

Leitfaden zur Implementierung

# Verteilte Lasttests auf AWS



# Verteilte Lasttests auf AWS: Leitfaden zur Implementierung

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Die Handelsmarken und Handelsaufmachung von Amazon dürfen nicht in einer Weise in Verbindung mit nicht von Amazon stammenden Produkten oder Services verwendet werden, durch die Kunden irregeführt werden könnten oder Amazon in schlechtem Licht dargestellt oder diskreditiert werden könnte. Alle anderen 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 .....	3
Anwendungsfälle .....	4
Konzepte und Definitionen .....	5
Übersicht über die Architektur .....	7
Architekturdiagramm .....	7
Überlegungen zum AWS-Well-Architected-Design .....	8
Operative Exzellenz .....	8
Sicherheit .....	9
Zuverlässigkeit .....	10
Leistungseffizienz .....	10
Kostenoptimierung .....	10
Nachhaltigkeit .....	11
Einzelheiten zur Architektur .....	12
Frontend .....	12
Belastungstest-API .....	12
Web-Konsole .....	13
Backend .....	13
Pipeline für Container-Images .....	13
Infrastruktur testen .....	14
Test-Engine .....	14
AWS-Services in dieser Lösung .....	15
So funktioniert Distributed Load Testing auf AWS .....	16
Designüberlegungen .....	18
Unterstützte Anwendungen .....	18
JMeter Unterstützung von Skripten .....	18
Unterstützung für K6-Skripte .....	19
Unterstützung für Locust-Skripte .....	19
Planung von Tests .....	20
Gleichzeitige Tests .....	20
Benutzerverwaltung .....	21
Regionale Bereitstellung .....	21
Planen Sie Ihren Einsatz .....	22

Kosten .....	22
Sicherheit .....	23
IAM-Rollen .....	24
Amazon CloudFront .....	24
AWS Fargate-Sicherheitsgruppe .....	24
Netzwerk-Stresstest .....	25
Beschränkung des Zugriffs auf die öffentliche Benutzeroberfläche .....	25
Unterstützte AWS Regionen .....	25
Kontingente .....	26
Kontingente für AWS-Services in dieser Lösung .....	26
CloudFormation AWS-Kontingente .....	26
Kontingente für Lasttests .....	26
Gleichzeitige Tests .....	20
EC2 Testrichtlinien von Amazon .....	27
Amazon-Richtlinie für CloudFront Auslastungstests .....	27
Überwachung der Lösung nach der Bereitstellung .....	28
CloudWatch Alarme aktivieren oder einrichten .....	28
Stellen Sie die Lösung bereit .....	29
Überblick über den Bereitstellungsprozess .....	29
CloudFormation AWS-Vorlage .....	29
Starten des -Stacks .....	30
Bereitstellung in mehreren Regionen .....	34
Aktualisieren Sie die Lösung .....	38
Bei der Aktualisierung von DLT-Versionen, die älter als v3.2.6 sind, auf v3.3.x und v3.3.x auf die neueste Version schlägt die Aktualisierung des Stacks fehl .....	39
Fehlerbehebung .....	41
Lösung eines bekannten Problems .....	41
Kontaktieren Sie AWS Support. ....	44
Fall erstellen .....	44
Wie können wir helfen? .....	44
Zusätzliche Informationen .....	44
Helfen Sie uns, Ihren Fall schneller zu lösen .....	44
Löse jetzt oder kontaktiere uns .....	45
Deinstallieren Sie die Lösung .....	46
Verwendung der AWS-Managementkonsole .....	46
Verwenden der AWS-Befehlszeilenschnittstelle .....	46

Löschen der Amazon S3 S3-Buckets .....	46
Benutze die Lösung .....	48
Testergebnisse .....	48
Arbeitsablauf bei der Testplanung .....	49
Ermitteln Sie die Anzahl der Benutzer .....	49
Live-Daten .....	50
Testen Sie den Stornierungs-Workflow .....	51
Entwicklerhandbuch .....	52
Quellcode .....	52
Wartung .....	52
Versionen .....	52
Anpassung des Container-Images .....	53
API für verteilte Lasttests .....	60
GET /scenarios .....	62
POST /szenarien .....	62
OPTIONEN/Szenarien .....	64
GET /scenarios/ {testId} .....	64
POST /scenarios/ {TestID} .....	66
LÖSCHEN SIE /scenarios/ {testId} .....	66
OPTIONEN /scenarios/ {testId} .....	67
GET /tasks .....	68
OPTIONEN/Aufgaben .....	69
GET /regions .....	69
OPTIONEN /Regionen .....	70
Erhöhen Sie die Container-Ressourcen .....	71
Erstellen Sie eine neue Revision der Aufgabendefinition .....	71
Aktualisieren Sie die DynamoDB-Tabelle .....	72
Referenz .....	73
Anonymisierte Datenerfassung .....	73
Mitwirkende .....	74
Überarbeitungen .....	75
Hinweise .....	76
.....	lxxvii

# Automatisieren Sie das Testen Ihrer Softwareanwendungen in großem Maßstab

Datum der Veröffentlichung: November 2019

Distributed Load Testing auf AWS hilft Ihnen dabei, das Testen Ihrer Softwareanwendungen im großen Maßstab und unter Last zu automatisieren, um Engpässe zu identifizieren, bevor Sie Ihre Anwendung veröffentlichen. Diese Lösung erstellt und simuliert Tausende von verbundenen Benutzern, die in konstantem Tempo Transaktionsdatensätze generieren, ohne dass Server bereitgestellt werden müssen.

Diese Lösung nutzt [Amazon Elastic Container Service \(Amazon ECS\) auf AWS Fargate](#), um Container bereitzustellen, die all Ihre Simulationen ausführen können und die folgenden Funktionen bieten:

- Stellen Sie Amazon ECS auf AWS Fargate-Containern bereit, die unabhängig voneinander ausgeführt werden können, um die Ladefähigkeit der getesteten Software zu testen.
- Simulieren Sie Zehntausende verbundener Benutzer in mehreren AWS-Regionen und generieren Sie kontinuierlich Transaktionsaufzeichnungen.
- [Passen Sie Ihre Anwendungstests an, indem Sie benutzerdefinierte JMeter Skripts erstellen.](#)
- Planen Sie Lasttests so, dass sie entweder automatisch zu einem future Zeitpunkt oder zu wiederkehrenden Terminen beginnen.
- Führen Sie Ihre Anwendungslasttests gleichzeitig oder mehrere Tests gleichzeitig aus.

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.	<a href="#">Kosten</a>
Machen Sie sich mit den Sicherheitsüberlegungen für diese Lösung vertraut.	<a href="#">Sicherheit</a>
Erfahren Sie, wie Sie Kontingente für diese Lösung einplanen.	<a href="#">Kontingente</a>
Erfahren Sie, welche AWS-Regionen diese Lösung unterstützen.	<a href="#">Unterstützte AWS-Regionen</a>
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.	<a href="#">CloudFormation AWS-Vorlage</a>
Greifen Sie auf den Quellcode zu und verwenden Sie optional das AWS Cloud Development Kit (AWS CDK), um die Lösung bereitzustellen.	<a href="#">GitHub Repositor y</a>

## Features

Die Lösung bietet die folgenden Funktionen:

Out-of-the-Box Konfigurierbare Leistungstests

Beinhaltet vorkonfigurierte Leistungstests, die sofort verwendet werden können.

Individuell anpassbare Anwendungstests

Ermöglicht eine flexible und präzise Testanpassung zur Identifizierung potenzieller Probleme und passt Tests mithilfe von JMeter Skripten an spezifische Anforderungen und Szenarien an.

Simuliert eine hohe Benutzerlast

Kann Zehntausende verbundener Benutzer simulieren, um Ihre Anwendung einem Stresstest zu unterziehen.

Kontinuierliche Generierung von Transaktionen

Generiert kontinuierlich Transaktionsdatensätze, um die Leistung bei konstanter Last zu bewerten.

### Überwachung in Echtzeit

Ermöglicht die Echtzeitüberwachung des Testfortschritts und der Ergebnisse und ermöglicht es Ihnen, Tests so zu planen, dass sie automatisch an bestimmten Terminen oder in wiederkehrenden Intervallen beginnen.

### Simulation regionaler Anfragen

Simulieren Sie Benutzeranfragen aus beliebigen Regionen, um die globale Leistung zu bewerten.

### Flexibilität der Endgeräte

Testen Sie jeden Endpunkt in AWS-Regionen, lokalen Umgebungen oder anderen Cloud-Anbietern.

### Detaillierte Testergebnisse

Sehen Sie sich umfassende Testergebnisse an, einschließlich durchschnittlicher Antwortzeit, Anzahl gleichzeitiger Benutzer, erfolgreicher Anfragen und fehlgeschlagener Anfragen.

### Intuitive Webkonsole

Bietet eine easy-to-use Webkonsole zur Verwaltung und Überwachung von Tests.

### Unterstützt mehrere Protokolle

Kompatibel mit verschiedenen Protokollen wie HTTP WebSocket, HTTPS, JDBC, JMS, FTP und gRPC.

## Vorteile

Die Lösung bietet die folgenden Vorteile:

Unterstützt umfassende Leistungstests

Erleichtert Last-, Stress- und Dauertests für eine gründliche Anwendungsevaluierung.

Früherkennung von Leistungsproblemen

Identifiziert Leistungsprobleme und Engpässe vor der Produktionsfreigabe.

Simulation der Nutzung in der realen Welt

Spiegelt reale Nutzungsmuster exakt wider, um Engpässe und Optimierungsbereiche aufzuzeigen.

Detaillierte Performance Insights

Bietet Einblicke in die Leistung und Stabilität von Software unter erheblicher Belastung.

Automatisierte Leistungsbewertung

Ermöglicht regelmäßige Leistungsbewertungen ohne manuelles Eingreifen.

Kosteneffizientes Testen

Bietet ein pay-as-you-go Modell, das die Notwendigkeit einer speziellen Testinfrastruktur und Abonnementgebühren überflüssig macht.

## Anwendungsfälle

Simulieren Sie die Produktionslast

Testen Sie Web- und Mobilanwendungen unter produktionsähnlichen Bedingungen, bevor Sie eine neue Version auf den Markt bringen.

Überprüfen Sie die Anwendungsleistung

Stellen Sie sicher, dass Ihre Anwendung den erwarteten Benutzerverkehr ohne Beeinträchtigung bewältigen kann. Testen Sie Anwendungslimits mithilfe von Standardressourcen und bewerten Sie die Skalierbarkeit der Infrastruktur.

