), `stdev_rt` (Reaktionszeit mit Standarda bweichung),, `concurrency throughput` , `succ` (Anzahl der Erfolge), `fail` (Anzahl der Fehler),, `bytes testDuration metricS3L ocation` , `rc` (Antwortcode-Array) und Array enthält `labels`
`testScenario`	Objekt, das eine Testkonfiguration mit Eigenschaften`execution` , und `reporting` enthält `scenarios`
`history`	Array mit historischen Testergebnissen (ausgenommen wann`history=false`)

Antworten auf Fehler

- `400`- Ungültige TestID oder Test RunId
- `404`- Testlauf wurde nicht gefunden
- `500`- Interner Serverfehler

LÖSCHEN Sie /scenarios/ {testId} /testruns/ {test} RunId

Description

Der `DELETE /scenarios/{testId}/testruns/{testRunId}` Vorgang löscht alle Daten und Artefakte, die sich auf einen bestimmten Testlauf beziehen. Die Testlaufdaten werden aus DynamoDB entfernt, während die tatsächlichen Testdaten in S3 unverändert bleiben.

Anforderungsparameter

`testId`

- Die ID des Testszenarios

  Typ: Zeichenfolge

  Erforderlich: Ja

`testRunId`

- Die spezifische Testlauf-ID, die gelöscht werden soll

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

204 — Erfolgreich

Testlauf wurde erfolgreich gelöscht (es wurde kein Inhalt zurückgegeben)

Antworten auf Fehler

- `400`- Ungültige TestID oder Test RunId
- `403`- Verboten: Unzureichende Berechtigungen zum Löschen des Testlaufs
- `404`- Testlauf nicht gefunden
- `409`- Konflikt: Der Testlauf läuft gerade und kann nicht gelöscht werden
- `500`- Interner Serverfehler

GET /scenarios/ {testId} /baseline

Description

Der `GET /scenarios/{testId}/baseline` Vorgang ruft das angegebene Baseline-Testergebnis für ein Szenario ab. Gibt je nach `data` Parameter entweder die ID des Ausgangstestlaufs oder die vollständigen Baseline-Ergebnisse zurück.

Anforderungsparameter

`testId`

- Die ID des Testszenarios

  Typ: Zeichenfolge

  Erforderlich: Ja

`data`

- Gibt die vollständigen Basisdaten des Testlaufs zurück`true`, andernfalls nur den Test RunId

  Typ: Boolescher Wert

  Standard: `false`

  Erforderlich: Nein

Antwort

200 — Erfolgreich

Wann `data=false` (Standard):

Name	Description
`testId`	Die ID des Testszenarios (zum Beispiel`eQUy12LKL`)
`baselineTestRunId`	Die ID des Ausgangstestlaufs (z. B.`2DEwHItEne`)

Wann`data=true`:

Name	Description
`testId`	Die ID des Testszenarios (zum Beispiel`eQUy12LKL`)

Name	Description
`baselineTestRunId`	Die ID des Ausgangstestlaufs (z. B.`2DEwHItEne`)
`baselineData`	Vollständiges Objekt mit den Ergebnissen des Testlaufs (gleiche Struktur wie`GET /scenarios/{testId}/testruns/{testRunId}`)

Antworten auf Fehler

- `400`- Ungültiger TestID-Parameter
- `404`- Das Testszenario wurde nicht gefunden oder es wurde kein Basiswert festgelegt
- `500`- Interner Serverfehler

PUT /scenarios/ {TestId} /baseline

Description

Der `PUT /scenarios/{testId}/baseline` Vorgang legt einen bestimmten Testlauf als Grundlage für den Leistungsvergleich fest. Pro Szenario kann nur eine Basislinie festgelegt werden.

Anforderungsparameter

`testId`

- Die ID des Testszenarios

  Typ: Zeichenfolge

  Erforderlich: Ja

Anforderungstext

Name	Description
`testRunId`	Die Testlauf-ID, die als Ausgangswert festgeleg t werden soll (z. B.`2DEwHItEne`)

Antwort

200 — Erfolgreich

Name	Description
`message`	Bestätigungsnachricht (zum Beispiel`Baseline set successfully`)
`testId`	Die ID des Testszenarios (zum Beispiel`seQUy12LKL`)
`baselineTestRunId`	Die eingestellte Baseline-Testlauf-ID (z. B.`2DEwHItEne`)

Antworten auf Fehler

- `400`- Ungültige TestID oder Test RunId
- `404`- Testszenario oder Testlauf nicht gefunden
- `409`- Konflikt: Der Testlauf kann nicht als Ausgangswert festgelegt werden (z. B. fehlgeschlagener Test)
- `500`- Interner Serverfehler

LÖSCHEN Sie /scenarios/ {testId} /baseline

Description

Die `DELETE /scenarios/{testId}/baseline` Operation löscht den Basiswert für ein Szenario, indem er auf eine leere Zeichenfolge gesetzt wird.

Anforderungsparameter

`testId`

- Die ID des Testszenarios

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

204 — Erfolg

Baseline wurde erfolgreich gelöscht (es wurde kein Inhalt zurückgegeben)

Antworten auf Fehler

- `400`- Ungültige TestID
- `500`- Interner Serverfehler

GET /tasks

Description

Mit `GET /tasks` diesem Vorgang können Sie eine Liste der laufenden Amazon Elastic Container Service (Amazon ECS) -Aufgaben abrufen.

Antwort

Name	Description
`tasks`	Eine Liste von Aufgaben-IDs für die Ausführung von Tests

OPTIONEN /Aufgaben

Description

Der Vorgang `OPTIONS /tasks` tasks liefert eine Antwort auf die Anfrage mit den richtigen CORS-Antwortheadern.

Antwort

Name	Description
`taskIds`	Eine Liste von Aufgaben-IDs für die Ausführung von Tests

GET /regions

Description

Mit diesem `GET /regions` Vorgang können Sie die regionalen Ressourceninformationen abrufen, die für die Durchführung eines Tests in dieser Region erforderlich sind.

Antwort

Name	Description
`testId`	Die Regions-ID
`ecsCloudWatchLogGroup`	Der Name der CloudWatch Amazon-Protokollgruppe für die AWS Fargate-Aufgaben in der Region
`region`	Die Region, in der die Ressourcen in der Tabelle existieren
`subnetA`	Die ID eines der Subnetze in der Region
`subnetB`	Die ID eines der Subnetze in der Region

Name	Description
`taskCluster`	Der Name des AWS Fargate-Clusters in der Region
`taskDefinition`	Der ARN der Aufgabendefinition in der Region
`taskImage`	Der Name des Task-Images in der Region
`taskSecurityGroup`	Die ID der Sicherheitsgruppe in der Region

OPTIONEN /Regionen

Description

Der `OPTIONS /regions` Vorgang liefert eine Antwort auf die Anfrage mit den richtigen CORS-Antwortheadern.

