Guida all'implementazione

Test di carico distribuito su AWS

Test di carico distribuito su AWS: Guida all'implementazione

Table of Contents

Automatizza il test delle tue applicazioni software su larga scala

Data di pubblicazione: dicembre 2025

Distributed Load Testing on AWS ti aiuta ad automatizzare i test delle prestazioni delle tue applicazioni software su larga scala per identificare i colli di bottiglia prima di rilasciare l'applicazione. Questa soluzione simula migliaia di utenti connessi che generano richieste HTTP a una velocità sostenuta senza la necessità di fornire server.

Questa soluzione sfrutta Amazon Elastic Container Service (Amazon ECS) su AWS Fargate per distribuire contenitori che eseguono simulazioni di test di carico e offre le seguenti funzionalità:

- Implementa Amazon ECS su contenitori AWS Fargate eseguiti in modo indipendente per testare la capacità di carico della tua applicazione.
- Simula decine di migliaia di utenti simultanei in più regioni AWS generando richieste a un ritmo continuo.
- Personalizza i test delle tue applicazioni utilizzando K6 JMeter, gli script di test Locust o una semplice configurazione degli endpoint HTTP.
- Pianifica i test di carico in modo che vengano eseguiti immediatamente, in una data e ora future o in base a una pianificazione ricorrente.
- Esegui più test di carico contemporaneamente in diversi scenari e regioni.

Questa guida all'implementazione fornisce una panoramica della soluzione Distributed Load Testing on AWS, della sua architettura e dei suoi componenti di riferimento, considerazioni per la pianificazione della distribuzione e delle fasi di configurazione per la distribuzione della soluzione nel cloud Amazon Web Services (AWS). Include collegamenti a un CloudFormation modello AWS che avvia e configura i servizi AWS necessari per distribuire questa soluzione utilizzando le best practice di AWS per la sicurezza e la disponibilità.

Il pubblico previsto per l'utilizzo delle caratteristiche e delle funzionalità di questa soluzione nel proprio ambiente include architetti di infrastrutture IT, amministratori e DevOps professionisti con esperienza pratica di architettura nel cloud AWS.

Utilizza questa tabella di navigazione per trovare rapidamente le risposte a queste domande:

Se vuoi.	Leggi..
Conosci il costo di esecuzione di questa soluzione. Il costo stimato per l'esecuzione di questa soluzione nella regione Stati Uniti orientali (Virginia settentrionale) è di 30,90 USD al mese per le risorse AWS.	Costo
Comprendi le considerazioni sulla sicurezza relative a questa soluzione.	Sicurezza
Scopri come pianificare le quote per questa soluzione.	Quote
Scopri quali regioni AWS supportano questa soluzione.	Regioni AWS supportate
Scopri il server MCP opzionale per l'analisi dei test di carico assistita dall'intelligenza artificiale.	Integrazione con server MCP
Visualizza o scarica il CloudFormation modello AWS incluso in questa soluzione per distribuire automaticamente le risorse dell'infrastruttura (lo «stack») per questa soluzione.	CloudFormation Modello AWS
Accedi al codice sorgente e, facoltativamente, utilizza AWS Cloud Development Kit (AWS CDK) per distribuire la soluzione.	GitHub repository

Funzionalità

La soluzione offre le seguenti funzionalità:

Supporto per Multiple Test Framework

Supporta JMeter gli script di test K6 e Locust, oltre a semplici test sugli endpoint HTTP senza richiedere script personalizzati. Per ulteriori informazioni, consulta Tipi di test nella sezione Dettagli sull'architettura.

Simulazione di un carico utente elevato

Simula decine di migliaia di utenti virtuali simultanei per sottoporre a stress test l'applicazione in condizioni di carico realistiche.

Distribuzione del carico in più regioni

Distribuisce test di carico su più regioni AWS per simulare il traffico utente distribuito geograficamente e valutare le prestazioni globali.

Pianificazione flessibile dei test

Pianifica l'esecuzione dei test immediatamente, in una data e ora future specifiche o in base a una pianificazione ricorrente utilizzando le espressioni cron per i test di regressione automatici.

monitoraggio in tempo reale

Fornisce uno streaming opzionale di dati in tempo reale per monitorare l'avanzamento dei test con metriche in tempo reale tra cui tempi di risposta, conteggi virtuali degli utenti e percentuali di successo delle richieste.

Risultati completi dei test

Visualizza i risultati dettagliati dei test con metriche delle prestazioni, percentili (p50, p90, p95, p99), analisi degli errori e artefatti scaricabili per l'analisi offline.

Confronto di base

Indica le esecuzioni dei test di base per il confronto delle prestazioni al fine di tenere traccia dei miglioramenti o delle regressioni nel tempo.

Flessibilità degli endpoint

Testa qualsiasi endpoint HTTP o HTTPS nelle regioni AWS, negli ambienti locali o in altri provider cloud.

Console Web intuitiva

Fornisce una console basata sul Web per la creazione, la gestione e il monitoraggio dei test senza la necessità di interazione da riga di comando.

Analisi assistita dall'intelligenza artificiale (opzionale)

Si integra con gli strumenti di sviluppo AI tramite il server Model Context Protocol (MCP) per l'analisi intelligente dei dati dei test di carico.

Supporto per protocolli multipli

Supporta vari protocolli tra cui HTTP, HTTPS WebSocket, JDBC, JMS, FTP e gRPC tramite script di test personalizzati.