Lastspitzen verwalten

Stellen Sie sicher, dass Ihre Infrastruktur Spitzenlasten oder unerwartete Verkehrsspitzen bewältigen kann, um Stabilität bei hoher Nachfrage zu gewährleisten.

Optimieren Sie die Leistung

Machen Sie sich mit dem Leistungsprofil Ihrer Anwendung vertraut und identifizieren Sie Engpässe wie ineffiziente Codeausführung, Datenbankabfragen und Netzwerklatenz.

Schneller Teststart

Beginnen Sie schnell mit dem Testen mit out-of-the-box Leistungstests.

## Individuell anpassbare Tests

Passen Sie die Tests an spezifische Szenarien und Anforderungen an und passen Sie die Anzahl der gleichzeitig ausgeführten Benutzer und Aufgaben an.

## Geplante Tests

Planen Sie Tests für Regressionstests und kontinuierliche Leistungsüberwachung ein, um eine konsistente Anwendungsleistung sicherzustellen.

## Geografische Leistungsbewertung

Bewerten Sie die Anwendungsleistung in verschiedenen geografischen Regionen, um die globale Effizienz sicherzustellen.

## Integration der CI/CD-Pipeline

Integrieren Sie Leistungstests in Ihre CI/CD Pipeline für nahtlose und automatisierte Tests während der Entwicklungszyklen.

# Konzepte und Definitionen

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

## Szenario

Testdefinition, einschließlich des Testnamens, der Beschreibung, der Anzahl der Aufgaben, der Parallelität, der AWS-Region, des Hochlaufs, der Wartezeit, des Testtyps, des geplanten Datums und der Wiederholungskonfigurationen.

## 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 Anzahl gleichzeitiger virtueller Benutzer, die pro Aufgabe generiert wurden. Das empfohlene Limit, das auf den Standardeinstellungen basiert, liegt bei 200 virtuellen Benutzern. Die Parallelität

ist durch CPU und Arbeitsspeicher begrenzt. Bei Tests, die auf Apache basieren JMeter, gilt: Je höher die Anzahl der virtuellen Benutzer, desto mehr Speicher wird von der JVM für die ECS-Task genutzt. Die standardmäßige ECS-Aufgabendefinition erstellt Aufgaben mit 4 GB Arbeitsspeicher. Es wird empfohlen, mit niedrigeren Werten für virtuelle Benutzer für eine Aufgabe zu beginnen und die CloudWatch ECS-Metriken für den Task-Cluster zu überwachen. Weitere Informationen finden Sie unter [Amazon ECS-Cluster-Nutzungsmetriken](#).

Hochlauf

Die Zeit bis zum Erreichen der Zielparallelität.

warten bis

Zeit, die Zielparallelität aufrechtzuerhalten.

Eine allgemeine Referenz der AWS-Begriffe 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 auf AWS

### 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. Eine verteilte Loadtester-API, die [Amazon API Gateway](#) nutzt, um die Microservices der Lösung ([AWS Lambda Lambda-Funktionen](#)) aufzurufen.
2. Die Microservices stellen die Geschäftslogik zur Verwaltung von Testdaten und zum Ausführen der Tests bereit.
3. Diese Microservices interagieren mit [Amazon Simple Storage Service](#) (Amazon S3), [Amazon DynamoDB](#) und [AWS Step Functions](#), um Speicherplatz für die Details und Ergebnisse des Testszenarios bereitzustellen und Testszenarien auszuführen.
4. Eine [Amazon Virtual Private Cloud](#) (Amazon VPC) -Netzwerktopologie wird bereitgestellt, die die [Amazon Elastic Container Service \(Amazon ECS\) -Container](#) der Lösung enthält, die auf [AWS Fargate](#) ausgeführt werden.
5. Die Container enthalten das mit der [Open Container Initiative AmazonLinux\(OCI\) konforme Container-Image](#) (bei installiertem Blazemeter Load Testing Framework), das zur Generierung von Last zum Testen der Leistung Ihrer Anwendung verwendet wird. Taurus/Blazemeter ist ein Open-Source-Framework zur Testautomatisierung. Das Container-Image wird von AWS in einem öffentlichen Repository von [Amazon Elastic Container Registry](#) (Amazon ECR) gehostet. Weitere Informationen zum ECR-Image-Repository finden Sie unter Anpassung von [Container-Images](#).

6. Eine von [AWS Amplify](#) betriebene Webkonsole wird in einem Amazon S3 S3-Bucket bereitgestellt, der für statisches Webhosting konfiguriert ist.
7. [Amazon CloudFront](#) bietet sicheren, öffentlichen Zugriff auf die Bucket-Inhalte der Website der Lösung.
8. Bei der Erstkonfiguration erstellt diese Lösung außerdem eine standardmäßige Rolle als Lösungsadministrator (IAM-Rolle) und sendet eine Zugangseinladung an eine vom Kunden angegebene Benutzer-E-Mail-Adresse.
9. Ein [Amazon Cognito Cognito-Benutzerpool](#) verwaltet den Benutzerzugriff auf die Konsole und die Distributed Load Tester API.
10. Nachdem Sie diese Lösung bereitgestellt haben, können Sie die Webkonsole verwenden, um ein Testszenario zu erstellen, das eine Reihe von Aufgaben definiert.
11. Die Microservices verwenden dieses Testszenario, um Amazon ECS auf AWS Fargate-Aufgaben in den angegebenen Regionen auszuführen.
12. [Zusätzlich zum Speichern der Ergebnisse in Amazon S3 und DynamoDB wird die Ausgabe nach Abschluss des Tests in Amazon protokolliert. CloudWatch](#)
13. Wenn Sie die Live-Datenoption auswählen, sendet die Lösung die CloudWatch Amazon-Protokolle für die AWS Fargate-Aufgaben während des Tests für jede Region, in der der Test ausgeführt wurde, an eine Lambda-Funktion.
14. Die Lambda-Funktion veröffentlicht dann die Daten zum entsprechenden Thema in [AWS IoT Core](#) in der Region, in der der Haupt-Stack bereitgestellt wurde. Die Webkonsole abonniert das Thema, und Sie können die Daten sehen, während der Test in der Webkonsole ausgeführt wird.

## Überlegungen zum AWS-Well-Architected-Design

Diese Lösung nutzt die Best Practices des [AWS Well-Architected Framework](#), das Kunden dabei unterstützt, zuverlässige, sichere, effiziente und kostengünstige Workloads in der Cloud zu entwerfen und zu betreiben.

In diesem Abschnitt wird beschrieben, wie die Entwurfsprinzipien und Best Practices des Well-Architected Framework 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.

- Ressourcen, die als Infrastruktur unter Verwendung CloudFormation von Code definiert sind.
- Die Lösung überträgt in verschiedenen Phasen Metriken CloudWatch an Amazon, um die Infrastruktur beobachtbar zu machen: Lambda-Funktionen, Amazon ECS-Aufgaben, Amazon S3-Buckets und die übrigen Lösungskomponenten.

## Sicherheit

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

- Amazon Cognito authentifiziert und autorisiert Benutzer von Web-UI-Apps.
- Die gesamte dienstübergreifende Kommunikation verwendet die entsprechenden [AWS Identity and Access Management](#) (IAM) -Rollen.
- Alle von der Lösung verwendeten Rollen folgen dem Zugriff mit den geringsten Rechten. Sie enthalten nur die Mindestberechtigungen, die zur Ausführung der Aufgabe erforderlich sind.
- Der gesamte Datenspeicher, einschließlich der S3-Buckets, verschlüsselt die Daten im Ruhezustand.
- Ein Amazon Cognito Cognito-Benutzerpool verwaltet den Benutzerzugriff auf die Konsole und die API-Gateway-Endpunkte des Distributed Load Testers.
- Protokollierung, Ablaufverfolgung und Versionierung sind gegebenenfalls aktiviert.
- Der Netzwerkzugriff ist standardmäßig privat, wobei [Amazon Virtual Private Cloud](#) (Amazon VPC) - Endpunkte aktiviert sind, sofern verfügbar.

### Note

Distributed Load Testing auf AWS erstellt mehrere CloudWatch Protokollgruppen, die auf dem verwendeten Service basieren. Die Aufbewahrungsdauer der Protokolle hängt von der Speicherung und den Kosten der Menge der generierten Protokollereignisse ab. Die Lösung erstellt Container-Insight-Logs für die ECS-Aufgaben. Für diese Protokolle ist die Aufbewahrung der Protokolle auf einen Tag konfiguriert. Die für die Services AWS Step Functions, ECS Load Testing Custom Logs und API Gateway Logs erstellten Protokolle sind mit einer Aufbewahrungsfrist von 1 Jahr konfiguriert. AWS Lambda Lambda-Laufzeitprotokollen ist die Protokollspeicherung auf 2 Jahre konfiguriert. Bei den Ausführungsprotokollen von API Gateway ist die Aufbewahrung von Protokollen so

konfiguriert, dass sie nie abläuft. Sie können die Aufbewahrungsfristen für Protokolle je nach Ihren Anforderungen in der CloudWatch Konsole ändern.

## Zuverlässigkeit

In diesem Abschnitt wird beschrieben, wie wir diese Lösung unter Verwendung der Prinzipien und bewährten Verfahren der [Zuverlässigkeitskomponente 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), um eine hohe Verfügbarkeit und Wiederherstellung nach einem Serviceausfall sicherzustellen.
- 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.
- AWS ECS auf AWS Fargate ermöglicht es Ihnen, nur für die Rechenressourcen zu zahlen, die Sie nutzen, ohne Vorabkosten.

## 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 zwei Komponenten auf hoher Ebene: einem [Frontend](#) und einem [Backend](#).

## Frontend

Das Frontend besteht aus einer Belastungstest-API und einer Webkonsole, über die Sie mit dem Backend der Lösung interagieren.

## Belastungstest-API

Distributed Load Testing auf AWS konfiguriert Amazon API Gateway so, dass es die RESTful API der Lösung hostet. Benutzer können über die mitgelieferte Webkonsole und RESTful API sicher mit Testdaten interagieren. Die API fungiert als „Eingangstür“ für den Zugriff auf Testdaten, die in Amazon DynamoDB gespeichert sind. Sie können die auch verwenden APIs , 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, mit dem die Konsole Anfragen an die Lösung APIs (Amazon API Gateway Gateway-Endpunkte) senden kann. HTTPS-Anfragen werden von der Konsole APIs mit dem Autorisierungsheader, der das Token enthält, an die Konsole 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](#) in diesem Handbuch.