Antwort

Name	Description
`testId`	Die Regions-ID
`ecsCloudWatchLogGroup`	Der Name der CloudWatch Amazon-Protokollgruppe für die AWS Fargate-Aufgaben in der Region
`region`	Die Region, in der die Ressourcen in der Tabelle existieren
`subnetA`	Die ID eines der Subnetze in der Region
`subnetB`	Die ID eines der Subnetze in der Region
`taskCluster`	Der Name des AWS Fargate-Clusters in der Region
`taskDefinition`	Der ARN der Aufgabendefinition in der Region

Name	Description
`taskImage`	Der Name des Task-Images in der Region
`taskSecurityGroup`	Die ID der Sicherheitsgruppe in der Region

Erhöhen Sie die Container-Ressourcen

Um die Anzahl der gleichzeitigen virtuellen Benutzer (Parallelität) zu erhöhen, die Ihre Auslastungstests simulieren können, müssen Sie die CPU- und Speicherressourcen erhöhen, die jeder Amazon ECS-Aufgabe zugewiesen sind. Dazu müssen Sie eine neue Version der Aufgabendefinition mit höheren Ressourcenlimits erstellen und anschließend die DynamoDB-Konfiguration der Lösung aktualisieren, um die neue Aufgabendefinition für future Testläufe zu verwenden.

Erstellen Sie eine neue Version der Aufgabendefinition

Gehen Sie wie folgt vor, um eine neue Aufgabendefinition mit erhöhten CPU- und Speicherressourcen zu erstellen:

1. Melden Sie sich bei der Amazon Elastic Container Service-Konsole an.
2. Wählen Sie im linken Navigationsmenü Aufgabendefinitionen aus.
3. Aktivieren Sie das Kontrollkästchen neben der Aufgabendefinition, die dieser Lösung entspricht. Zum Beispiel `[replaceable]` <stackName>`- EcsTaskDefinition -*<system-generated-random-Hash>*.
4. Wählen Sie Create new revision (Neue Revision erstellen) aus.
5. Führen Sie auf der Seite Neue Revision erstellen die folgenden Aktionen aus:
   a. Ändern Sie unter Task-Größe die Werte Task-Speicher und Task-CPU auf die gewünschten Werte. Höhere Werte ermöglichen mehr gleichzeitige virtuelle Benutzer pro Aufgabe.
   b. Überprüfen Sie unter Containerdefinitionen die Hard/Soft Speicherlimits. Wenn dieses Limit unter dem von Ihnen gewünschten Speicherplatz liegt, wählen Sie den Container aus.
   c. Gehen Sie im Dialogfeld „Container bearbeiten“ zu Speicherlimits und aktualisieren Sie das feste Limit, sodass es mit Ihrer Aufgabenspeicherzuweisung übereinstimmt oder darunter liegt.
   d. Wählen Sie Aktualisieren aus.
6. Wählen Sie auf der Seite Neue Revision erstellen die Option Erstellen aus.

7. Nachdem die Aufgabendefinition erfolgreich erstellt wurde, notieren Sie den vollständigen ARN der Aufgabendefinition einschließlich der Versionsnummer. Zum Beispiel: `[replaceable]` <stackName>`- EcsTaskDefinition -*<system-generated-random-Hash>*: [austauschbar]`<system-generated-versionNumber>`.

Aktualisieren Sie die DynamoDB-Tabelle

Nachdem Sie die neue Version der Aufgabendefinition erstellt haben, müssen Sie die DynamoDB-Tabelle der Lösung aktualisieren, sodass future Testläufe die neue Aufgabendefinition verwenden. Wiederholen Sie diese Schritte für jede AWS-Region, in der Sie die aktualisierte Aufgabendefinition verwenden möchten:

1. Navigieren Sie zur DynamoDB-Konsole.
2. Wählen Sie im linken Navigationsbereich unter Tabellen die Option Elemente durchsuchen aus.
3. Wählen Sie die `scenarios-table` DynamoDB-Tabelle aus, die dieser Lösung zugeordnet ist. Zum Beispiel `[replaceable]` <stackName>`- -DLTTestRunnerStorageDLTScenariosTable. *<system-generated-random-Hash>*
4. Wählen Sie das Element aus, das der Region entspricht, in der Sie die neue Version der Aufgabendefinition erstellt haben. Zum Beispiel `region-[replaceable]` <region-name>`.
5. Suchen Sie im Element-Editor das TaskDefinition-Attribut und aktualisieren Sie seinen Wert mit dem vollständigen Task-Definition-ARN, den Sie im vorherigen Abschnitt aufgezeichnet haben (einschließlich der Versionsnummer).
6. Wählen Sie Änderungen speichern aus.

> **Note**
>
> Die aktualisierte Aufgabendefinition wird nur für neue Testläufe verwendet. Für alle derzeit laufenden oder geplanten Tests wird weiterhin die vorherige Aufgabendefinition verwendet.

Spezifikation der MCP-Tools

Die Distributed Load Testing-Lösung bietet eine Reihe von MCP-Tools, mit denen KI-Agenten mit Testszenarien und Ergebnissen interagieren können. Diese Tools bieten umfassende, abstrakte Funktionen, die auf die Art und Weise abgestimmt sind, wie KI-Agenten Informationen verarbeiten,

sodass sie sich auf Analysen und Erkenntnisse konzentrieren können, anstatt sich auf detaillierte API-Verträge zu konzentrieren.

Der MCP-Server unterstützt zwei Zugriffsmodi, die durch den `MCPServerAccessMode` CloudFormation AWS-Parameter gesteuert werden:

- ReadOnly(Standard) — Nur Lesetools sind registriert. Agenten sehen 7 Tools über`tools/list`. Es sind keine Mutationsoperationen verfügbar.
- ReadWrite— Sowohl Lese- als auch Schreibwerkzeuge sind registriert. Agenten sehen alle Tools (Lese- und Schreibzugriff) über `tools/list` und können Tests erstellen, Läufe auslösen, Zeitpläne verwalten und Skripte hochladen.

Der Zugriffsmodus wird bei der Bereitstellung festgelegt. Um den Zugriffsmodus nach der ersten Bereitstellung zu ändern, führen Sie ein CloudFormation Stack-Update mit dem neuen `MCPServerAccessMode` Parameterwert durch. Die Änderung wird wirksam, wenn das Stack-Update abgeschlossen ist — es sind keine weiteren manuellen Schritte erforderlich.

Im ReadOnly Modus sind Schreibwerkzeuge überhaupt nicht registriert — Agenten sehen sie nie`tools/list`. Die AWS Identity and Access Management (IAM) -Richtlinie für die AWS-Lambda-Funktion des MCP-Servers ist entsprechend festgelegt. ReadOnly erlaubt nur GET-Anfragen an die API. ReadWrite erlaubt GET, POST, PUT und DELETE.

Tools lesen

list_scenarios

Description

Das `list_scenarios` Tool ruft eine Liste aller verfügbaren Testszenarien mit grundlegenden Metadaten ab.