Vantaggi

La soluzione offre i seguenti vantaggi:

Test completi delle prestazioni

Supporta test di carico, stress test e test di resistenza per valutare a fondo le prestazioni delle applicazioni in varie condizioni.

Rilevamento precoce dei problemi

Identifica i rallentamenti delle prestazioni, le perdite di memoria e i problemi di scalabilità prima dell'implementazione in produzione, riducendo il rischio di interruzioni.

Simulazione di utilizzo nel mondo reale

Simula accuratamente il comportamento degli utenti e i modelli di traffico nel mondo reale per convalidare le prestazioni delle applicazioni in condizioni realistiche.

Performance Insights utilizzabili

Fornisce metriche dettagliate, percentili e analisi degli errori per comprendere il comportamento delle applicazioni e guidare gli sforzi di ottimizzazione.

Flussi di lavoro di test automatizzati

Consente test programmati e ricorrenti per il monitoraggio continuo delle prestazioni e i test di regressione senza intervento manuale.

Infrastruttura efficiente in termini di costi

Utilizza contenitori AWS Fargate serverless pay-per-use con prezzi, eliminando la necessità di un'infrastruttura di test dedicata e di canoni di abbonamento continui.

Distribuzione rapida dei test

Implementa e ridimensiona l'infrastruttura di test in pochi minuti senza fornire o gestire server.

Interrogazione semplificata dei risultati dei test

Si integra con gli strumenti di sviluppo AI tramite un server opzionale Model Context Protocol (MCP), che consente query in linguaggio naturale e analisi intelligente dei dati dei test di carico per informazioni e risoluzione dei problemi più rapide.

Casi d’uso

Validazione pre-produzione

Testa le applicazioni web e mobili in condizioni di carico simili a quelle di produzione prima di lanciare una nuova versione per convalidare le prestazioni e identificare i problemi.

Pianificazione della capacità

Determina il numero massimo di utenti simultanei che l'applicazione può supportare con l'infrastruttura corrente e identifica quando è necessaria la scalabilità.

Verifica del carico di picco

Verifica che la tua infrastruttura sia in grado di gestire picchi di carico, picchi di traffico stagionali o picchi imprevisti della domanda senza un peggioramento delle prestazioni.

Ottimizzazione delle prestazioni

Identifica gli ostacoli alle prestazioni come le query lente sul database, l'esecuzione inefficiente del codice, la latenza di rete o i vincoli di risorse.

Test di regressione

Pianifica test di carico ricorrenti per rilevare le regressioni delle prestazioni introdotte da nuove implementazioni di codice o modifiche all'infrastruttura.

Valutazione globale delle prestazioni

Valuta le prestazioni delle applicazioni da più aree geografiche per garantire un'esperienza utente coerente per un pubblico globale.

Test di carico delle API

Testa gli endpoint REST APIs, GraphQL o i microservizi per convalidare i tempi di risposta, la velocità effettiva e i tassi di errore sotto carico.

Integrazione della pipeline CI/CD

Integra i test automatizzati delle prestazioni in pipeline di integrazione e distribuzione continue per individuare i problemi di prestazioni nelle prime fasi del ciclo di sviluppo.

Test di servizi di terze parti

Verifica le prestazioni e l'affidabilità dei servizi APIs di terze parti da cui dipende l'applicazione in varie condizioni di carico.

Concetti e definizioni

Questa sezione descrive i concetti chiave e definisce la terminologia specifica di questa soluzione:

scenario

Definizione del test che include nome, descrizione, numero di attività, concorrenza, regione AWS, ramp-up, hold-for, tipo di test, data di pianificazione e configurazioni di ricorrenza.

conteggio delle attività

Numero di container che verranno lanciati nel cluster Fargate per eseguire lo scenario di test. Non verranno create attività aggiuntive una volta raggiunto il limite dell'account per le risorse Fargate. Tuttavia, le attività già in esecuzione continueranno.

concurrency

La concorrenza (numero di utenti virtuali simultanei per attività). La concorrenza consigliata in base alle impostazioni predefinite è 200. La concorrenza è limitata dalla CPU e dalla memoria. Per i test basati su Apache JMeter, una maggiore concorrenza aumenta la memoria utilizzata dalla JVM per l'attività ECS. L'impostazione predefinita di ECS Task Definition crea attività con 4 GB di memoria. Si consiglia di iniziare con valori di concorrenza inferiori per 1 attività e di monitorare le CloudWatch metriche ECS per il Task Cluster. Fai riferimento ai parametri di utilizzo del cluster Amazon ECS.

accelerazione

Il periodo di tempo necessario per passare gradualmente da zero al livello di concorrenza previsto.

tieni premuto per

Il periodo di tempo necessario per mantenere il livello di concorrenza previsto dopo il completamento dell'accelerazione.