## 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 in Amazon S3 gehostet wird und auf die über Amazon zugegriffen wird. CloudFront 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 Webkonsole soll demonstrieren, wie Sie mit dieser Lasttestlösung interagieren können. In einer Produktionsumgebung empfehlen wir, die Webkonsole an Ihre spezifischen Bedürfnisse anzupassen oder eine eigene Konsole zu erstellen.

Die URL der Webkonsole ist der Name der CloudFront Distributionsdomäne, der in den CloudFormation Ausgaben als Konsole 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.

## Backend

Das Backend besteht aus einer Container-Image-Pipeline und einer Lasttest-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 [Benutzerdefinierte Cost Allocation Tags](#) im AWS Billing and Cost Management-Benutzerhandbuch.

## Pipeline für Container-Images

Diese Lösung nutzt ein Container-Image, das [AmazonLinux](#) als Basis-Image mit installiertem Blazemeter-Load-Testing-Framework erstellt wurde. Dieses Bild wird in einem öffentlichen Repository von Amazon Elastic Container Registry (Amazon ECR) gehostet. Das Image wird verwendet, um Aufgaben im Amazon ECS on AWS Fargate-Cluster auszuführen.

Weitere Informationen finden Sie im Abschnitt zur [Anpassung von Container-Images](#) in diesem Handbuch.

## Infrastruktur testen

Zusätzlich zur Hauptvorlage erstellt die Lösung eine sekundäre Vorlage, um die erforderlichen Ressourcen für die Durchführung von Tests in mehreren Regionen bereitzustellen. Die Vorlage wird in Amazon S3 gespeichert, und ein Link zur Vorlage wird in der Webkonsole bereitgestellt. Die sekundären Vorlagen erstellen eine VPC, einen AWS Fargate-Cluster und eine Lambda-Funktion für die Verarbeitung von Live-Daten.

Weitere Informationen zum Starten einer sekundären Region finden Sie im Abschnitt [Bereitstellung in mehreren Regionen dieses Handbuchs](#).

## Test-Engine

Die Distributed Load Testing-Lösung verwendet Amazon Elastic Container Service (Amazon ECS) und AWS Fargate, um Tausende von verbundenen Benutzern in mehreren Regionen zu simulieren und eine ausgewählte Anzahl von Transaktionen pro Sekunde zu generieren.

Sie definieren die Parameter für die Aufgaben, die im Rahmen des Tests ausgeführt werden, mithilfe der mitgelieferten Webkonsole. Die Lösung verwendet diese Parameter, um ein JSON-Testscenario zu generieren und es in Amazon S3 zu speichern.

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 mit ecr-checker, eine AWS-Lambda-Funktion, eine task-status-checker AWS-Lambda-Funktion für Task-Runner, eine AWS-Lambda-Funktion zum Abbrechen von Aufgaben und eine AWS-Lambda-Funktion zum Analysieren von Ergebnissen. [Weitere Informationen zum Workflow finden Sie im Abschnitt „Workflow testen“ dieses Handbuchs](#). Weitere Informationen zu Testergebnissen finden Sie im Abschnitt [Testergebnisse](#) dieses Handbuchs. Weitere Informationen zum Ablauf der Teststornierung finden Sie im Abschnitt zum [Ablauf der Teststornierung](#) in diesem Handbuch.

Wenn Sie Live-Daten auswählen, initiiert die Lösung in jeder Region eine real-time-data-publisher Lambda-Funktion 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 [Live-Daten](#) dieses Handbuchs.

# AWS-Services in dieser Lösung

Die folgenden AWS-Services sind in dieser Lösung enthalten:

AWS Service	Beschreibung
<a href="#">Amazon API Gateway</a>	Kern. Hostet REST-API-Endpunkte in der Lösung.
<a href="#">AWS CloudFormation</a>	Kern. Verwaltet Bereitstellungen für die Lösungsinfrastruktur.
<a href="#">Amazon CloudFront</a>	Kern. Stellt die in Amazon S3 gehosteten Webinhalte bereit.
<a href="#">Amazon CloudWatch</a>	Kern. Speichert die Lösungsprotokolle und Metriken.
<a href="#">Amazon Cognito</a>	Kern. Verwaltet die Benutzerverwaltung und Authentifizierung für die API.
<a href="#">Amazon-DynamoDB</a>	Kern. Speichert Bereitstellungsinformationen und testet Szenariodetails und Ergebnisse.
<a href="#">Amazon Elastic Container Service</a>	Kern. Stellt unabhängige Amazon ECS-Aufgaben auf AWS Fargate-Containern bereit und verwaltet sie.
<a href="#">AWS Fargate</a>	Kern. Hostet die Amazon ECS-Container der Lösung
<a href="#">AWS Identity and Access Management</a>	Kern. Kümmt sich um die Verwaltung von Benutzerrollen und Berechtigungen.
<a href="#">AWS Lambda</a>	Kern. Stellt Logik für die APIs Implementierung, das Analysieren von Testergebnissen und das Starten von workers/leader Aufgaben bereit.
<a href="#">AWS Step Functions</a>	Kern. Orchestriert die Bereitstellung von Amazon ECS-Containern für AWS Fargate-Aufgaben in den angegebenen Regionen
<a href="#">AWS Amplify</a>	Unterstützend. Stellt eine von <a href="#">AWS Amplify</a> betriebene Webkonsole bereit.
<a href="#">CloudWatch Amazon-Veranstaltungen</a>	Unterstützend. Plant Tests so, dass sie automatisch an einem bestimmten Datum oder an wiederkehrenden Terminen beginnen.

AWS Service	Beschreibung
<a href="#">Amazon Elastic Container Registry</a>	Unterstützend. Hostet das Container-Image in einem öffentlichen ECR-Repository.
<a href="#">AWS IoT Core</a>	Unterstützend. Ermöglicht die Anzeige von Live-Daten für einen laufenden Test, indem Sie das entsprechende Thema in AWS IoT Core abonnieren.
<a href="#">AWS Systems Manager</a>	Unterstützend. Bietet Ressourcenüberwachung und Visualisierung von Ressourcenoperationen und Kostendaten auf Anwendungsebene.
<a href="#">Amazon S3</a>	Unterstützend. Hostet die statischen Webinhalte, Protokolle, Metriken und Testdaten.
<a href="#">Amazon Virtual Private Cloud</a>	Unterstützend. Enthält die Amazon ECS-Container der Lösung, die auf AWS Fargate ausgeführt werden.

## 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 S3s3://<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 eine CloudWatch Ereignisregel, die AWS Step Functions am angegebenen Datum auslöst. Weitere Informationen zum Planungs-Workflow finden Sie im Abschnitt [Test-Scheduling-Workflow](#) in diesem Handbuch.
4. Konfigurationsdetails werden in der Amazon DynamoDB-Szenario-Tabelle gespeichert.

5. Im Task Runner-Workflow von AWS Step Functions prüft die task-status-checker AWS Lambda Lambda-Funktion, 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 task-status-checker AWS Lambda Lambda-Funktion prüft, ob alle Amazon ECS-Worker-Aufgaben mit derselben Test-ID ausgeführt werden. Wenn Aufgaben noch bereitgestellt werden, wartet sie eine Minute und prüft erneut. Sobald alle Amazon ECS-Aufgaben ausgeführt werden, gibt es die Test-ID, die Anzahl der Aufgaben, den Testtyp, alle Aufgaben und das Präfix zurück IDs und übergibt sie an die Task-Runner-Funktion.
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 task-status-checker AWS Lambda Lambda-Funktion 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://dlte-bucket/test-scenarios/ <$TEST_ID> /results/ <$UUID> .json`.
16. Wenn die XML-Dateien auf Amazon S3 hochgeladen werden, liest die AWS Lambda Lambda-Funktion für 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.

## Designüberlegungen

### Unterstützte Anwendungen

Diese Lösung unterstützt cloudbasierte Anwendungen und lokale Anwendungen, sofern Sie über eine Netzwerkverbindung von Ihrem AWS-Konto zu Ihrer Anwendung verfügen. Die Lösung unterstützt APIs die Verwendung von HTTP oder HTTPS. Sie haben auch die Kontrolle über die HTTP-Anforderungsheader, sodass Sie Autorisierungs- oder benutzerdefinierte Header hinzufügen können, um Token oder API-Schlüssel zu übergeben.

### JMeter Unterstützung von Skripten

Wenn Sie ein Testszenario mithilfe der Benutzeroberfläche (UI) dieser Lösung erstellen, können Sie ein JMeter Testskript verwenden. Nachdem Sie die JMeter Skriptdatei ausgewählt haben, wird sie in den `<stack-name>Amazon Simple Storage Service (Amazon S3) -Bucket -scenariosbucket` hochgeladen. Wenn Amazon Elastic Container Service (Amazon ECS) -Aufgaben ausgeführt werden, wird das JMeter Skript aus dem `<stack-name>Amazon S3 S3-Bucket -scenariosbucket` heruntergeladen und der Test wird ausgeführt.

Wenn Sie JMeter Eingabedateien haben, können Sie die Eingabedateien zusammen mit dem Skript komprimieren. JMeter 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 Ihrer JMeter Skriptdatei JMeter Eingabedateien hinzufügen, müssen Sie den relativen Pfad der Eingabedateien in Ihrer Skriptdatei angeben. JMeter Außerdem müssen sich die Eingabedateien im relativen Pfad befinden. Wenn sich Ihre JMeter Eingabedateien und die Skriptdatei beispielsweise stattdessen im Verzeichnis/home/user directory and you refer to the input files in the JMeter script file, the path of input files must be `./INPUT_FILES`. If you use `/home/user/INPUT_FILES` befinden, schlägt der Test fehl, da die Eingabedateien nicht gefunden werden können.

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 Skripten finden Sie im Benutzerhandbuch. JMeter JMeter](#)

## Unterstützung für K6-Skripte

Die Lösung unterstützt Tests auf Basis des K6-Frameworks. [K6 wird mit einer AGPL-3.0-Lizenz veröffentlicht](#). Die Lösung zeigt eine Lizenzbestätigungsmeldung an, wenn ein neuer Test für K6 erstellt wird. Die K6-Testdatei kann zusammen mit allen erforderlichen Eingabedateien in eine Archivdatei aufgenommen und mit der Upload-Option für das Testszenario hochgeladen werden.