Endpoint

`GET /scenarios`

Parameters

Keine

Antwort

Name	Description
`testId`	Eindeutiger Bezeichner für das Testszenario
`testName`	Name des Testszenarios
`status`	Aktueller Status des Testszenarios
`startTime`	Wann der Test erstellt oder zuletzt ausgeführt wurde
`testDescription`	Beschreibung des Testszenarios

get_scenario_details

Description

Das `get_scenario_details` Tool ruft die Testkonfiguration und den letzten Testlauf für ein einzelnes Testszenario ab.

Die Antwort gibt den Traffic-Shape-Modus des Szenarios an. Ein `nativeRunMode` Objekt weist auf den einheitlichen Modus hin, und sein Fehlen weist auf den Standardmodus hin. In einem nativen Szenario geben die `holdFor` Felder `concurrency` `rampUp`, und nicht die Last wieder, die durch den Lauf generiert wurde. Die Last stammt stattdessen aus dem Skript. Weitere Informationen finden Sie unter Verkehrsshape-Modi.

Endpoint

`GET /scenarios/<test_id>?history=false&results=false`

Anforderungsparameter

`test_id`

- Die eindeutige Kennung für das Testszenario

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`testTaskConfigs`	Aufgabenkonfiguration für jede Region
`testScenario`	Testdefinition und Parameter
`status`	Aktueller Teststatus
`startTime`	Zeitstempel für den Teststart
`endTime`	Endzeitstempel des Tests (falls abgeschlossen)

list_test_runs

Description

Das `list_test_runs` Tool ruft eine Liste von Testläufen für ein bestimmtes Testszenario ab, sortiert vom neuesten zum ältesten. Gibt maximal 30 Ergebnisse zurück. Es `start_timestamp` kann nur eines von `limit` oder angegeben werden, nicht beide.

Endpoint

`GET /scenarios/<testid>/testruns/?limit=<limit>`

oder

`GET /scenarios/<testid>/testruns/?start_timestamp=<start_timestamp>`

Anforderungsparameter

`test_id`

- Die eindeutige Kennung für das Testszenario

  Typ: Zeichenfolge

  Erforderlich: Ja

`limit`

- Maximale Anzahl zurückzugebender Testläufe. Kann nicht mit `start_timestamp` verwendet werden.

Typ: Ganzzahl

Standard: 20

Maximum: 30

Erforderlich: Nein

`start_timestamp`

- Gibt alle Testläufe zurück, die bis zu diesem Zeitstempel zurückreichen. Kann nicht mit `limit` verwendet werden.

  Typ: Zeichenfolge (z. B. Datums- und Uhrzeitformat nach ISO 8601)
  `2024-01-15T14:30:00.000Z`

  Erforderlich: Nein

Antwort

Name	Description
`testRuns`	Reihe von Testlaufzusammenfassungen mit Leistungskennzahlen und Perzentilen für jeden Lauf

get_test_run

Description

Das `get_test_run` Tool ruft detaillierte Ergebnisse für einen einzelnen Testlauf mit regionalen und Endpunktaufschlüsselungen ab.

Endpoint

`GET /scenarios/<testid>/testruns/<testrunid>`

Anforderungsparameter

`test_id`

- Die eindeutige Kennung für das Testszenario

  Typ: Zeichenfolge

  Erforderlich: Ja

`test_run_id`

- Die eindeutige Kennung für den spezifischen Testlauf

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`results`	Vollständige Testlaufdaten, einschließlich Aufschlüsselung der regionalen Ergebnisse, endpunktspezifischer Metriken, Leistungsperzentile (p50, p90, p95, p99), Erfolgs- und Fehlschlagzahlen, Reaktionszeiten und Latenz sowie der für den Testlauf verwendeten Testkonfiguration

get_latest_test_run

Description

Das `get_latest_test_run` Tool ruft den letzten Testlauf für ein bestimmtes Testszenario ab.

Endpoint

```
GET /scenarios/<testid>/testruns/?limit=1
```

> **Note**
>
> Die Ergebnisse werden anhand eines Global Secondary Index (GSI) nach Zeit sortiert, sodass der letzte Testlauf zurückgegeben wird.

Anforderungsparameter

`test_id`

- Die eindeutige Kennung für das Testszenario

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`results`	Aktuelle Testlaufdaten mit demselben Format wie `get_test_run`

get_baseline_test_run

Description

Das `get_baseline_test_run` Tool ruft den Basistestlauf für ein bestimmtes Testszenario ab. Die Baseline wird zu Leistungsvergleichszwecken verwendet.

Endpoint

`GET /scenarios/<test_id>/baseline`

Anforderungsparameter

`test_id`

- Die eindeutige Kennung für das Testszenario

  Typ: Zeichenfolge

Erforderlich: Ja

Antwort

Name	Description
`baselineData`	Basisdaten des Testlaufs zu Vergleichszwecken, einschließlich aller Metriken und Konfigurationen aus dem angegebenen Baselinelauf

get_test_run_artifacts

Description

Das `get_test_run_artifacts` Tool ruft Amazon S3 S3-Bucket-Informationen für den Zugriff auf Testartefakte wie Protokolle, Fehlerdateien und Ergebnisse ab.

Endpoint

```
GET /scenarios/<testid>/testruns/<testrunid>
```

Anforderungsparameter

`test_id`

- Die eindeutige Kennung für das Testszenario

  Typ: Zeichenfolge

  Erforderlich: Ja

`test_run_id`

- Die eindeutige Kennung für den spezifischen Testlauf

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`bucketName`	Name des S3-Buckets, in dem Artefakte gespeichert werden
`testRunPath`	Pfadpräfix für den aktuellen Artefaktspeicher (Version 4.0+)
`testScenarioPath`	Pfadpräfix für älteren Artefaktspeicher (vor Version 4.0)

Tools schreiben

Schreibwerkzeuge sind nur verfügbar, wenn auf eingestellt `MCPServerAccessMode` ist`ReadWrite`. Sie ermöglichen es Agenten, Testszenarien zu erstellen, zu ändern und auszuführen.

create_test

Description

Das `create_test` Tool erstellt ein neues Lasttestszenario, ohne es auszuführen. Der Test wird gespeichert und kann später mit ausgeführt werden`start_run`. Rufen Sie bei skriptbasierten Tests (jmeter, k6, locust) `upload_test_script` zuerst auf und übergeben Sie den zurückgegebenen. `test_id`

Parameters

`test_id`

- Die eindeutige Kennung des Testszenarios. Bei einfachen HTTP-Tests weglassen (das System generiert einen). Erforderlich für skriptbasierte Tests — verwenden Sie den `test_id` zurückgegebenen Wert von. `upload_test_script`

  Typ: Zeichenfolge

  Erforderlich: Nein (für skriptbasierte Tests erforderlich)

`test_name`

- Human-readable Name für das Testszenario

Typ: Zeichenfolge

Erforderlich: Ja

`test_description`

- Beschreibung dessen, was dieser Test validiert

  Typ: Zeichenfolge

  Erforderlich: Ja

`test_type`

- Art des Tests. `simple`für inline konfigurierte HTTP-Endpunkttests. `jmeter`,`k6`, oder `locust` für skriptbasierte Tests, die auf eine hochgeladene Skriptdatei verweisen.