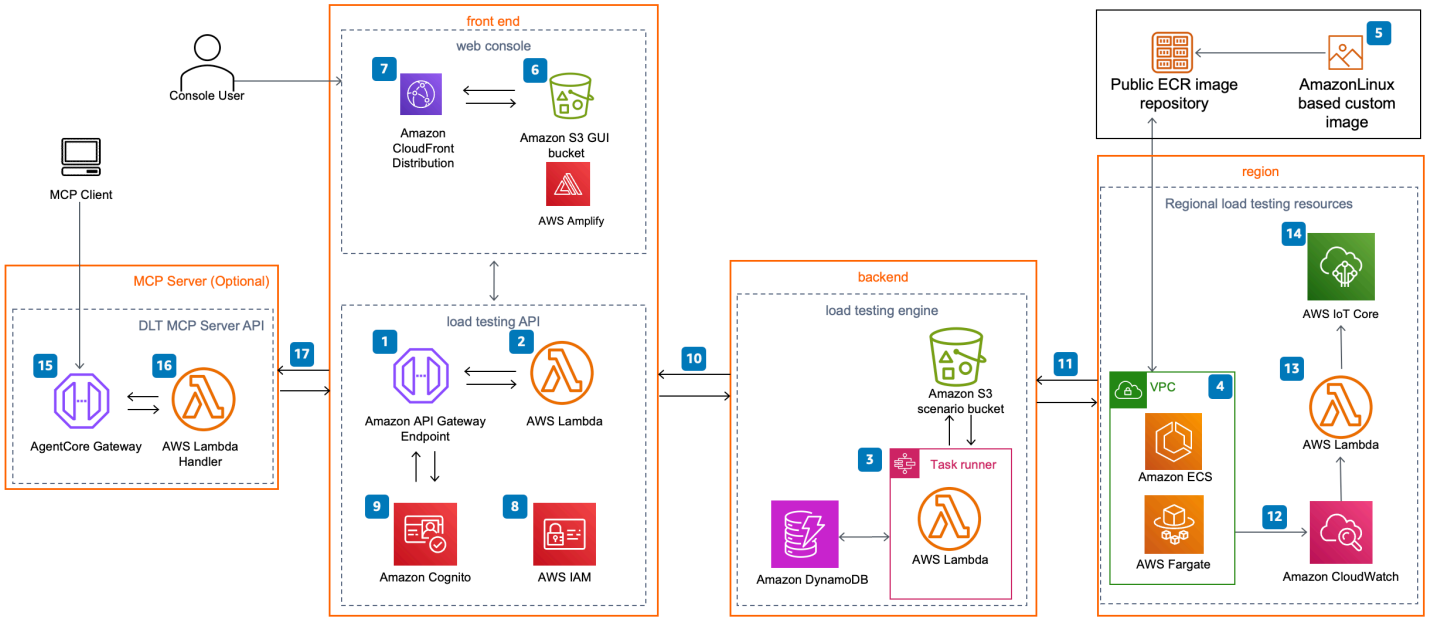
Per un riferimento generale ai termini di AWS, consulta il Glossario AWS.

Panoramica dell’architettura

Diagramma architetturale

La distribuzione di questa soluzione con i parametri predefiniti distribuisce i seguenti componenti nel tuo account AWS.

Test di carico distribuito sull'architettura AWS su AWS



> **Note**
>
> Le CloudFormation risorse AWS vengono create a partire da costrutti di AWS Cloud Development Kit (AWS CDK).

Il flusso di processo di alto livello per i componenti della soluzione distribuiti con il CloudFormation modello AWS è il seguente:

1. Un'API di test di carico distribuita sfrutta Amazon API Gateway per richiamare i microservizi della soluzione (funzioni AWS Lambda).
2. I microservizi forniscono la logica aziendale per gestire i dati di test ed eseguire i test.

3. Questi microservizi interagiscono con Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB e AWS Step Functions per archiviare dettagli e risultati dello scenario di test e orchestrare l'esecuzione dei test.
4. Viene distribuita una topologia di rete Amazon Virtual Private Cloud (Amazon VPC) contenente i contenitori Amazon Elastic Container Service (Amazon ECS) della soluzione in esecuzione su AWS Fargate.
5. I container utilizzano un'immagine base di Amazon Linux 2023 con il framework di load testing Taurus installato. Taurus è un framework di automazione dei test open source che supporta K6 JMeter, Locust e altri strumenti di test. L'immagine del contenitore è conforme a Open Container Initiative (OCI) e ospitata da AWS in un repository pubblico Amazon Elastic Container Registry (Amazon ECR). Per ulteriori informazioni, consulta la sezione Personalizzazione dell'immagine del contenitore.
6. Una console Web basata su AWS Amplify viene distribuita in un bucket S3 configurato per l'hosting web statico.
7. Amazon CloudFront fornisce un accesso pubblico e sicuro ai contenuti del bucket del sito Web della soluzione.
8. Durante la configurazione iniziale, la soluzione crea un ruolo di amministratore predefinito (ruolo IAM) e invia un invito di accesso a un indirizzo e-mail utente specificato dal cliente.
9. Un pool di utenti di Amazon Cognito gestisce l'accesso degli utenti alla console, all'API del tester di carico distribuito e al server MCP.
10. Dopo aver distribuito questa soluzione, puoi utilizzare la console Web o creare ed eseguire scenari APIs di test che definiscono una serie di attività.
11. I microservizi utilizzano questo scenario di test per eseguire attività ECS su Fargate nelle regioni specificate.
12. Al termine del test, la soluzione archivia i risultati in S3 e DynamoDB e registra l'output in Amazon. CloudWatch
13. Se si abilita l'opzione live data, la soluzione invia CloudWatch i log delle attività di Fargate a una funzione Lambda durante il test per ogni regione in cui viene eseguito il test.
14. La funzione Lambda pubblica i dati nell'argomento corrispondente in AWS IoT Core nella regione in cui è stato distribuito lo stack principale. La console web sottoscrive l'argomento e visualizza i dati in tempo reale durante l'esecuzione del test.

> **Note**
>
> I passaggi seguenti descrivono l'integrazione opzionale del server MCP per l'analisi dei test di carico assistita dall'intelligenza artificiale. Questo componente viene distribuito solo se si seleziona l'opzione MCP Server durante la distribuzione della soluzione.