## Unterstützung für Locust-Skripte

Die Lösung unterstützt Tests auf Basis des Locust Frameworks. Die Locust-Testdatei kann zusammen mit allen erforderlichen Eingabedateien in eine Archivdatei aufgenommen und mit der Upload-Option für das Testszenario hochgeladen werden.

## Planung von Tests

Sie können Tests so planen, dass sie zu einem future Zeitpunkt ausgeführt werden, oder die Option Jetzt ausführen verwenden. Sie können einen Test als einmaligen Testlauf in der future planen oder einen wiederkehrenden Test einrichten, bei dem Sie ein Datum für die erste Ausführung und eine geplante Wiederholung angeben. Zu den Optionen für die Wiederholung gehören: täglich, wöchentlich, zweiwöchentlich und monatlich. Weitere Informationen zur Funktionsweise der Terminplanung finden Sie im Abschnitt zum [Testen des Planungsablaufs](#) in diesem Handbuch.

Ab Version 3.3.0 können Benutzer mit Distributed Load Testing auf AWS Lasttests mithilfe von Cron-Ausdrücken planen. Wählen Sie Nach Zeitplan ausführen und dann die Registerkarte CRON, um entweder manuell einen Cron-Wert einzugeben oder die Dropdown-Felder zu verwenden. Das cronExpiryDate muss mit dem geplanten Testlaufdatum übereinstimmen. Überprüfen Sie die Termine für den nächsten Testlauf (UTC), um Ihren Zeitplan zu bestätigen.

### 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

Diese Lösung beinhaltet ein CloudWatch Amazon-Dashboard für jeden Test und zeigt die kombinierte Ausgabe aller Aufgaben, die für diesen Test im Amazon ECS-Cluster ausgeführt werden, in Echtzeit an. Das CloudWatch Dashboard zeigt die durchschnittliche Antwortzeit, die Anzahl der gleichzeitigen Benutzer, die Anzahl der erfolgreichen Anfragen und die Anzahl der fehlgeschlagenen Anfragen an. Jede Metrik wird sekundengenau aggregiert und das Dashboard wird jede Minute aktualisiert.

## 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 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](#).

## 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-Servicealiste](#).

# Planen Sie Ihren Einsatz

In diesem Abschnitt werden die [Kosten](#), die [Sicherheit](#), die [Regionen](#) und andere Überlegungen vor der Bereitstellung der Lösung beschrieben.

## Kosten

Sie sind für die Kosten der AWS-Services verantwortlich, die Sie beim Betrieb dieser Lösung in Anspruch nehmen. Die Gesamtkosten für den Betrieb dieser Lösung hängen von der Anzahl der ausgeführten Lasttests, der Dauer dieser Lasttests und der Menge der im Rahmen der Tests verwendeten Daten ab. Zum jetzigen Zeitpunkt belaufen sich die 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 V CPUs - und 4 GB-Speicher), 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](#).

#### Important

Ab Version 1.3.0 wird die CPU auf 2 vCPU und der Arbeitsspeicher auf 4 GB erhöht. Diese Änderungen erhöhen die geschätzten Kosten im Vergleich zu früheren Versionen dieser Lösung. Wenn Ihre Auslastungstests diese Erhöhungen Ihrer AWS-Ressourcen nicht erfordern, können Sie sie reduzieren. Weitere Informationen finden Sie im Abschnitt [Erhöhung 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.

Die Preise sind freibleibend. Vollständige Informationen finden Sie auf der Preisseite für jeden AWS-Service, den Sie in dieser Lösung verwenden werden.

## Sicherheit

Wenn Sie Systeme auf der AWS-Infrastruktur aufbauen, werden Sie und AWS gemeinsam für die Sicherheit verantwortlich sein. 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-Funktionen der Lösung Zugriff gewähren, um regionale Ressourcen zu erstellen.

## Amazon CloudFront

Diese Lösung stellt eine Web-UI 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 verwendet die CloudFront Distribution TLS 1.2, um die höchste Sicherheitsstufe durchzusetzen. 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?](#)

## 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 ein bestimmtes Classless Inter-Domain Routing (CIDR).

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.

## Netzwerk-Stresstest

Sie sind dafür verantwortlich, diese Lösung gemäß der [Netzwerkstresstest-Richtlinie](#) zu verwenden. Diese Richtlinie gilt beispielsweise für Situationen, in denen Sie planen, Netzwerktests mit hohem Volumen direkt von Ihren EC2 Amazon-Instances zu anderen Standorten wie anderen EC2 Amazon-Instances, AWS-Eigenschaften/-Services oder externen Endpunkten durchzuführen. Diese Tests werden manchmal als Stresstests, Belastungstests oder Gameday-Tests bezeichnet. Die meisten Kundentests fallen nicht unter diese Richtlinie. Beziehen Sie sich jedoch auf diese Richtlinie, wenn Sie glauben, dass Sie Traffic generieren, der insgesamt länger als 1 Minute über 1 Gbit/s (1 Milliarde Bit pro Sekunde) oder über 1 Gpps (1 Milliarde Pakete pro Sekunde) anhält.

## Beschränkung des Zugriffs auf die öffentliche Benutzeroberfläche

Verwenden Sie die Sicherheitsautomatisierungslösung [AWS WAF \(Web Application Firewall\)](#), um den Zugriff auf die öffentlich zugängliche Benutzeroberfläche über die von IAM und Amazon Cognito bereitgestellten Authentifizierungs- und Autorisierungsmechanismen hinaus einzuschränken.

Diese Lösung stellt automatisch eine Reihe von AWS-WAF-Regeln bereit, die häufig vorkommende webbasierte Angriffe filtern. Benutzer können aus vorkonfigurierten Schutzfunktionen wählen, die die Regeln definieren, die in einer AWS WAF Web Access Control List (Web ACL) enthalten sind.

## Unterstützte AWS Regionen

Diese Lösung verwendet den Amazon Cognito-Service, der derzeit nicht in allen AWS-Regionen verfügbar ist. Die aktuelle Verfügbarkeit von AWS-Services nach Regionen finden Sie in der [regionalen AWS-Serviceliste](#).

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

Name der Region	
USA Ost (Ohio)	Asien-Pazifik (Tokio)
USA Ost (Nord-Virginia)	Kanada (Zentral)
USA West (Nordkalifornien)	Europa (Frankfurt)
USA West (Oregon)	Europa (Irland)

Name der Region	
Asien-Pazifik (Mumbai)	Europa (London)
Asien-Pazifik (Seoul)	Europa (Paris)
Asien-Pazifik (Singapur)	Europa (Stockholm)
Asien-Pazifik (Sydney)	Südamerika (São Paulo)

## 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.

Das AmazonLinux Container-Image für das Image (mit installiertem Blazemeter) begrenzt die Anzahl gleichzeitiger Verbindungen pro Aufgabe nicht, was jedoch nicht bedeutet, dass es 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 [Ermitteln der Benutzeranzahl](#) in diesem Handbuch.

#### Note

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

## Gleichzeitige Tests

Diese Lösung beinhaltet ein CloudWatch Amazon-Dashboard für jeden Test und zeigt die kombinierte Ausgabe aller Aufgaben, die für diesen Test im Amazon ECS-Cluster ausgeführt werden, in Echtzeit an. Das CloudWatch Dashboard zeigt die durchschnittliche Antwortzeit, die Anzahl der gleichzeitigen Benutzer, die Anzahl der erfolgreichen Anfragen und die Anzahl der fehlgeschlagenen Anfragen an. Jede Metrik wird sekundengenau aggregiert und das Dashboard wird jede Minute aktualisiert.

## EC2 Testrichtlinien von Amazon

Sie benötigen keine Genehmigung von AWS, um Lasttests mit dieser Lösung durchzuführen, solange Ihr Netzwerkverkehr unter 1 Gbit/s bleibt. Wenn Ihr Test mehr als 1 Gbit/s generiert, wenden Sie sich an AWS. Weitere Informationen finden Sie in den [EC2 Testrichtlinien von Amazon](#).

## 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

### CloudWatch Alarme aktivieren oder einrichten

Es wird empfohlen, die Ressourcen der Lösung nach der Bereitstellung kontinuierlich zu überwachen. Sie können erwägen, [CloudWatch Metriken](#) für die von der Lösung erstellten Ressourcen einzurichten.

### CloudFront Amazon-Vertriebsmetriken

Überprüfen Sie die [CloudFront Vertriebsmetriken](#) und richten Sie Alarme ein, um Benachrichtigungen zu überwachen und zu empfangen.

### Amazon API Gateway Gateway-Metriken

Überprüfen Sie die [Dimensionen und Metriken von Amazon API Gateway](#) und richten Sie Alarme ein, um Benachrichtigungen zu überwachen und zu empfangen.

# Stellen Sie die Lösung bereit

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.

## Überblick über den Bereitstellungsprozess

Folgen Sie den step-by-step Anweisungen in diesem Abschnitt, um die Lösung zu konfigurieren und in Ihrem Konto bereitzustellen.

Bevor Sie die Lösung auf den Markt bringen, sollten Sie sich mit den [Kosten](#), der [Architektur](#), der [Netzwerksicherheit](#) und anderen Überlegungen befassen, die weiter oben in diesem Handbuch erörtert wurden.

Zeit für die Bereitstellung: Ungefähr 15 Minuten

## CloudFormation AWS-Vorlage

Sie können die CloudFormation Vorlage 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 folgende CloudFormation AWS-Vorlage, die Sie vor der Bereitstellung herunterladen können:

[load-testing-on-aws.template](#) — Verwenden Sie diese Vorlage, um die Lösung und alle zugehörigen Komponenten zu starten. 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.

### 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 des -Stacks

### Important

Wenn Sie den Stack von einer Version vor v3.2.6 auf die neueste Version aktualisieren, lesen Sie [diesen Abschnitt](#), bevor Sie den Stack aktualisieren.

Bevor Sie mit der automatisierten Bereitstellung beginnen, sollten Sie sich mit der Architektur und anderen Überlegungen, die in diesem Handbuch besprochen werden, vertraut machen. Folgen Sie den step-by-step Anweisungen in diesem Abschnitt, um Distributed Load Testing auf AWS zu konfigurieren und für Ihr Konto bereitzustellen.

Zeit für die Bereitstellung: Ungefähr 15 Minuten

### Important

Diese Lösung beinhaltet eine Option zum Senden anonymisierter Betriebsmetriken an 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](#).