  Typ: Zeichenfolge

  Erforderlich: Ja

`test_task_configs`

- Regionale Aufgabenkonfiguration. Jeder Eintrag gibt eine Region, die Anzahl der AWS Fargate-Aufgaben und gleichzeitige virtuelle Benutzer pro Aufgabe an. Gesamtzahl gleichzeitiger Benutzer für eine Region = ×. `task_count` `concurrency`

  Typ: Anordnung von Objekten (jeweils mit`region`,`task_count`,`concurrency`)

  Erforderlich: Ja

`test_scenario`

- Testausführungsszenario, das das Lastprofil und die Zielendpunkte definiert. Enthält `execution` (Ramp-up, Hold-for, Szenarioname) und `scenarios` (benannte Szenariodefinitionen mit entweder einem `requests` Array für einfache Tests oder einer `script` Zeichenfolge für skriptbasierte Tests).

  Typ: Objekt

  Erforderlich: Ja

`show_live`

- Ob die Live-Überwachung während der Testausführung aktiviert werden soll.

  Typ: Boolescher Wert

Standard: `false`

Erforderlich: Nein

`tags`

- Tags für die Organisation von Testszenarien. Maximal 5 Tags.

  Typ: Zeichenfolgen-Array

  Erforderlich: Nein

`native_run_mode`

- Ein Objekt, das den Verkehrsformmodus auswählt. Lassen Sie es im Standardmodus weg, in dem die Lösung die Last steuert. Schließen Sie es für den systemeigenen Modus ein, in dem Ihr hochgeladenes Skript den Ladevorgang steuert. Weitere Informationen finden Sie unter Traffic Shape-Modi.

  Typ: Objekt

  Erforderlich: Nein

Der native Modus unterscheidet sich wie folgt vom Standardmodus:

- Das Objekt benötigt ein Feld mit einer maximalen Laufzeit von 24 Stunden. `max_test_duration_seconds`
- Nur skriptbasierte Tests (`jmeter`k6, oder`locust`) akzeptieren den systemeigenen Modus.
- Einfache HTTP-Endpunkttests werden immer im Standardmodus ausgeführt.
- `test_task_configs`bleibt erforderlich, und jeder Eintrag ist weiterhin erforderlich`concurrency`.
- Eine Anfrage, die `concurrency` mit Erfolg beendet wird, `native_run_mode` gibt zurück.
- Die Last, die der Test generiert, ist die Last, die Ihr Skript deklariert.
- Die Gesamtlast pro Region ist die Last Ihres Skripts multipliziert mit. `task_count`

Antwort

Name	Description
`testId`	Die eindeutige ID des erstellten Tests

Name	Description
`testName`	Name des Tests
`status`	Status des Tests (zum Beispiel`created`)

update_test

Description

Das `update_test` Tool aktualisiert die Konfiguration eines vorhandenen Testszenarios. Dies ist ein vollständiger Ersatz — die gesamte Testkonfiguration muss bereitgestellt werden, nicht nur geänderte Felder. Der Test darf derzeit nicht ausgeführt werden.

Parameters

Entsprichtcreate_test, `test_id` ist jedoch erforderlich und muss auf einen vorhandenen Test verweisen.

Antwort

Name	Description
`testId`	Die eindeutige ID des aktualisierten Tests
`testName`	Name des Tests
`status`	Status des Tests

delete_test

Description

Das `delete_test` Tool löscht ein Testszenario und alle zugehörigen Daten, einschließlich Testlaufverlauf, Zeitpläne und CloudWatch Amazon-Dashboards, dauerhaft. Diese Aktion kann nicht rückgängig gemacht werden. Der Test darf derzeit nicht ausgeführt werden.

Parameters

`test_id`

- Die eindeutige Kennung des Testszenarios

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`status`	Bestätigung des Löschvorgangs

start_run

Description

Das `start_run` Tool startet die Ausführung eines Testszenarios. Der MCP-Server ruft die gespeicherte Konfiguration des Tests ab und löst die Ausführung aus. Kehrt sofort mit Status zurück. `queued` Wird verwendet`get_latest_test_run`, um den Abschluss abzufragen.

Parameters

`test_id`

- Die eindeutige Kennung des Testszenarios

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`testId`	Die eindeutige ID des Tests

Name	Description
`status`	Status des Tests (zum Beispielqueued)

stop_run

Description

Das `stop_run` Tool stoppt einen aktuell laufenden Test. Sendet ein Abbruchsignal an alle laufenden Fargate-Aufgaben. Der Teststatus wechselt zu`cancelled`. Teilergebnisse sind verfügbar unter`get_latest_test_run`.

Parameters

`test_id`

- Die eindeutige Kennung des Testszenarios

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`status`	Bestätigung der Stornierung

create_simple_schedule

Description

Das `create_simple_schedule` Tool erstellt einen einmaligen geplanten Test, der automatisch an einem bestimmten Datum und zu einer bestimmten Uhrzeit ausgeführt wird. Erfordert alle Felder für die Standardtestkonfiguration sowie Zeitplanfelder.

Parameters

Alle `create_test` Parameter (mit `test_id` optionalen, gleichen Regeln) plus:

`schedule_date`

- Datum für den geplanten Lauf. Muss in der future sein.

  Typ: Zeichenfolge (Format:`YYYY-MM-DD`)

  Erforderlich: Ja

`schedule_time`

- Zeit für den geplanten Lauf.

  Typ: Zeichenfolge (Format:`HH:MM`, 24 Stunden)

  Erforderlich: Ja

`schedule_timezone`

- IANA-Zeitzone für die Interpretation von Zeitplänen (z. B.`America/New_York`,`UTC`).

  Typ: Zeichenfolge

  Standard: `UTC`

  Erforderlich: Nein

Antwort

Name	Description
`testId`	Die eindeutige ID des Tests
`status`	Status des Tests (zum Beispiel`scheduled`)
`nextRun`	Nächster geplanter Ausführungszeitpunkt

create_cron_schedule

Description

Das `create_cron_schedule` Tool erstellt einen wiederkehrenden geplanten Test, der automatisch gemäß einem Cron-Ausdruck ausgeführt wird. Erfordert alle Felder für die Standardtestkonfiguration sowie Felder für den Cron-Zeitplan.

Parameters

Alle `create_test` Parameter (mit `test_id` optionalen, gleichen Regeln) plus:

`cron_value`

- Cron-Ausdruck für einen wiederkehrenden Zeitplan. Standardformat mit 5 Feldern (z. `0 9 * * *` B. täglich um 9:00 Uhr).

  Typ: Zeichenfolge

  Erforderlich: Ja

`recurrence`

- Human-readable Bezeichnung für Wiederholungen (z. B.`daily`,`weekly`).

  Typ: Zeichenfolge

  Erforderlich: Ja

`cron_expiry_date`

- Datum, an dem der wiederkehrende Zeitplan nicht mehr ausgeführt wird.