15. Un client MCP (strumento di sviluppo AI) si connette all'endpoint AWS AgentCore Gateway per accedere ai dati della soluzione Distributed Load Testing tramite il Model Context Protocol. AgentCore Gateway convalida il token di autenticazione Cognito dell'utente per garantire l'accesso autorizzato al server MCP.
16. Una volta completata l'autenticazione, AgentCore Gateway inoltra la richiesta dello strumento MCP alla funzione Lambda del server DLT MCP. La funzione Lambda restituisce i dati strutturati a AgentCore Gateway, che li invia al client MCP per analisi e approfondimenti assistiti dall'intelligenza artificiale.
17. La funzione Lambda elabora la richiesta e interroga le risorse AWS appropriate (tabelle DynamoDB, bucket S3 o CloudWatch log) per recuperare i dati di test di carico richiesti.

Considerazioni sulla progettazione di AWS Well-Architected

Questa soluzione utilizza le best practice di AWS Well-Architected Framework, che aiuta i clienti a progettare e gestire carichi di lavoro affidabili, sicuri, efficienti ed economici nel cloud.

Questa sezione descrive in che modo i principi di progettazione e le migliori pratiche di Well-Architected Framework favoriscono questa soluzione.

Eccellenza operativa

Questa sezione descrive come abbiamo progettato questa soluzione utilizzando i principi e le migliori pratiche del pilastro dell'eccellenza operativa.

- Tutte le risorse sono definite come infrastruttura sotto forma di codice utilizzando CloudFormation modelli AWS generati da costrutti AWS CDK.
- La soluzione utilizza le metriche CloudWatch in varie fasi per fornire osservabilità nelle funzioni Lambda, nelle attività ECS, nei bucket S3 e in altri componenti della soluzione.

Sicurezza

Questa sezione descrive come abbiamo progettato questa soluzione utilizzando i principi e le migliori pratiche del pilastro della sicurezza.

- Cognito autentica e autorizza gli utenti della console Web e le richieste API.
- Tutte le comunicazioni interservizi utilizzano ruoli AWS Identity and Access Management (IAM) con accesso con privilegi minimi, contenenti solo le autorizzazioni minime richieste.
- Tutto lo storage di dati, inclusi i bucket S3 e le tabelle DynamoDB, crittografa i dati inattivi utilizzando chiavi gestite AWS.
- La registrazione, il tracciamento e il controllo delle versioni sono abilitati ove applicabile per scopi di controllo e conformità.
- L'accesso alla rete è privato per impostazione predefinita con endpoint VPC abilitati, ove disponibili, per mantenere il traffico all'interno della rete AWS.

Note

La soluzione crea più gruppi di CloudWatch log con periodi di conservazione diversi in base al volume di log e alle considerazioni relative ai costi:

Tipo di log	Periodo di retention
Informazioni sui container ECS	1 giorno
Step Functions, log personalizzati ECS, log di accesso API Gateway	1 anno
Registri di runtime Lambda	2 anni
Log di esecuzione di API Gateway	Non scadono mai

È possibile modificare questi periodi di conservazione nella CloudWatch console in base alle proprie esigenze.

Affidabilità

Questa sezione descrive come abbiamo progettato questa soluzione utilizzando i principi e le migliori pratiche del pilastro dell'affidabilità.

- La soluzione utilizza i servizi serverless AWS laddove possibile (esempi: Lambda, API Gateway, Amazon S3, AWS Step Functions, Amazon DynamoDB e AWS Fargate) per garantire alta disponibilità e ripristino in caso di guasto del servizio.
- Tutte le elaborazioni di calcolo utilizzano funzioni Lambda o Amazon ECS su AWS Fargate.
- I dati vengono archiviati in DynamoDB e Amazon S3, quindi persistono in più zone di disponibilità per impostazione predefinita.

Efficienza delle prestazioni

Questa sezione descrive come abbiamo progettato questa soluzione utilizzando i principi e le migliori pratiche del pilastro prestazione-efficienza.

- La soluzione utilizza un'architettura serverless con la possibilità di scalare orizzontalmente in base alle esigenze.
- La soluzione può essere lanciata in qualsiasi regione che supporti i servizi AWS in questa soluzione, ad esempio: AWS Lambda, Amazon API Gateway, Amazon S3, AWS Step Functions, Amazon DynamoDB, Amazon ECS, AWS Fargate e Amazon Cognito.
- La soluzione utilizza dappertutto servizi gestiti per ridurre l'onere operativo legato all'approvvigionamento e alla gestione delle risorse.
- La soluzione viene testata e distribuita automaticamente ogni giorno per garantire la coerenza man mano che i servizi AWS cambiano, nonché esaminata da architetti di soluzioni ed esperti in materia per individuare aree da sperimentare e migliorare.

Ottimizzazione dei costi

Questa sezione descrive come abbiamo progettato questa soluzione utilizzando i principi e le migliori pratiche del pilastro dell'ottimizzazione dei costi.

- La soluzione utilizza un'architettura serverless; pertanto, ai clienti viene addebitato solo ciò che utilizzano.

- Amazon DynamoDB ridimensiona la capacità su richiesta, quindi paghi solo per la capacità utilizzata.
- AWS ECS su AWS Fargate ti consente di pagare solo per le risorse di elaborazione che utilizzi, senza spese iniziali.
- AWS AgentCore Gateway funge da proxy economico basato su Lambda per l'API di test di carico distribuito, eliminando la necessità di un'infrastruttura dedicata e riducendo i costi grazie a prezzi serverless. pay-per-request

Sostenibilità

Questa sezione descrive come abbiamo progettato questa soluzione utilizzando i principi e le migliori pratiche del pilastro della sostenibilità.