Um diese Funktion zu deaktivieren, laden Sie die Vorlage herunter, ändern Sie den Abschnitt CloudFormation AWS-Zuordnung und verwenden Sie dann die CloudFormation AWS-Konsole, um Ihre aktualisierte Vorlage hochzuladen und die Lösung bereitzustellen. Weitere Informationen finden Sie im Abschnitt [Anonymisierte Datenerfassung](#) dieses Handbuchs.

Diese automatisierte CloudFormation AWS-Vorlage stellt Distributed Load Testing auf AWS bereit.

### Note

Sie sind für die Kosten der AWS-Services verantwortlich, die beim Betrieb dieser Lösung verwendet werden. Weitere Informationen finden Sie im Abschnitt [Kosten](#) in diesem Handbuch und auf der Preisseite für jeden AWS-Service, der in dieser Lösung verwendet wird.

1. Melden Sie sich bei der AWS-Managementkonsole an und klicken Sie auf die Schaltfläche unten, um die CloudFormation AWS-Vorlage distributed-load-testing-on -aws zu starten.

Alternativ können Sie [die Vorlage auch 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-Service-Liste](#).

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 Lösungstapel 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	Beschreibung
Name des Administrators	<Erfordert Eingabe>	Benutzername für den ursprünglichen Lösungsadministrator.
E-Mail-Adresse des Administrators	<i>&lt;Requires input&gt;</i>	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	Beschreibung
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	Beschreibung
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.
Container-Image automatisch aktualisieren	Yes	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.

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.

## Bereitstellung in mehreren Regionen

Bereitstellungszeit: ungefähr fünf Minuten

Sie können Tests in mehreren Regionen ausführen. Wenn Sie die Distributed Load Testing-Lösung bereitstellen, erstellt sie drei Amazon S3 S3-Buckets. Die Lösung erstellt einen sekundären regionalen Stack und speichert ihn im Amazon S3 S3-Szenario-Bucket.

**Note**

Die Benennungskonvention für Buckets lautet `<stack-name> -dltestrunnerstoragedltszenariosbucket <_[0-9][0-9]..-<[0-9][0-9].._` mit dem Schlüsselwort `scenarios` im Bucket-Namen, den Sie finden können, indem Sie zur S3-Konsole und dann zu Buckets navigieren.

Um eine Bereitstellung in mehreren Regionen auszuführen, müssen Sie die regionale CloudFormation Vorlage, die im Amazon S3 S3-Szenario-Bucket gespeichert ist, in den Regionen bereitstellen, in denen Sie den Test ausführen möchten. Sie können die regionale Vorlage wie folgt installieren:

1. Navigieren Sie in der Webkonsole der Lösung im oberen Menü zu Regionen verwalten.
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 Lösungstapel 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	Beschreibung
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.

Parameter	Standard	Beschreibung
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-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.

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. Stellen Sie sicher, dass Sie das Kästchen ankreuzen, um zu bestätigen, 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, wird die neue Region im Modal „Regionen verwalten“ aufgeführt. Sie können diese Region in einem Test verwenden, indem Sie sie bei der Testerstellung auswählen. Die Lösung erstellt für jede Region, die in der Szenariotabelle gestartet wird, ein DynamoDB-Element, das die erforderlichen Informationen zu den Testressourcen in dieser Region enthält. Sie können die Testergebnisse in der Webkonsole nach Regionen sortieren. Aufgrund von API-Einschränkungen können Sie die Gesamtergebnisse aller Regionen in einem Test mit mehreren Regionen nur anzeigen, indem Sie sie in CloudWatch Amazon-Metriken grafisch darstellen. Den Quellcode für das Diagramm finden Sie in den Testergebnissen, sobald der Test abgeschlossen ist.

**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.

# Aktualisieren Sie die Lösung

Wenn Sie die Lösung bereits bereitgestellt haben, gehen Sie wie folgt vor, um den CloudFormation Stack der Lösung zu aktualisieren und die neueste Version des Lösungsframeworks zu erhalten.

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.

## Bei der Aktualisierung von DLT-Versionen, die älter als v3.2.6 sind, auf v3.3.x und v3.3.x auf die neueste Version schlägt die Aktualisierung des Stacks fehl

1. Laden Sie die Vorlage [distributed-load-testing-on-aws.template](#) herunter.
2. Öffnen Sie die Vorlage und navigieren Sie zu Bedingungen: und suchen Sie nach DLTCommonResourcesAppRegistryCondition
3. Die Ausgabe sollte wie folgt aussehen:

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

4. Ändern Sie den zweiten Wert „wahr“ in „Falsch“:

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

5. Verwenden Sie die benutzerdefinierte Vorlage, um Ihren Stack zu aktualisieren.
6. Dieser Stack entfernt Ressourcen, die sich auf die App-Registrierung beziehen, aus dem Stack. Daher sollte das Update abgeschlossen sein.
7. Führen Sie ein weiteres Stack-Update mit der neuesten Vorlagen-URL durch, um Ihrem Stack wieder Anwendungsressourcen für die App-Registrierung hinzuzufügen.

### Note

AWS Systems Manager Application Manager bietet Ihnen einen Überblick über diese Lösung und ihre Ressourcen auf Anwendungsebene, sodass Sie:

1. Überwachen Sie die Ressourcen, die Kosten für die bereitgestellten Ressourcen in allen Stacks und AWS-Konten sowie die mit dieser Lösung verknüpften Protokolle von einem zentralen Standort aus.

2. Zeigen Sie Betriebsdaten für die Ressourcen dieser Lösung im Kontext einer Anwendung an, z. B. den Bereitstellungsstatus, CloudWatch Alarme, Ressourcenkonfigurationen und Betriebsprobleme.

# 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 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 der Tests dauert zu lange oder sie laufen auf unbestimmte Zeit fest

- 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/ - herunterlädt. distributed-load-testing-on aws-load-tester](#) 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

Beachten Sie beim Testen von privaten API-Endpunkten, auf die nicht über das Internet-Gateway zugegriffen werden kann, die folgenden Ansätze:

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, in der die Lösung bereitgestellt wird, und der VPC mit dem privaten

Endpunkt ein. Konfigurieren Sie in beiden Fällen die entsprechenden Routing-Tabellen. VPCs  
Weitere Informationen finden Sie in der [VPC-Peering-Dokumentation](#).

5. Transit Gateway: Für komplexere Netzwerkszenarien mit mehreren VPCs Komponenten 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 Balancer 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 sind 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 je nach Komplexität des Testskripts, Parallelität und 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.

## Kontaktieren Sie AWS Support.

Wenn Sie über [AWS Developer Support](#), [AWS Business Support](#) oder [AWS Enterprise Support](#) verfügen, können Sie das Support Center nutzen, um kompetente Unterstützung 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 das Problem im Feld Beschreibung detailliert.
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 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.

## Deinstalliere die Lösung

Sie können die Lösung Distributed Load Testing on AWS über die AWS-Managementkonsole oder über die AWS-Befehlszeilenschnittstelle deinstallieren. Sie müssen die mit dieser Lösung erstellten Konsolen-, Szenario- und Protokollierungs-Buckets von Amazon Simple Storage Service (Amazon S3) manuell löschen. AWS-Lösungsimplementierungen löschen sie nicht automatisch, falls Sie Daten aufbewahren müssen.

### Note

Wenn Sie regionale Stacks bereitgestellt haben, müssen Sie die Stacks in diesen Regionen löschen, bevor Sie den Haupt-Stack löschen.