  Typ: Zeichenfolge (Format:`YYYY-MM-DD`)

  Erforderlich: Nein

`schedule_timezone`

- IANA-Zeitzone für die Interpretation von Zeitplänen.

  Typ: Zeichenfolge

  Standard: `UTC`

  Erforderlich: Nein

Antwort

Name	Description
`testId`	Die eindeutige ID des Tests
`status`	Status des Tests (zum Beispiel`scheduled`)

Name	Description
`nextRun`	Nächster geplanter Ausführungszeitpunkt

update_simple_schedule

Description

Das `update_simple_schedule` Tool aktualisiert die Zeitplankonfiguration für einen vorhandenen einmaligen geplanten Test. Vollständiger Ersatz der Testkonfiguration einschließlich der Zeitplanfelder. Der Test muss sich im `scheduled` Status befinden.

Parameters

Entsprich`create_simple_schedule`, `test_id` ist jedoch erforderlich und muss auf einen vorhandenen geplanten Test verweisen.

Antwort

Entspricht `create_simple_schedule`.

update_cron_schedule

Description

Das `update_cron_schedule` Tool aktualisiert die Zeitplankonfiguration für einen vorhandenen wiederkehrenden geplanten Test. Vollständiger Ersatz der Testkonfiguration, einschließlich der Felder für den Cron-Zeitplan. Der Test muss sich im `scheduled` Status befinden.

Parameters

Entsprich`create_cron_schedule`, `test_id` ist jedoch erforderlich und muss auf einen vorhandenen geplanten Test verweisen.

Antwort

Entspricht `create_cron_schedule`.

upload_test_script

Description

Das `upload_test_script` Tool lädt eine Skriptdatei (JMeter`.jmx`, k6, Locust `.py` oder) hoch, die für `.js` skriptbasierte Tests erforderlich ist. `.zip` Muss vor oder für skriptbasierte Tests aufgerufen werden. `create_test` `update_test` Gibt ein `test_id` und zurück, `script_filename` das in nachfolgenden Toolaufrufen verwendet werden kann.

Parameters

`test_id`

- Die eindeutige Kennung des Testszenarios. Bei neuen Tests weglassen (das System generiert einen). Sorgen Sie dafür, dass vorhandene Tests an den richtigen Ort hochgeladen werden.

  Typ: Zeichenfolge

  Erforderlich: Nein

`test_type`

- Testtyp:`jmeter`,`k6`, oder`locust`.

  Typ: Zeichenfolge

  Erforderlich: Ja

`file_extension`

- Dateierweiterung:`jmx`,`js`,`py`, oder`zip`.

  Typ: Zeichenfolge

  Erforderlich: Ja

`file_content`

- Base64-encoded Inhalt der Datei.

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`test_id`	Die Test-ID (generiert oder bereitgestellt)
`script_filename`	Dateiname in S3 (Format:`<test_id>.<extension>`). Verweisen Sie darauf in `test_scenario.scenarios`.

Workflow-Anleitungen

Workflow-Anleitungen sind mehrstufige Anleitungen, die Agenten dabei helfen, mehrere Tools für gemeinsame Aufgaben miteinander zu verknüpfen. Das `get_workflow_guides` Tool bietet strukturierte, schrittweise Anleitungen für jeden Workflow.

get_workflow_guides

Description

Das `get_workflow_guides` Tool gibt schrittweise Workflow-Rezepte für gängige DLT-Operationen mit mehreren Tools zurück. Gibt strukturierte Anleitungen dazu zurück, welche Tools in welcher Reihenfolge aufgerufen werden müssen und wie die Ergebnisse zwischen den Schritten zu interpretieren sind.

Parameters

`workflow`

- Der Workflow, für den Anleitungen abgerufen werden sollen. Einer von:`run_and_monitor`,`baseline_comparison`,`schedule_test`,`create_and_run`,`update_and`

  Typ: Zeichenfolge

  Erforderlich: Ja

Antwort

Name	Description
`workflow`	Workflow-ID
`description`	Kurze Beschreibung des Zwecks des Workflows
`steps`	Anordnung von Step-Objekten, jeweils mit `step` (Nummer), `action` (was zu tun ist), `tool` (welches MCP-Tool aufgerufen werden soll, oder Null für Schritte, die keine Tools sind) und `details` (spezifische Anweisungen)

Verfügbare Workflows

run_and_monitor

Startet einen bestehenden Testlauf und fragt bis zum Abschluss ab.

1. Suchen Sie den Test mit `list_scenarios` oder `get_scenario_details`
2. Starten Sie den Testlauf mit `start_run`
3. Abfrage zur Fertigstellung mit `get_latest_test_run` (empfohlenes Intervall: 30 Sekunden; erste 404-Fehler 1—3 Minuten lang bearbeiten, während Amazon Elastic Container Service (Amazon ECS) -Aufgaben gestartet werden)
4. Ergebnisse melden, sobald ein Terminalstatus erreicht ist (`completefailed`, oder`cancelled`)

Baseline_Vergleich

Führen Sie einen Test aus und vergleichen Sie die Ergebnisse mit einer gespeicherten Basislinie.

1. Suchen Sie den Test mit `list_scenarios` oder `get_scenario_details`
2. Starten Sie den Testlauf mit `start_run`
3. Abfrage zur Fertigstellung mit `get_latest_test_run` (empfohlenes Intervall: 30 Sekunden)
4. Rufen Sie die Basislinie ab mit `get_baseline_test_run` (Vergleich überspringen, wenn keine Basislinie festgelegt ist)

5. Metriken vergleichen (durchschnittliche Antwortzeit, Latenz, Durchsatz, Perzentile, Fehlerrate)

schedule_test

Erstellen Sie einen Test mit einem wiederkehrenden oder einmaligen Zeitplan.

1. Ermitteln Sie den Zeitplantyp (einmalig →`create_simple_schedule`, wiederkehrend →`create_cron_schedule`)
2. Laden Sie das Testskript hoch, falls es skriptbasiert ist `upload_test_script`
3. Erstellen Sie den geplanten Test mit vollständiger Konfiguration und Zeitplanfeldern
4. Vergewissern Sie sich, dass der Zeitplan mit `get_scenario_details` (Häkchen `status: scheduled` und`nextRun`) erstellt wurde

Einschränkungen: Mindestintervall von 1 Stunde zwischen wiederkehrenden Läufen, das Intervall muss die Testdauer überschreiten, Cron muss genau einen Minutenwert angeben.

create_and_run

Erstellen Sie einen neuen Test von Grund auf und führen Sie ihn sofort aus.

1. Laden Sie das Testskript hoch, falls es skriptbasiert ist mit `upload_test_script`
2. Erstellen Sie den Test mit `create_test`
3. Starten Sie den Testlauf `start_run` mit dem zurückgegebenen `test_id`
4. Abfrage zur Fertigstellung mit `get_latest_test_run` (empfohlenes Intervall: 30 Sekunden)
5. Ergebnisse melden

update_and_run

Ändern Sie die Konfiguration eines vorhandenen Tests und führen Sie ihn sofort erneut aus.