- La soluzione utilizza servizi serverless gestiti per ridurre al minimo l'impatto ambientale dei servizi di backend rispetto ai servizi locali operativi in modo continuo.
- I servizi serverless consentono la scalabilità verso l'alto o verso il basso in base alle esigenze.

Dettagli dell'architettura

Questa sezione descrive i componenti e i servizi AWS che compongono questa soluzione e i dettagli dell'architettura su come questi componenti interagiscono.

La soluzione Distributed Load Testing on AWS è composta da tre componenti di alto livello: un front-end, un backend e un server MCP opzionale.

Front-end

Il front-end fornisce le interfacce per interagire con la soluzione e include:

- Un'API di test di carico per l'accesso programmatico
- Una console web per la creazione, la pianificazione e l'esecuzione di test delle prestazioni
- Un server MCP opzionale per l'analisi assistita dall'intelligenza artificiale dei risultati e degli errori dei test

API di test di carico

Distributed Load Testing su AWS configura Amazon API Gateway per ospitare l' RESTful API della soluzione. Gli utenti possono interagire con il sistema di test di carico in modo sicuro tramite la console web, l' RESTful API e il server MCP opzionale inclusi. L'API funge da «porta d'ingresso» per l'accesso ai dati di test archiviati in Amazon DynamoDB. Puoi anche utilizzare il APIs per accedere a qualsiasi funzionalità estesa incorporata nella soluzione.

Questa soluzione sfrutta le funzionalità di autenticazione degli utenti dei pool di utenti di Amazon Cognito. Dopo aver autenticato correttamente un utente, Amazon Cognito emette un token web JSON che viene utilizzato per consentire alla console di inviare richieste alla soluzione (endpoint Amazon APIs API Gateway). Le richieste HTTPS vengono inviate dalla console a APIs con l'intestazione di autorizzazione che include il token.

In base alla richiesta, API Gateway richiama la funzione AWS Lambda appropriata per eseguire le attività necessarie sui dati archiviati nelle tabelle DynamoDB, archiviare scenari di test come oggetti JSON in Amazon S3, recuperare immagini dei parametri Amazon e inviare scenari di test alla macchina a stati AWS CloudWatch Step Functions.

Per ulteriori informazioni sull'API della soluzione, consulta la sezione Distributed load testing API di questa guida.

Console Web

Questa soluzione include una console Web che è possibile utilizzare per configurare ed eseguire test, monitorare i test in esecuzione e visualizzare i risultati dettagliati dei test. La console è un'applicazione ReactJS creata con Cloudscape, un sistema di progettazione open source per la creazione di applicazioni web intuitive. La console è ospitata in Amazon S3 e vi si accede tramite Amazon. CloudFront L'applicazione sfrutta AWS Amplify per l'integrazione con Amazon Cognito per autenticare gli utenti. La console Web contiene anche un'opzione per visualizzare i dati in tempo reale per un test in esecuzione, in cui sottoscrive l'argomento corrispondente in AWS IoT Core.

L'URL della console Web è il nome del dominio di CloudFront distribuzione che può essere trovato negli CloudFormation output come Console. Dopo aver avviato il CloudFormation modello, riceverai anche un'e-mail contenente l'URL della console Web e la password monouso per accedervi.

Server MCP (opzionale)

Il server opzionale Model Context Protocol (MCP) fornisce un'interfaccia aggiuntiva per gli strumenti di sviluppo AI per accedere e analizzare i dati dei test di carico tramite interazioni in linguaggio naturale. Questo componente viene distribuito solo se si seleziona l'opzione MCP Server durante la distribuzione della soluzione.

Il server MCP consente agli agenti AI di interrogare i risultati dei test, analizzare le metriche delle prestazioni e ottenere informazioni sui dati dei test di carico utilizzando strumenti come Amazon Q, Claude e altri assistenti AI compatibili con MCP. Per informazioni dettagliate sull'architettura e la configurazione del server MCP, consulta MCP Server in questa sezione.

Backend

Il backend è costituito da una pipeline di immagini del contenitore e da un motore di test del carico utilizzato per generare il carico per i test. Interagisci con il backend tramite il front-end. Inoltre, le attività di Amazon ECS su AWS Fargate avviate per ogni test sono contrassegnate con un identificatore di test (ID) univoco. Questi tag ID di test possono essere utilizzati per aiutarti a monitorare i costi di questa soluzione. Per ulteriori informazioni, consulta i tag di allocazione dei costi definiti dall'utente nella AWS Billing and Cost Management User Guide.

Pipeline di immagini del contenitore

Questa soluzione utilizza un'immagine del contenitore creata con Amazon Linux 2023 come immagine di base con il framework di load testing Taurus installato. Taurus è un framework di

automazione dei test open source che supporta K6 JMeter, Locust e altri strumenti di test. AWS ospita questa immagine in un repository pubblico Amazon Elastic Container Registry (Amazon ECR). La soluzione utilizza questa immagine per eseguire attività nel cluster Amazon ECS on AWS Fargate.

Per ulteriori informazioni, consulta la sezione sulla personalizzazione dell'immagine del contenitore di questa guida.