## Verwendung der AWS-Managementkonsole

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

## 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>
```

## Löschen der Amazon S3 S3-Buckets

Diese Lösung ist so konfiguriert, dass die von der Lösung erstellten Amazon S3 S3-Buckets (für die Bereitstellung in einer Opt-in-Region) beibehalten werden, falls Sie sich entscheiden, den CloudFormation AWS-Stack zu löschen, um versehentlichen Datenverlust zu verhindern. Nach der

Deinstallation der Lösung können Sie diesen S3-Bucket manuell löschen, wenn Sie die Daten nicht behalten müssen. Gehen Sie wie folgt vor, um den Amazon S3 S3-Bucket zu löschen.

1. Melden Sie sich bei der [Amazon S3-Konsole](#) an.
2. Wählen Sie im linken Navigationsbereich Buckets aus.
3. Geben Sie im Feld Buckets nach Namen suchen den Namen des Stacks dieser Lösung ein.
4. Wählen Sie einen der S3-Buckets der Lösung aus und wählen Sie Leer aus.
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 aus.

Wiederholen Sie die Schritte 4 bis 7, bis Sie alle S3-Buckets gelöscht haben.

Führen Sie den folgenden Befehl aus, um den S3-Bucket mithilfe der AWS-CLI zu löschen:

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

# Benutze die Lösung

Dieser Abschnitt enthält Informationen zur Verwendung der Lösung Distributed Load Testing on AWS, einschließlich [Testergebnisse](#), [Arbeitsablauf zur Testplanung](#) und [Live-Daten](#).

## Testergebnisse

Distributed Load Testing auf AWS nutzt das Load Testing-Framework, um Anwendungstests in großem Umfang durchzuführen. Wenn ein Test abgeschlossen ist, wird ein detaillierter Bericht mit den folgenden Ergebnissen generiert.

- 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 benötigt wird, um für alle vom Test generierten Anfragen eine Verbindung zum Host herzustellen.
- Durchschnittliche Bandbreite — Die durchschnittliche Bandbreite für alle vom 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.
- Perzentil — Das Perzentil der Antwortzeit für den Test. Die maximale Reaktionszeit beträgt 100%; die minimale Reaktionszeit beträgt 0%.

### Note

Die Testergebnisse werden in der Konsole angezeigt. Sie können die XML-Dateien für die Roh-testergebnisse im Results Ordner des Scenarios Amazon S3 S3-Buckets einsehen.

Weitere Informationen zu Taurus-Testergebnissen finden Sie unter [Generieren von Testberichten](#) im Taurus-Benutzerhandbuch.

## Arbeitsablauf bei der Testplanung

Verwenden Sie die Webkonsole, um einen Auslastungstest zu planen. Bei der Planung eines Tests wird der folgende Workflow ausgeführt:

- Wenn ein Belastungstest mit der Option zum Planen erstellt wird, werden die Zeitplanparameter über Amazon API Gateway an die API der Lösung gesendet.
- Die API übergibt die Parameter dann an eine Lambda-Funktion, die eine CloudWatch Ereignisregel erstellt, deren Ausführung an dem angegebenen Datum geplant wird.
- Wenn es sich bei dem Test um einen einmaligen Test handelt, wird die CloudWatch Events-Regel am angegebenen Datum ausgeführt. Die `api-services` Lambda-Funktion führt einen neuen Test durch den Test-Workflow durch.
- Wenn es sich bei dem Test um einen wiederkehrenden Test handelt, wird die CloudWatch Ereignisregel am angegebenen Datum aktiviert. Die `api-services` Lambda-Funktion wird ausgeführt, wodurch die aktuelle CloudWatch Ereignisregel gelöscht und eine weitere Regel erstellt wird, die sofort ausgeführt wird, wenn sie erstellt wird, und danach wiederholt, basierend auf der angegebenen Wiederholungshäufigkeit.

## 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). Anhand des folgenden Beispiels können Sie damit beginnen, die Grenzwerte für gleichzeitige Benutzer für Ihren Test zu ermitteln:

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 linken 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 die Container-Ressourcen erhöht werden, um eine höhere Anzahl gleichzeitiger Benutzer zu ermöglichen. Dies führt jedoch zu höheren Kosten. Einzelheiten finden Sie im Abschnitt [Erhöhung der Container-Ressourcen](#) in diesem Handbuch.

#### 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

## Live-Daten

Sie können bei der Ausführung eines Tests optional Live-Daten einbeziehen, um in Echtzeit Einblicke in das Geschehen zu erhalten. Die CloudWatch Protokollgruppe für die Fargate-Aufgaben enthält einen Abonnementfilter für Ergebnisse von Tests, die die Live-Datenoption enthalten. Wenn die Lösung das Muster findet, initiiert sie eine Lambda-Funktion, die die Daten strukturiert und in einem AWS IoT Core Core-Thema veröffentlicht. Die Webkonsole abonniert das Thema, empfängt die eingehenden Daten und stellt die aggregierten Daten in Intervallen von einer Sekunde grafisch dar. Die Webkonsole enthält vier Diagramme: durchschnittliche Antwortzeit, virtuelle Benutzer, Erfolge und Misserfolge.

**Note**

Die Daten sind kurzlebig und dienen nur dazu, zu sehen, was passiert, während der Test läuft. Sobald ein Test abgeschlossen ist, speichert die Lösung die Ergebnisdaten in DynamoDB und Amazon S3. 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.

## Testen Sie den Stornierungs-Workflow

Wenn Sie einen Auslastungstest über die Webkonsole abbrechen, führt die Lösung den folgenden Testabbruch-Workflow aus.

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

# Entwicklerhandbuch

Dieser Abschnitt enthält den Quellcode für die Lösung und zusätzliche 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 zu jeder Lösungsversion passen, wenn automatische Updates nicht ausgewählt sind. Das AWS-Lösungsteam verwendet 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 das Problem zu beheben CVEs, ohne die Kompatibilität mit der veröffentlichten Lösungsversion zu beeinträchtigen. Wenn Bilder gepatcht werden und sie sich auf derselben Nebenversion befinden, wird das Stable-Tag automatisch aktualisiert, und es wird ein zusätzliches Image-Tag in dem Format erstellt. `<solution-version>_<date-of-fix>` Wenn eine Haupt- oder Nebenversion veröffentlicht wird, ist ein Full-Stack-Update erforderlich, um die neueste Image-Version zu erhalten, da das Stable-Tag inkrementiert wird, sodass seine Version mit der Version der Lösung übereinstimmt. Wenn Sie sich für automatische Updates entscheiden, werden die Änderungen am Image, einschließlich der Fehlerbehebungen CVEs und kleinerer Bugfixes, automatisch auf das Image angewendet, bis die neueste passende Nebenversion verfügbar ist.

## Versionen

Kunden, die die neueste Lösungsversion verwenden, erhalten automatisch Sicherheitspatches und kleinere, unveränderliche Bugfixes, wenn sie sich für automatische Image-Updates entscheiden. Das Image ruft das neueste Image automatisch auf die neueste passende Nebenversion auf. Um den Container an eine bestimmte Version zu binden, kann die Aufgabendefinition so bearbeitet werden, dass der Container eine bestimmte Image-Version verwendet, indem die markierte Version des Images verwendet wird. Automatische Updates können auch deaktiviert werden, indem Sie beim Starten des Stacks Nein für automatische Updates auswählen. CloudFormation Dadurch wird die Image-Version gestartet, die der Lösungsversion entspricht.

## Anpassung des Container-Images

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. Wenn Sie das Container-Image anpassen möchten, können Sie das Image neu erstellen und in ein ECR-Image-Repository in Ihrem eigenen AWS-Konto übertragen.

Wenn Sie diese Lösung anpassen möchten, können Sie das Standard-Container-Image verwenden oder diesen Container nach Ihren Bedürfnissen bearbeiten. Wenn Sie die Lösung anpassen, verwenden Sie das folgende Codebeispiel, um die Umgebungsvariablen zu deklarieren, bevor Sie Ihre benutzerdefinierte Lösung 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
```

Wenn Sie das Container-Image anpassen möchten, können Sie es entweder in einem privaten Image-Repository oder in einem öffentlichen Image-Repository in Ihrem AWS-Konto hosten. Die Image-Ressourcen befinden sich im `deployment/ecr/distributed-load-testing-on-aws-load-tester` Verzeichnis, das sich in der Codebasis befindet.

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

- Informationen zu privaten Amazon ECR-Repositories und -Images finden Sie unter [Private Amazon ECR-Repositories](#) und [private Images](#) im Amazon ECR-Benutzerhandbuch.
- Informationen zu öffentlichen Amazon ECR-Repositories und -Images finden Sie unter [Öffentliche Amazon ECR-Repositories](#) und [öffentliche Images](#) im Amazon ECR Public User Guide.

Sobald Sie Ihr eigenes Image erstellt haben, können Sie die folgenden Umgebungsvariablen deklarieren, bevor Sie Ihre maßgeschneiderte 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 folgende Beispiel zeigt die Containerdatei.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:2023-minimal

RUN dnf update -y && \
    dnf install -y python3.11 python3.11-pip java-21-amazon-corretto bc procs jq
    findutils unzip && \
    dnf clean all

ENV PIP_INSTALL="pip3.11 install --no-cache-dir"

# install bzt
RUN $PIP_INSTALL --upgrade bzt awscli setuptools==78.1.1 h11 urllib3==2.2.2 && \
    $PIP_INSTALL --upgrade bzt
COPY ./bzt-rc /root/.bzt-rc
RUN chmod 755 /root/.bzt-rc

# install bzt tools
RUN bzt -install-tools -o modules.install-checker.exclude=selenium,gatling,tsung,siege,ab,k6,external-results-loader,locust,junit,testng,rSpec,mocha,nunit,xunit,wdio,robot,newman
RUN rm -rf /root/.bzt/selenium-taurus
RUN mkdir /bzt-configs /tmp/artifacts
ADD ./load-test.sh /bzt-configs/
ADD /*.jar /bzt-configs/
ADD /*.py /bzt-configs/

RUN chmod 755 /bzt-configs/load-test.sh
RUN chmod 755 /bzt-configs/ecslister.py
RUN chmod 755 /bzt-configs/ecscontroller.py
RUN chmod 755 /bzt-configs/jar_updater.py
RUN python3.11 /bzt-configs/jar_updater.py

# Remove jar files from /tmp
RUN rm -rf /tmp/jmeter-plugins-manager-1* && \
    rm -rf /usr/local/lib/python3.11/site-packages/setuptools-65.5.0.dist-info && \
    rm -rf /usr/local/lib/python3.11/site-packages/urllib3-1.26.17.dist-info
```

```
# Add settings file to capture the output logs from bzt cli
RUN mkdir -p /etc/bzt.d && echo '{"settings": {"artifacts-dir": "/tmp/artifacts"}}' > /
etc/bzt.d/90-artifacts-dir.json

WORKDIR /bzt-configs
ENTRYPOINT ["/load-test.sh"]
```

Zusätzlich zu einer Container-Datei enthält das Verzeichnis das folgende Bash-Skript, das die Testkonfiguration von Amazon S3 herunterlädt, bevor das Taurus/Blazemeter Programm ausgeführt wird.

```
#!/bin/bash

# set a uuid for the results xml file name in S3
UUID=$(cat /proc/sys/kernel/random/uuid)
pypid=0
echo "S3_BUCKET:: ${S3_BUCKET}"
echo "TEST_ID:: ${TEST_ID}"
echo "TEST_TYPE:: ${TEST_TYPE}"
echo "FILE_TYPE:: ${FILE_TYPE}"
echo "PREFIX:: ${PREFIX}"
echo "UUID:: ${UUID}"
echo "LIVE_DATA_ENABLED:: ${LIVE_DATA_ENABLED}"
echo "MAIN_STACK_REGION:: ${MAIN_STACK_REGION}"

cat /proc/self/cgroup
TASK_ID=$(grep -oE '[a-f0-9]{32}' /proc/self/cgroup | head -n 1)
echo $TASK_ID

sigterm_handler() {
    if [ $pypid -ne 0 ]; then
        echo "container received SIGTERM."
        kill -15 $pypid
        wait $pypid
        exit 143 #128 + 15
    fi
}
trap 'sigterm_handler' SIGTERM

echo "Download test scenario"
aws s3 cp s3://$S3_BUCKET/test-scenarios/$TEST_ID-$AWS_REGION.json test.json --region
$MAIN_STACK_REGION
```

```
# Set the default log file values to jmeter
LOG_FILE="jmeter.log"
OUT_FILE="jmeter.out"
ERR_FILE="jmeter.err"
KPI_EXT="jtl"

# download JMeter jmx file
if [ "$TEST_TYPE" != "simple" ]; then
    # setting the log file values to the test type
    LOG_FILE="${TEST_TYPE}.log"
    OUT_FILE="${TEST_TYPE}.out"
    ERR_FILE="${TEST_TYPE}.err"