1. Rufen Sie die aktuelle Konfiguration ab mit `get_scenario_details`
2. Laden Sie ein neues Skript hoch, wenn Sie das Skript ändern mit `upload_test_script`
3. Aktualisieren Sie die Testkonfiguration mit `update_test` (vollständiger Ersatz — alle Felder einschließen)
4. Starten Sie den Testlauf mit `start_run`
5. Abfrage zur Fertigstellung mit `get_latest_test_run` (empfohlenes Intervall: 30 Sekunden)

6. Ergebnisse melden

> **Note**
>
> Alle MCP-Tools nutzen bestehende API-Endpunkte. Zur Unterstützung der MCP-Funktionalität sind keine Änderungen an den zugrunde liegenden APIs erforderlich.

DLT-CLI

Mit der DLT-CLI können Sie direkt von Ihrem Terminal aus mit der Distributed Load Testing on AWS-REST-API interagieren. Sie ermöglicht die skriptgesteuerte Automatisierung von Lasttests, sodass Sie Testszenarien auflisten, überprüfen und starten sowie Ausführungsergebnisse als Teil wiederholbarer Workflows abfragen können.

Die CLI zielt auf dasselbe IAM-authorized API Gateway ab wie die DLT-Webkonsole, sodass alle in der browserbasierten Oberfläche verfügbaren Operationen auch von der Befehlszeile aus mit derselben zugrunde liegenden API und demselben Berechtigungsmodell ausgeführt werden können.

Der CLI-Quellcode und die vollständige Dokumentation befinden sich im GitHub Repository der Lösung unter dem `source/cli` Verzeichnis. Die DLT-CLI-README-Datei auf der GitHub Website ist die maßgebliche Referenz für Installation, Konfiguration, Authentifizierung und die vollständige Befehlsreferenz.

Wichtigste Funktionen

Befehls-Gruppe	Description
`configure`	Konfigurieren Sie die CLI mit Ihren DLT-Stack-Einstellungen, entweder interaktiv oder durch Importieren einer `aws-exports.json` Datei.
`login`	Authentifizieren Sie sich beim DLT-Dienst und holen Sie sich die Anmeldeinformationen, die zum Aufrufen der API erforderlich sind.
`logout`	Entfernen Sie die gespeicherten Anmeldeinformationen vom lokalen Computer.

Befehls-Gruppe	Description
`token`	Überprüfen Sie die Authentifizierungstoken und den Ablaufstatus der Anmeldeinformationen und geben Sie sie aus.
`scenarios`	Testszenarien erstellen, aktualisieren, auflisten, Details anzeigen, starten (erneut ausführen), abbrechen und löschen. Creation unterstützt einfache HTTP-Tests, Uploads von Skriptdateien (JMeter, k6, Locust) und wiederkehrende Zeitpläne . Der `results` Unterbefehl ruft die Metriken des letzten abgeschlossenen Laufs ab. Er kann CI/CD Pipelines auch mithilfe von Schwellenwert-Flags abzweigen.
`runs`	Testlaufergebnisse abfragen, Ausführungsdetails anzeigen, Testlaufverlauf löschen, Basisläufe verwalten (festlegen, abrufen und löschen), Läufe mit einer Baseline vergleichen und Ausführungsartefakte herunterladen.

Die CLI unterstützt drei Authentifizierungsmodi: browserbasierte Anmeldung mit dem Proof Key for Code Exchange (PKCE) -Flow über die Cognito Hosted UI, Headless Secure Remote Password (SRP) -Authentifizierung und direkte Authentifizierung mit IAM-Anmeldeinformationen. Vollständige Einrichtungsdetails zu den einzelnen Modi finden Sie in der DLT-CLI-README-Datei auf der GitHub Website.

Die `scenarios results` Befehle `scenarios start --wait` und unterstützen auch Schwellenwert-Flags (z. B. Fehlerrate, p95/p99 Latenz, Durchsatz und Baseline-Regression). Diese Flags verwandeln die Ergebnisse von Lasttests in pass/fail CI/CD Pipeline-Gates und verwenden eindeutige Ausgangscodes, um einen Verstoß zu signalisieren. Datenbefehle unterstützen zusätzlich`table`,`json`, und `csv` Ausgabeformate. Die vollständige Referenz zu Flag und Exit-Code finden Sie in der DLT-CLI-README-Datei auf der Website. GitHub

Installation und ausführliche Referenz

Die CLI wird als portables Node.js Paket vertrieben, das direkt ausgeführt werden kann, oder Sie können sie mithilfe des Repositorys aus dem Quellcode erstellen. Informationen zu den ersten Schritten finden Sie in der DLT-CLI-README-Datei auf der GitHub Website, die Folgendes enthält:

- Installationsanweisungen (portables Paket und Build-from-source)

- Konfiguration und Authentifizierungseinrichtung für alle drei Authentifizierungsmodi
- Vollständige Befehlsreferenz mit Anwendungsbeispielen

Schnellstart

> **Note**
>
> Bei diesem Schnellstart wird davon ausgegangen, dass der DLT-Stack bereits bereitgestellt ist.

Option 1: Tragbares Paket (empfohlen)

Der schnellste Weg, um loszulegen, ist das portable Bundle. Dies erfordert weder das Klonen des Repositorys noch irgendwelche Build-Schritte:

```
# Download the portable bundle
curl -sLo /usr/local/bin/dlt \
  https://raw.githubusercontent.com/aws-solutions/distributed-load-testing-on-aws/main/
deployment/cli/dlt-cli.mjs

# Make it executable
chmod +x /usr/local/bin/dlt

# Point the CLI at your deployed DLT stack
dlt configure

# Authenticate with the DLT service
dlt login

# Run a command (for example, list test scenarios)
dlt scenarios list
```

Option 2: Aus dem Quellcode erstellen

Wenn Sie lieber aus dem Quellcode bauen möchten, klonen Sie zuerst das Repository. Das Projekt verwendet NPM-Workspaces. Führen Sie daher die folgenden Befehle im Stammverzeichnis des Repositorys aus:

```
# Install all workspace dependencies
```

```
npm install

# Build the CLI
npm run build -w source/cli

# Link the dlt command into your PATH
npm link -w source/cli

# Point the CLI at your deployed DLT stack
dlt configure

# Authenticate with the DLT service
dlt login

# Run a command (for example, list test scenarios)
dlt scenarios list
```

Referenz zum Cron-Ausdruck

Diese Lösung verwendet eine Teilmenge des standardmäßigen Linux-Cron-Formats mit 5 Feldern für wiederkehrende Testpläne. Der Cron-Ausdruck besteht aus fünf Feldern, die durch Leerzeichen getrennt sind.

```
########## minute (0-59)
# ######## hour (0-23, *, */N, or comma list)
# # ###### day of month (1-31 or *)
# # # #### month (1-12 or *)
# # # # ## day of week (0-6, *, range, or list)
# # # # #
0 9 * * 1-5
```

Akzeptierte Werte

In der folgenden Tabelle wird beschrieben, was jedes Feld akzeptiert.