Infrastruttura di test

Oltre al CloudFormation modello principale, la soluzione fornisce un modello regionale per avviare le risorse necessarie per eseguire i test in più regioni. La soluzione archivia questo modello in Amazon S3 e fornisce un collegamento ad esso nella console Web. Ogni stack regionale include un VPC, un cluster AWS Fargate e una funzione Lambda per l'elaborazione di dati in tempo reale.

Per ulteriori informazioni su come implementare l'infrastruttura di test in regioni aggiuntive, consulta la sezione Distribuzione multiregionale di questa guida.

Motore di test di carico

La soluzione Distributed Load Testing utilizza Amazon Elastic Container Service (Amazon ECS) e AWS Fargate per simulare migliaia di utenti simultanei in più regioni, generando richieste HTTP a una velocità sostenuta.

È possibile definire i parametri del test utilizzando la console Web inclusa. La soluzione utilizza questi parametri per generare uno scenario di test JSON e lo archivia in Amazon S3. Per ulteriori informazioni sugli script di test e sui parametri di test, consulta la sezione Tipi di test in questa sezione.

Una macchina a stati AWS Step Functions esegue e monitora le attività di Amazon ECS in un cluster AWS Fargate. La macchina a stati AWS Step Functions include una funzione AWS Lambda ecr-checker, una funzione AWS Lambda, una funzione AWS Lambda task-runner, una funzione task-status-checker AWS Lambda con cancellazione delle attività e una funzione AWS Lambda con analisi dei risultati. Per ulteriori informazioni sul flusso di lavoro, consulta la sezione Test workflow di questa guida. Per ulteriori informazioni sui risultati dei test, consulta la sezione Risultati dei test di questa guida. Per ulteriori informazioni sul flusso di lavoro per l'annullamento del test, consulta la sezione Flusso di lavoro per l'annullamento del test di questa guida.

Se si selezionano dati in tempo reale, la soluzione avvia una funzione real-time-data-publisher Lambda in ogni regione tramite CloudWatch i log che corrispondono alle attività Fargate in quella

regione. La soluzione elabora e pubblica quindi i dati su un argomento in AWS IoT Core all'interno della regione in cui è stato lanciato lo stack principale. Per ulteriori informazioni, consulta la sezione Dati in tempo reale di questa guida.

Server MCP

L'integrazione opzionale del server Model Context Protocol (MCP) consente agli agenti AI di accedere e analizzare in modo programmatico i dati dei test di carico attraverso interazioni in linguaggio naturale. Questo componente viene distribuito solo se si seleziona l'opzione MCP Server durante la distribuzione della soluzione.

Il server MCP funge da ponte tra gli strumenti di sviluppo AI e l'implementazione DLT, fornendo un'interfaccia standardizzata per l'analisi intelligente dei risultati dei test delle prestazioni. L'architettura integra diversi servizi AWS per creare un'interfaccia sicura e scalabile per le interazioni con gli agenti AI:

AgentCore Gateway AWS

AWS AgentCore Gateway è un servizio completamente gestito che fornisce hosting standardizzato e gestione dei protocolli per i server MCP. In questa soluzione, AgentCore Gateway funge da endpoint pubblico a cui gli agenti AI si connettono quando richiedono l'accesso ai dati dei test di carico.

Il servizio gestisce tutte le comunicazioni del protocollo MCP, tra cui l'individuazione degli strumenti, la convalida dei token di autenticazione e il routing delle richieste. AgentCore Gateway funziona come un servizio multi-tenant con protezioni di sicurezza integrate contro le minacce comuni agli endpoint pubblici, convalidando al contempo le firme e le attestazioni dei token Cognito per ogni richiesta.

Server DLT MCP Lambda

La funzione DLT MCP Server Lambda è un componente serverless personalizzato che elabora le richieste MCP degli agenti AI e le traduce in query relative alle risorse DLT.

Questa funzione Lambda funge da livello di intelligenza dell'integrazione MCP, recuperando i risultati dei test dalle tabelle DynamoDB, accedendo agli artefatti prestazionali archiviati nei bucket S3 e interrogando i log per informazioni dettagliate sull'esecuzione. CloudWatch La funzione Lambda implementa modelli di accesso in sola lettura e trasforma i dati DLT non elaborati in formati strutturati e compatibili con l'intelligenza artificiale che gli agenti possono facilmente interpretare e analizzare.

Integrazione dell'autenticazione

Il sistema di autenticazione sfrutta l'infrastruttura del pool di utenti Cognito esistente per mantenere controlli di accesso coerenti sia sulla console Web che sulle interfacce del server MCP.

Questa integrazione utilizza OAuth l'autenticazione basata su token 2.0. Gli utenti si autenticano una sola volta tramite la procedura di accesso a Cognito e ricevono token che funzionano sia per le interazioni dell'interfaccia utente che per l'accesso al server MCP. Il sistema mantiene gli stessi limiti di autorizzazione e gli stessi controlli di accesso dell'interfaccia web, garantendo che gli utenti possano accedere solo tramite agenti di intelligenza artificiale agli stessi dati dei test di carico a cui possono accedere tramite la console.