    # set variables based on TEST_TYPE
    if [ "$TEST_TYPE" == "jmeter" ]; then
        EXT="jmx"
        TYPE_NAME="JMeter"
        # Copy *.jar to JMeter library path. See the Taurus JMeter path: https://
gettaurus.org/docs/JMeter/
        JMETER_LIB_PATH=`find ~/.bzt/jmeter-taurus -type d -name "lib"`
        echo "cp $PWD/*.jar $JMETER_LIB_PATH"
        cp $PWD/*.jar $JMETER_LIB_PATH
    elif [ "$TEST_TYPE" == "k6" ]; then
        curl --output /tmp/artifacts/k6.rpm https://dl.k6.io/rpm/x86_64/k6-v0.58.0-
amd64.rpm
        rpm -ivh /tmp/artifacts/k6.rpm
        dnf install -y k6
        rm -rf /tmp/artifacts/k6.rpm
        EXT="js"
        KPI_EXT="csv"
        TYPE_NAME="K6"
    elif [ "$TEST_TYPE" == "locust" ]; then
        EXT="py"
        TYPE_NAME="Locust"
    fi

    if [ "$FILE_TYPE" != "zip" ]; then
        aws s3 cp s3://$S3_BUCKET/public/test-scenarios/$TEST_TYPE/$TEST_ID.$EXT ./ --
region $MAIN_STACK_REGION
    else
        aws s3 cp s3://$S3_BUCKET/public/test-scenarios/$TEST_TYPE/$TEST_ID.zip ./ --region
$MAIN_STACK_REGION
        unzip $TEST_ID.zip
    fi
fi
```

```

echo "UNZIPPED"
ls -l

# If zip and locust, make sure to pick locustfile
if [ "$TEST_TYPE" != "locust" ]; then
    TEST_SCRIPT=$(find . -name "*.${EXT}" | head -n 1)
else
    TEST_SCRIPT=$(find . -name "locustfile.py" | head -n 1)
fi
# only looks for the first test script file.
TEST_SCRIPT=`find . -name "*.${EXT}" | head -n 1`
echo $TEST_SCRIPT
if [ -z "$TEST_SCRIPT" ]; then
    echo "There is no test script (}.${EXT}) in the zip file."
    exit 1
fi

sed -i -e "s|${TEST_ID}.${EXT}|${TEST_SCRIPT}|g" test.json

# copy bundled plugin jars to jmeter extension folder to make them available to
jmeter
BUNDLED_PLUGIN_DIR=`find $PWD -type d -name "plugins" | head -n 1`
# attempt to copy only if a /plugins folder is present in upload
if [ -z "$BUNDLED_PLUGIN_DIR" ]; then
    echo "skipping plugin installation (no /plugins folder in upload)"
else
    # ensure the jmeter extensions folder exists
    JMETER_EXT_PATH=`find ~/.bzt/jmeter-taurus -type d -name "ext"`
    if [ -z "$JMETER_EXT_PATH" ]; then
        # fail fast - if plugins bundled they will be needed for the tests
        echo "jmeter extension path (~/.bzt/jmeter-taurus/**/ext) not found - cannot
install bundled plugins"
        exit 1
    fi
    cp -v $BUNDLED_PLUGIN_DIR/*.jar $JMETER_EXT_PATH
fi
fi

#Download python script
if [ -z "$IPNETWORK" ]; then
    python3.11 -u $SCRIPT $TIMEOUT &
    pypid=$!
    wait $pypid

```

```

    pypid=0
else
    aws s3 cp s3://$S3_BUCKET/Container_IPs/${TEST_ID}_IPHOSTS_${AWS_REGION}.txt ./ --
region $MAIN_STACK_REGION
    export IPHOSTS=$(cat ${TEST_ID}_IPHOSTS_${AWS_REGION}.txt)
    python3.11 -u $SCRIPT $IPNETWORK $IPHOSTS
fi

echo "Running test"

stdbuf -i0 -o0 -e0 bzt test.json -o modules.console.disable=true | stdbuf -i0 -o0 -e0
tee -a result.tmp | sed -u -e "s|^|${TEST_ID} $LIVE_DATA_ENABLED |"
CALCULATED_DURATION=`cat result.tmp | grep -m1 "Test duration" | awk -F ' ' '{ print
$5 }' | awk -F ':' '{ print ($1 * 3600) + ($2 * 60) + $3 }`

# upload custom results to S3 if any
# every file goes under $TEST_ID/$PREFIX/$UUID to distinguish the result correctly
if [ "$TEST_TYPE" != "simple" ]; then
    if [ "$FILE_TYPE" != "zip" ]; then
        cat $TEST_ID.$EXT | grep filename > results.txt
    else
        cat $TEST_SCRIPT | grep filename > results.txt
    fi

    if [ -f results.txt ]; then
        sed -i -e 's/<stringProp name="filename">//g' results.txt
        sed -i -e 's/<\/stringProp>//g' results.txt
        sed -i -e 's/ //g' results.txt

        echo "Files to upload as results"
        cat results.txt

        files=(`cat results.txt`)
        extensions=()
        for f in "${files[@]}"; do
            ext="${f##*}"
            if [[ ! " ${extensions[@]} " =~ " $ext " ]]; then
                extensions+=("$ext")
            fi
        done

        # Find all files in the current folder with the same extensions
        all_files=()
        for ext in "${extensions[@]}"; do

```

```

    for f in *."$ext"; do
        all_files+=("$f")
    done
done

for f in "${all_files[@]}"; do
    p="s3://$S3_BUCKET/results/$TEST_ID/${TYPE_NAME}_Result/$PREFIX/$UUID/$f"
    if [[ $f = /* ]]; then
        p="s3://$S3_BUCKET/results/$TEST_ID/${TYPE_NAME}_Result/$PREFIX/$UUID$f"
    fi

    echo "Uploading $p"
    aws s3 cp $f $p --region $MAIN_STACK_REGION
done
fi

fi

if [ -f /tmp/artifacts/results.xml ]; then

    # Insert the Task ID at the same level as <FinalStatus>
    curl -s $ECS_CONTAINER_METADATA_URI_V4/task
    Task_CPU=$(curl -s $ECS_CONTAINER_METADATA_URI_V4/task | jq '.Limits.CPU')
    Task_Memory=$(curl -s $ECS_CONTAINER_METADATA_URI_V4/task | jq '.Limits.Memory')
    START_TIME=$(curl -s "$ECS_CONTAINER_METADATA_URI_V4/task" | jq -r
'.Containers[0].StartedAt')
    # Convert start time to seconds since epoch
    START_TIME_EPOCH=$(date -d "$START_TIME" +%s)
    # Calculate elapsed time in seconds
    CURRENT_TIME_EPOCH=$(date +%s)
    ECS_DURATION=$((CURRENT_TIME_EPOCH - START_TIME_EPOCH))

    sed -i.bak 's/<\FinalStatus>/<TaskId>"$TASK_ID"<\TaskId><\FinalStatus>/' /tmp/
artifacts/results.xml
    sed -i 's/<\FinalStatus>/<TaskCPU>"$Task_CPU"<\TaskCPU><\FinalStatus>/' /tmp/
artifacts/results.xml
    sed -i 's/<\FinalStatus>/<TaskMemory>"$Task_Memory"<\TaskMemory><\
FinalStatus>/' /tmp/artifacts/results.xml
    sed -i 's/<\FinalStatus>/<ECSDuration>"$ECS_DURATION"<\ECSDuration><\
FinalStatus>/' /tmp/artifacts/results.xml

    echo "Validating Test Duration"
    TEST_DURATION=$(grep -E '<TestDuration>[0-9]+.[0-9]+</TestDuration>' /tmp/artifacts/
results.xml | sed -e 's/<TestDuration> //' | sed -e 's/<\TestDuration> //')

```

```

if (( $(echo "$TEST_DURATION > $CALCULATED_DURATION" | bc -l) )); then
    echo "Updating test duration: $CALCULATED_DURATION s"
    sed -i.bak.td 's/<TestDuration>[0-9]*\.[0-9]*</TestDuration>/
<TestDuration>'"$CALCULATED_DURATION"'</TestDuration>/' /tmp/artifacts/results.xml
fi

if [ "$TEST_TYPE" == "simple" ]; then
    TEST_TYPE="jmeter"
fi

echo "Uploading results, bzt log, and JMeter log, out, and err files"
aws s3 cp /tmp/artifacts/results.xml s3://$S3_BUCKET/results/${TEST_ID}/${PREFIX}-
${UUID}-${AWS_REGION}.xml --region $MAIN_STACK_REGION
aws s3 cp /tmp/artifacts/bzt.log s3://$S3_BUCKET/results/${TEST_ID}/bzt-${PREFIX}-
${UUID}-${AWS_REGION}.log --region $MAIN_STACK_REGION
aws s3 cp /tmp/artifacts/$LOG_FILE s3://$S3_BUCKET/results/${TEST_ID}/${TEST_TYPE}-
${PREFIX}-${UUID}-${AWS_REGION}.log --region $MAIN_STACK_REGION
aws s3 cp /tmp/artifacts/$OUT_FILE s3://$S3_BUCKET/results/${TEST_ID}/${TEST_TYPE}-
${PREFIX}-${UUID}-${AWS_REGION}.out --region $MAIN_STACK_REGION
aws s3 cp /tmp/artifacts/$ERR_FILE s3://$S3_BUCKET/results/${TEST_ID}/${TEST_TYPE}-
${PREFIX}-${UUID}-${AWS_REGION}.err --region $MAIN_STACK_REGION
aws s3 cp /tmp/artifacts/kpi.${KPI_EXT} s3://$S3_BUCKET/results/${TEST_ID}/kpi-
${PREFIX}-${UUID}-${AWS_REGION}.${KPI_EXT} --region $MAIN_STACK_REGION