Feld	Akzeptierte Werte	Beispiele
Minute	Ein einzelner Wert zwischen 0 und 59.	`0`, `30`, `45`

Feld	Akzeptierte Werte	Beispiele
Stunde	(jede Stunde), ein Schrittwert, ein einzelner Wert zwischen 0 und 23 oder eine durch Kommas getrennte Werteliste.	, , 9, 9,17
Tag des Monats	*(jeden Tag) oder ein einzelner Wert zwischen 1 und 31.	*, 1, 15, 31
Monat	*(jeden Monat) oder ein einzelner Wert von 1 bis 12.	*, 1, 6, 12
Wochentag	*(jeden Tag), ein einzelner Wert zwischen 0 und 6, ein Bereich mit einem Bindestrich oder eine durch Kommas getrennte Liste.	*, 0, 1-5, 0,6

Für Wochentagswerte wird die folgende Zuordnung verwendet: 0 = Sonntag, 1 = Montag, 2 = Dienstag, 3 = Mittwoch, 4 = Donnerstag, 5 = Freitag, 6 = Samstag.

Nicht unterstützte Muster

Die folgenden Muster sind eine gültige Linux-Cron-Syntax, werden jedoch von dieser Lösung nicht unterstützt.

Muster	Beispiel	Grund
Werte in Minutenschritten	*/15 * * * *	Das minimale Planungsintervall beträgt eine Stunde.
Minutenlisten	0,30 * * * *	Das Mindestplanungsintervall beträgt eine Stunde.
Der Tag des Monats reicht	0 9 1-15 * *	Es wird nur ein einziger Tageswert oder Platzhalter akzeptiert.

Muster	Beispiel	Grund
Listen mit den Wochentagen	`0 9 1,15 * *`	Es wird nur ein einziger Tageswert oder Platzhalter akzeptiert.
Monatsbereiche	`0 9 * 3-9 *`	Es wird nur ein einziger Monatswert oder Platzhalter akzeptiert.
Monatslisten	`0 9 1 1,6,12 *`	Es wird nur ein einziger Monatswert oder Platzhalter akzeptiert.
Fragezeichen	`0 9 ? * ?`	Das Fragezeichen ist im Standard-Linux-Cron nicht gültig. Verwenden Sie stattdessen `*`.

Voreingestellte Muster

Die Webkonsole bietet die folgenden voreingestellten Muster, die Sie direkt auswählen können.

Name des Musters	Expression	Description
Stündlich	`0 * * * *`	Läuft bei Minute 0 jeder Stunde.
Täglich um 9:00 Uhr	`0 9 * * *`	Läuft einmal täglich um 9:00 Uhr in der ausgewählten Zeitzone.
Wochentags um 8:00 Uhr	`0 8 * * 1-5`	Läuft von Montag bis Freitag um 8:00 Uhr.
Jeden Sonntag um 17 Uhr	`0 17 * * 0`	Läuft jeden Sonntag um 17:00 Uhr.
1. des Monats um 11 Uhr	`0 11 1 * *`	Läuft am ersten Tag eines jeden Monats um 11:00 Uhr.

Einschränkungen bei der Terminplanung

- Das Mindestintervall zwischen geplanten Testläufen beträgt eine Stunde. Das System überprüft, ob das Intervall zwischen aufeinanderfolgenden Testläufen länger als die geschätzte Testdauer ist.
- Für alle wiederkehrenden Zeitpläne ist ein Ablaufdatum erforderlich. Tests werden nach Ablauf des Ablaufdatums (UTC) nicht mehr ausgeführt.
- Die Zeitzone des Zeitplans bestimmt, wann der Cron ausgelöst wird. Die Umstellung auf Sommerzeit erfolgt automatisch. Wenn aufgrund einer Umstellung auf die Sommerzeit keine geplante Zeit existiert, wird dieser Vorgang übersprungen.
- Das System validiert den Cron-Ausdruck sowohl auf der Webkonsole als auch auf der API. Wenn der Ausdruck nicht dem akzeptierten Format entspricht, kann der Test nicht erstellt werden.

Referenz

Dieser Abschnitt enthält Informationen zur Datenerfassung, Hinweise auf verwandte Ressourcen und eine Liste der Entwickler, die zu dieser Lösung beigetragen haben.

Datenerfassung

Diese Lösung sendet Betriebsmetriken (die „Daten“) über die Verwendung dieser Lösung an AWS. Wir verwenden diese Daten, um besser zu verstehen, wie Kunden diese Lösung und die damit verbundenen Dienstleistungen und Produkte nutzen. Die Erfassung dieser Daten durch AWS unterliegt der AWS-Datenschutzerklärung.

Mitwirkende

- Tom Nightingale
- Fernando Dingler
- Beomseok Lee
- George Lenz
- Erin McGill
- Dmitri López
- Kamyar Ziabari
- Bassem Wanis
- Garvit Singh
- Nikhil Reddy
- Simon Kroll
- Ahern Knox
- Ian Downard
- Owen Brady
- Jim Thario
- Thyag Ramachandran
- Yang Qin
- James Wang

- Brad Hong
- Colin McCoy
- David Chung
- Jiali Zhang

Glossar

In diesem Glossar werden die im Implementierungsleitfaden für Distributed Load Testing on AWS verwendeten Akronyme und Abkürzungen definiert.

Technische Protokolle und Formate

AGPL

Allgemeine öffentliche Lizenz von Affero. Eine von k6 verwendete Open-Source-Softwarelizenz.

API

Schnittstelle zur Anwendungsprogrammierung. Eine Reihe von Protokollen und Tools zum Erstellen von Softwareanwendungen und zum Ermöglichen der Kommunikation zwischen verschiedenen Systemen.

CLI

Befehlszeilenschnittstelle. Eine textbasierte Oberfläche für die Interaktion mit Software und Betriebssystemen.

KERNE

Cross-Origin Gemeinsame Nutzung von Ressourcen. Eine Sicherheitsfunktion, die Webanwendungen, die an einem Ursprung ausgeführt werden, den Zugriff auf Ressourcen von einem anderen Ursprung ermöglicht oder einschränkt.

CSV

Comma-Separated Werte. Ein Dateiformat, das zum Speichern von Tabellendaten im Klartext verwendet wird und häufig für den Datenexport verwendet wird.

gRPC

gRPC Remote Procedure Call. Ein leistungsstarkes Open-Source-Framework für Remote-Prozeduraufrufe.

HTTP

Hypertext-Übertragungsprotokoll. Das Basisprotokoll, das für die Übertragung von Daten im World Wide Web verwendet wird.

HTTPS

HTTP Secure. Eine Erweiterung von HTTP, die Verschlüsselung für die sichere Kommunikation über ein Netzwerk verwendet.

JSON

JavaScript Objektnotation. Ein leichtes Datenaustauschformat, das für Menschen leicht zu lesen und zu schreiben und für Maschinen einfach zu analysieren und zu generieren ist.