Servizi AWS in questa soluzione

I seguenti servizi AWS sono inclusi in questa soluzione:

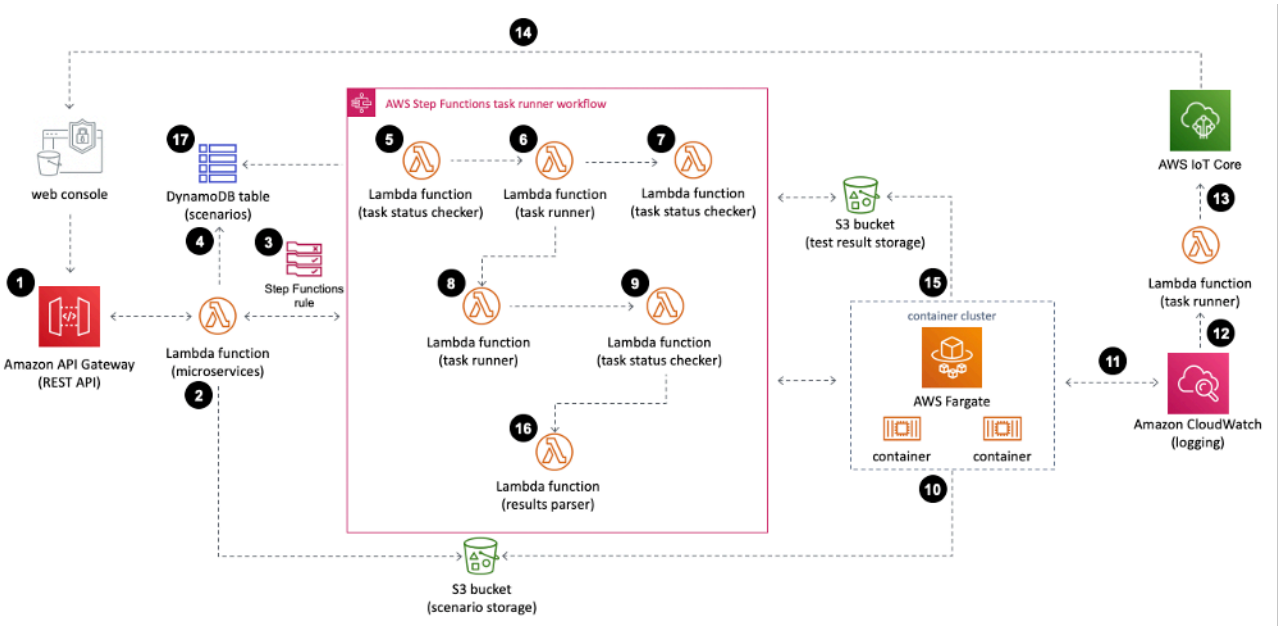
Servizio AWS	Description
Gateway Amazon API	Core. Ospita gli endpoint dell'API REST nella soluzione.
AWS CloudFormation	Nucleo. Gestisce le implementazioni per l'infrastruttura della soluzione.
Amazon CloudFront	Core. Serve i contenuti Web ospitati in Amazon S3.
Amazon CloudWatch	Nucleo. Memorizza i log e le metriche della soluzione.
Amazon Cognito	Nucleo. Gestisce la gestione e l'autenticazione degli utenti per l'API.
Amazon DynamoDB	Nucleo. Archivia le informazioni sulla distribuzione e i dettagli e i risultati dello scenario di test.
Amazon Elastic Container Service	Core. Distribuisce e gestisce attività Amazon ECS indipendenti su contenitori AWS Fargate.
AWS Fargate	Nucleo. Contenitori Amazon ECS della soluzione Hosts
AWS Identity and Access Management	Nucleo. Gestisce la gestione dei ruoli e delle autorizzazioni degli utenti.

Servizio AWS	Description
AWS Lambda	Nucleo. Fornisce la logica per APIs l'implementazione, l'analisi dei risultati dei test e l'avvio workers/leader delle attività.
AWS Step Functions	Nucleo. Orchestra il provisioning dei contenitori Amazon ECS sulle attività di AWS Fargate nelle regioni specificate
AWS Amplify	Supporto. Fornisce una console Web basata su AWS Amplify.
CloudWatch Eventi Amazon	Supporto. Pianifica l'inizio automatico dei test in una data specificata o in date ricorrenti.
Amazon Elastic Container Registry	Supporto. Ospita l'immagine del contenitore in un repository ECR pubblico.
AWS IoT Core	Supporto. Consente la visualizzazione di dati in tempo reale per un test in esecuzione sottoscrivendo l'argomento corrispondente in AWS IoT Core.
AWS Systems Manager	Supporto. Fornisce il monitoraggio delle risorse a livello di applicazione e la visualizzazione delle operazioni relative alle risorse e dei dati sui costi.
Amazon S3	Supporto. Ospita contenuti web statici, log, metriche e dati di test.
Amazon Virtual Private Cloud	Supporto. Contiene i contenitori Amazon ECS della soluzione in esecuzione su AWS Fargate.
Amazon Bedrock AgentCore	Supporto, opzionale. Ospita il server MCP (Remote Model Context Protocol) opzionale della soluzione per l'integrazione degli agenti AI con l'API.

Come funziona il test di carico distribuito su AWS

La seguente analisi dettagliata mostra i passaggi necessari per l'esecuzione di uno scenario di test.