else
    echo "An error occurred while the test was running."
fi

```

Neben dem [Dockerfile](#) und dem Bash-Skript sind auch zwei Python-Skripte im Verzeichnis enthalten. Jede Aufgabe führt ein Python-Skript innerhalb des Bash-Skripts aus. Die Worker-Tasks führen das `ecslister.py` Skript aus, während die Leader-Task das `ecscontroller.py` Skript ausführt. Das `ecslister.py` Skript erstellt einen Socket auf Port 50000 und wartet auf eine Nachricht. Das `ecscontroller.py` Skript stellt eine Verbindung zum Socket her und sendet die Starttestnachricht an die Worker-Aufgaben, sodass sie gleichzeitig starten können.

## 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 auch verwenden APIs , 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.

### Szenarien

- [GET /scenarios](#)
- [POST /szenarien](#)
- [OPTIONEN /Szenarien](#)
- [GET /scenarios/ {testId}](#)
- [POSTE /scenarios/ {testId}](#)
- [LÖSCHEN SIE /scenarios/ {testId}](#)
- [OPTIONEN /scenarios/ {testID}](#)

### Aufgaben

- [GET /tasks](#)
- [OPTIONEN /Aufgaben](#)

### Regionen

- [HOLEN SIE SICH /regions](#)

- [OPTIONEN /regions](#)

## GET /scenarios

### Beschreibung

Die GET /scenarios Operation ermöglicht es Ihnen, eine Liste von Testszenarien abzurufen.

### Antwort

Name	Beschreibung
data	Eine Liste von Szenarien, einschließlich der ID, des Namens, der Beschreibung, des Status und der Laufzeit für jeden Test

## POST /szenarien

### Beschreibung

Die POST /scenarios Operation ermöglicht es Ihnen, ein Testszenario zu erstellen oder zu planen.

### Anforderungstext

Name	Beschreibung
testName	Der Name des Tests
testDescription	Die Beschreibung des Tests
testTaskConfigs	Ein Objekt, das <code>concurrency</code> (die Anzahl der parallel Läufe), <code>taskCount</code> (die Anzahl der Aufgaben, die zur Ausführung eines Tests erforderlich sind) und <code>region</code> für das Szenario angibt
testScenario	Die Testspezifikation, einschließlich Parallelität, Testzeit, Host und Methode für den Test

Name	Beschreibung
testType	Der Testtyp (zum Beispielsimple,jmeter)
fileType	Der Upload-Dateityp (z. B.none,script,zip)
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 undstart)
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.
recurrence	Die Wiederholung eines geplanten Tests. Wird nur bereitgestellt, wenn ein wiederkehrender Test geplant wird (z. B.daily, weeklybiweekly, odermonthly)

## Antwort

Name	Beschreibung
testId	Die eindeutige ID des Tests

Name	Beschreibung
testName	Der Name des Tests
status	Der Status des Tests

## OPTIONEN/Szenarien

### Beschreibung

Die `OPTIONS /scenarios` Operation liefert eine Antwort auf die Anfrage mit den richtigen CORS-Antwortheadern.

### Antwort

Name	Beschreibung
testId	Die eindeutige ID des Tests
testName	Der Name des Tests
status	Der Status des Tests

## GET /scenarios/ {testId}

### Beschreibung

Die `GET /scenarios/{testId}` Operation ermöglicht es Ihnen, die Details eines bestimmten Testszenarios abzurufen.

### Parameter anfordern

#### testId

- Die eindeutige ID des Tests

Typ: Zeichenfolge

Erforderlich: Ja

## Antwort

Name	Beschreibung
testId	Die eindeutige ID des Tests
testName	Der Name des Tests
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)
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 zum Ausführen 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

Name	Beschreibung
nextRun	Der nächste geplante Lauf (zum Beispiel 2017-04-22 17:18:00 )
scheduleRecurrence	Die Wiederholung des Tests (z. B., daily, weekly, biweekly, monthly)

## POST /scenarios/ {TestID}

### Beschreibung

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	Beschreibung
status	Der Status des Tests

## LÖSCHEN SIE /scenarios/ {testId}

### Beschreibung

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	Beschreibung
status	Der Status des Tests

## OPTIONEN /scenarios/ {testId}

### Beschreibung

Die `OPTIONS /scenarios/{testId}` Operation liefert eine Antwort auf die Anfrage mit den richtigen CORS-Antwortheadern.

## Antwort

Name	Beschreibung
testId	Die eindeutige ID des Tests
testName	Der Name des Tests
testDescription	Die Beschreibung des Tests
testType	Die Art des Tests, der ausgeführt wird (z. B. <code>simple,jmeter</code> )
fileType	Der Typ der Datei, die hochgeladen wird (z. B. <code>none,script,zip</code> )

Name	Beschreibung
<code>status</code>	Der Status des Tests
<code>startTime</code>	Die Uhrzeit und das Datum, an dem der letzte Test gestartet wurde
<code>endTime</code>	Uhrzeit und Datum, an dem der letzte Test beendet wurde
<code>testScenario</code>	Die Testspezifikation, einschließlich Parallelität, Testzeit, Host und Methode für den Test
<code>taskCount</code>	Die Anzahl der Aufgaben, die zur Ausführung des Tests erforderlich sind
<code>taskIds</code>	Eine Liste von Aufgaben IDs zum Ausführen von Tests
<code>results</code>	Die endgültigen Ergebnisse des Tests
<code>history</code>	Eine Liste der Endergebnisse vergangener Tests
<code>errorReason</code>	Eine Fehlermeldung, die generiert wird, wenn ein Fehler auftritt

## GET /tasks

### Beschreibung

Mit GET /tasks diesem Vorgang können Sie eine Liste der laufenden Amazon Elastic Container Service (Amazon ECS) -Aufgaben abrufen.

## Antwort

Name	Beschreibung
tasks	Eine Liste von Aufgaben IDs zum Ausführen von Tests

## OPTIONEN/Aufgaben

### Beschreibung

Der Vorgang `OPTIONS /tasks tasks` liefert eine Antwort auf die Anfrage mit den richtigen CORS-Antwortheadern.

### Antwort

Name	Beschreibung
taskIds	Eine Liste von Aufgaben IDs zum Ausführen von Tests

## GET /regions

### Beschreibung

Mit diesem `GET /regions` Vorgang können Sie die regionalen Ressourceninformationen abrufen, die für die Durchführung eines Tests in dieser Region erforderlich sind.

### Antwort

Name	Beschreibung
testId	Die Regions-ID
ecsCloudWatchLogGroup	Der Name der CloudWatch Amazon-Protokollgruppe für die Amazon Fargate-Aufgaben in der Region

Name	Beschreibung
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
taskImage	Der Name des Task-Images in der Region
taskSecurityGroup	Die ID der Sicherheitsgruppe in der Region

## OPTIONEN /Regionen

### Beschreibung

Der `OPTIONS /regions` Vorgang liefert eine Antwort auf die Anfrage mit den richtigen CORS-Antwortheadern.

### Antwort

Name	Beschreibung
testId	Die Regions-ID
ecsCloudWatchLogGroup	Der Name der CloudWatch Amazon-Protokollgruppe für die Amazon 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

Name	Beschreibung
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
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 derzeit unterstützten Benutzer zu erhöhen, erhöhen Sie die Container-Ressourcen. Auf diese Weise können Sie den CPUs Arbeitsspeicher erhöhen, um der Zunahme gleichzeitiger Benutzer gerecht zu werden.

### Erstellen Sie eine neue Revision der Aufgabendefinition

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 den Task-Speicher und die Task-CPU.
  - b. Überprüfen Sie unter Containerdefinitionen die Grenzwerte für Hard/Soft Memory. 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 Hardlimit auf den gewünschten Speicher.
  - d. Wählen Sie Aktualisieren.
6. Wählen Sie auf der Seite Neue Revision erstellen die Option Erstellen aus.

7. Nachdem die Aufgabendefinition erfolgreich erstellt wurde, notieren Sie sich den Namen der neuen Aufgabendefinition. Dieser Name beinhaltet die Versionsnummer, zum Beispiel:  
[replaceable] <stackName>`- EcsTaskDefinition -<system-generated-random-Hash>:  
[austauschbar]<system-generated-versionNumber>.

## Aktualisieren Sie die DynamoDB-Tabelle

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>`- DLTest RunnerStorage DLScenarios Table-  
<system-generated-random-Hash>
4. Wählen Sie das Element aus, das der Region entspricht, in der Sie die Aufgabendefinition geändert haben. Zum Beispiel Region-<region-name>.
5. Aktualisieren Sie das TaskDefinition-Attribut mit der neuen Aufgabendefinition.

# Referenz

Dieser Abschnitt enthält Informationen zu einer optionalen Funktion zum Sammeln einzigartiger Messwerte für diese Lösung, Hinweise auf verwandte Ressourcen und eine Liste der Entwickler, die zu dieser Lösung beigetragen haben.

## Anonymisierte Datenerfassung

Diese Lösung beinhaltet eine Option zum Senden anonymisierter Betriebsmetriken an AWS. Wir verwenden diese Daten, um besser zu verstehen, wie Kunden diese Lösung und die damit verbundenen Services und Produkte nutzen. Wenn sie aufgerufen werden, werden die folgenden Informationen gesammelt und an AWS gesendet:

- Lösungs-ID — Die AWS-Lösungs-ID
- Eindeutige ID (UUID) — Zufällig generierte, eindeutige Kennung für jede Lösungsbereitstellung
- Zeitstempel — Zeitstempel für die Datenerfassung
- Testtyp — Die Art des Tests, der ausgeführt wird
- Dateityp — Der Typ der Datei, die hochgeladen wird
- Anzahl der Aufgaben — Die Anzahl der Aufgaben für jeden Test, der über die API der Lösung eingereicht wurde
- Aufgabendauer — Die Gesamtlaufzeit aller Aufgaben, die zur Ausführung eines Tests benötigt werden
- Testergebnis — Das Ergebnis des Tests, der ausgeführt wurde

AWS ist Eigentümer der im Rahmen dieser Umfrage gesammelten Daten. Die Datenerfassung unterliegt der [AWS-Datenschutzrichtlinie](#). Um diese Funktion zu deaktivieren, führen Sie die folgenden Schritte aus, bevor Sie die CloudFormation AWS-Vorlage starten.

1. Laden Sie die [CloudFormation AWS-Vorlage](#) auf Ihre lokale Festplatte herunter.
2. Öffnen Sie die CloudFormation AWS-Vorlage mit einem Texteditor.
3. Ändern Sie den Abschnitt CloudFormation AWS-Vorlagenzuordnung von:

Solution:  
Config:

```
SendAnonymousData: "Yes"
```

auf:

```
Solution:  
Config:  
  SendAnonymousData: "No"
```

4. Melden Sie sich bei der [CloudFormation AWS-Konsole](#) an.
5. Wählen Sie Stack erstellen aus.
6. Wählen Sie auf der Seite Stack erstellen im Abschnitt Vorlage angeben die Option Vorlagendatei hochladen aus.
7. Wählen Sie unter Vorlagendatei hochladen die Option Datei auswählen und wählen Sie die bearbeitete Vorlage von Ihrem lokalen Laufwerk aus.
8. Wählen Sie Weiter und folgen Sie den Schritten unter [Stack starten](#) im Abschnitt Lösung bereitstellen dieses Handbuchs.

## Mitwirkende

- Tom Nightingale
- Fernando Dingler
- Beomseok Lee
- George Lenz
- Erin McGill
- Dmitri López
- Kamyar Ziabari
- Bassem Wanis
- Garvit Singh
- Nikhil Reddy
- Simon Kroll

# 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.