JWT

JSON-Webtoken. Ein kompaktes URL-safe Mittel zur Darstellung von Ansprüchen, die zur Authentifizierung und Autorisierung zwischen zwei Parteien übertragen werden sollen.

OAuth

Öffnen Sie die Autorisierung. Ein offener Standard für die Zugriffsdelegierung, der häufig für die tokenbasierte Authentifizierung und Autorisierung verwendet wird.

REST

Repräsentativer Staatstransfer. Ein Architekturstil für den Entwurf von Netzwerkanwendungen mit zustandsloser Kommunikation und Standard-HTTP-Methoden.

SSE

Server-Sent Ereignisse. Eine Server-Push-Technologie, die es einem Client ermöglicht, automatische Updates von einem Server über eine HTTP-Verbindung zu erhalten.

Benutzeroberfläche

Benutzerschnittstelle. Die visuellen Elemente und Steuerelemente, über die Benutzer mit Softwareanwendungen interagieren.

URL

Einheitlicher Resource Locator. Die Adresse, die für den Zugriff auf Ressourcen im Internet verwendet wird.

XML

Erweiterbare Auszeichnungssprache. Eine Auszeichnungssprache, die Regeln für die Kodierung von Dokumenten in einem Format definiert, das sowohl für Menschen als auch für Maschinen lesbar ist.

Begriffe zum Testen und zur Datenbank

FTP

Dateiübertragungsprotokoll. Ein Standard-Netzwerkprotokoll, das für die Übertragung von Dateien zwischen einem Client und einem Server verwendet wird.

GSI

Globaler sekundärer Index. Eine DynamoDB-Funktion, mit der Sie Daten mithilfe eines alternativen Schlüssels abfragen können.

JDBC

Java-Datenbankkonnektivität. Eine Java-API zum Verbinden und Ausführen von Abfragen mit Datenbanken.

JMS

Java-Nachrichtendienst. Eine Java-API zum Senden von Nachrichten zwischen zwei oder mehr Clients.

TPS

Transaktionen pro Sekunde. Ein Maß für die Anzahl der Transaktionen, die ein System in einer Sekunde verarbeiten kann.

AWS- und Systembedingungen

ARN

Amazon-Ressourcenname. Eine eindeutige Kennung für AWS-Ressourcen, die zur Spezifizierung von Ressourcen für alle AWS-Services verwendet wird.

ISO

Internationale Organisation für Standardisierung. Eine unabhängige Nichtregierungsorganisation, die internationale Standards entwickelt. In diesem Handbuch wird auf das ISO 8601-Zeitstempelformat verwiesen.

SLA

Vereinbarung zum Servicelevel. Eine Verpflichtung zwischen einem Dienstanbieter und einem Kunden, die das erwartete Serviceniveau definiert.

UUID

Universell eindeutiger Bezeichner. Eine 128-Bit-Nummer, die zur eindeutigen Identifizierung von Informationen in Computersystemen verwendet wird.

vCPU

Virtuelle Zentraleinheit. Ein virtueller Prozessor, der einer virtuellen Maschine oder einem Container zugewiesen ist und einen Teil der Rechenleistung der physischen CPU darstellt.

Bedingungen für den Belastungstest

concurrency

Die Anzahl gleichzeitiger virtueller Benutzer pro Aufgabe. Dieser Parameter steuert, wie viele simulierte Benutzer jede Fargate-Aufgabe während eines Auslastungstests generiert. Parallelität gilt für den Standardmodus. Im systemeigenen Modus legt das Skript seine eigene Anzahl virtueller Benutzer fest.

Systemeigener Modus

Der Traffic Shape-Modus, in dem Ihr hochgeladenes Skript die Last steuert. In diesem Modus führt die Lösung das Skript unter der eigenen Befehlszeile des Test-Frameworks aus und übergibt keine Ladeparameter. Es legt nur die Anzahl der Aufgaben pro Region und eine Sicherheitsdauer fest.

regionaler Stapel

Ein CloudFormation Stack, der in einer AWS-Region bereitgestellt wird, um eine Testinfrastruktur für Multi-Region Auslastungstests bereitzustellen.

Dauer der Sicherheit

Im systemeigenen Modus beträgt die maximale Dauer, die ein Test ausgeführt werden darf, bis zu 24 Stunden. Wenn der Test nach Ablauf der Dauer noch läuft, stoppt die Lösung das Test-Framework. Es behält die Ergebnisse für den Teil bei, der ausgeführt wurde, und zeichnet den Testlauf als abgeschlossen auf.

Standardmodus

Der standardmäßige Traffic Shape-Modus, in dem die Lösung die Last steuert. Sie legen die Anzahl der Aufgaben, die Parallelität, die Anlaufzeit und die Haltedauer fest. Die Lösung wendet

sie über das Taurus Automation Framework an und überschreibt dabei die von Ihrem Skript deklarierte Last.

Anzahl der Aufgaben

Die Anzahl der Fargate-Container (Aufgaben), die zur Ausführung eines Testszenarios gestartet wurden. Im Standardmodus entspricht die generierte Gesamtlast der Anzahl der Aufgaben multipliziert mit der Parallelität. Im systemeigenen Modus führt jede Aufgabe eine vollständige Kopie Ihres Skripts aus. Die Gesamtlast entspricht der Anzahl der Aufgaben multipliziert mit der vom Skript deklarierten Last.

Testszenario

Ein konfigurierter Auslastungstest, der Testtyp, Traffic-Shape-Modus, Zielendpunkte, Anzahl der Aufgaben, Parallelität, Dauer und andere Parameter umfasst.

Traffic Shape-Modus

Welche Seite steuert die Last, die ein Test generiert, entweder im Standardmodus oder im systemeigenen Modus.

Überarbeitungen

Besuchen Sie CHANGELOG.md in unserem GitHub Repository, um versionsspezifische Verbesserungen und Korrekturen nachzuverfolgen.

Hinweise

Kunden sind dafür verantwortlich, Ihre eigene unabhängige Bewertung der Informationen in diesem Dokument vorzunehmen. Dieses Dokument: (a) dient nur zu Informationszwecken, (b) stellt aktuelle Produktangebote und Praktiken von AWS dar, die ohne vorherige Ankündigung geändert werden können, und (c) stellt keine Verpflichtungen oder Zusicherungen von AWS und seinen verbundenen Unternehmen, Lieferanten oder Lizenzgebern dar. AWS-Produkte oder -Services werden „wie sie sind“ ohne ausdrückliche oder stillschweigende Garantien, Zusicherungen oder Bedingungen jeglicher Art bereitgestellt. Die Verantwortlichkeiten und Verbindlichkeiten von AWS gegenüber seinen Kunden werden durch AWS-Verträge geregelt. Dieses Dokument ist weder Teil einer Vereinbarung zwischen AWS und seinen Kunden noch ändert es diese.

Distributed Load Testing auf AWS ist unter den Bedingungen der Apache License Version 2.0 lizenziert, die bei The Apache Software Foundation erhältlich ist.

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.