Workflow di test



1. Si utilizza la console Web per inviare uno scenario di test che include i dettagli di configurazione all'API della soluzione.
2. La configurazione dello scenario di test viene caricata su Amazon Simple Storage Service (Amazon S3) come file JSON (). `s3://<bucket-name>/test-scenarios/<$TEST_ID>/<$TEST_ID>.json`
3. Una macchina a stati AWS Step Functions viene eseguita utilizzando l'ID di test, il conteggio delle attività, il tipo di test e il tipo di file come input della macchina a stati AWS Step Functions. Se il test è pianificato, creerà innanzitutto una regola CloudWatch Events, che attiva AWS Step Functions alla data specificata. Per maggiori dettagli sul flusso di lavoro di pianificazione, consulta la sezione Flusso di lavoro di pianificazione dei test di questa guida.
4. I dettagli di configurazione sono archiviati nella tabella degli scenari Amazon DynamoDB.
5. Nel flusso di lavoro del task runner di AWS Step Functions, la funzione task-status-checker AWS Lambda verifica se le attività di Amazon Elastic Container Service (Amazon ECS) sono già in esecuzione per lo stesso ID di test. Se vengono rilevate attività con lo stesso ID di test in esecuzione, viene generato un errore. Se non ci sono attività Amazon ECS in esecuzione nel cluster AWS Fargate, la funzione restituisce l'ID di test, il conteggio delle attività e il tipo di test.
6. La funzione task-runner AWS Lambda ottiene i dettagli delle attività dal passaggio precedente ed esegue le attività dei worker di Amazon ECS nel cluster AWS Fargate. L'API Amazon ECS utilizza l' RunTask azione per eseguire le attività dei lavoratori. Queste attività lavorative vengono avviate e quindi attendono un messaggio di avvio dall'attività principale per iniziare il test. L' RunTask azione è limitata a 10 attività per definizione. Se il numero di attività è superiore a 10, la definizione

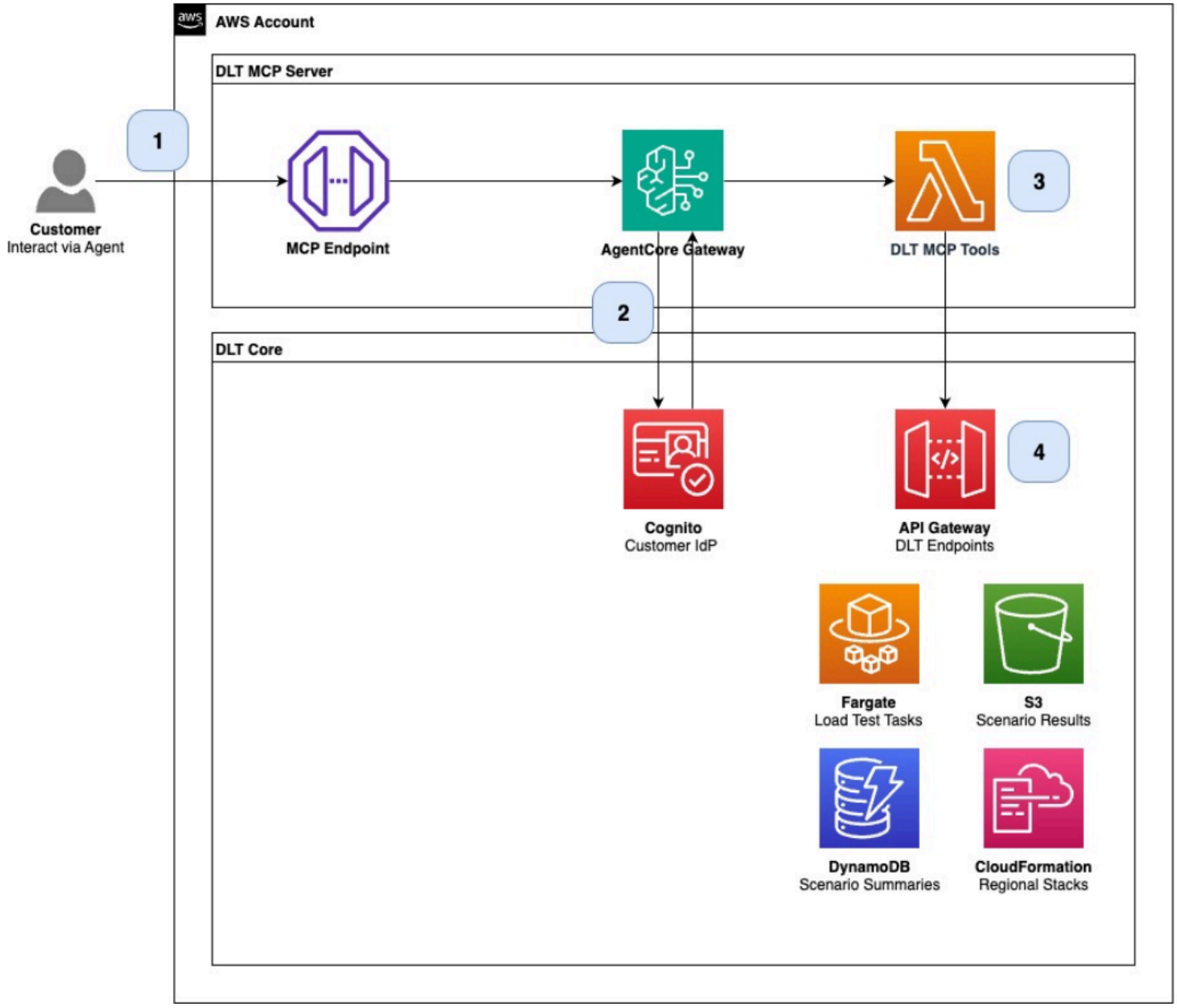
dell'attività verrà eseguita più volte fino all'avvio di tutte le attività dei lavoratori. La funzione genera anche un prefisso per distinguere il test corrente nella funzione di analisi dei risultati di AWS Lambda.

7. La funzione task-status-checker AWS Lambda verifica se tutte le attività di lavoro di Amazon ECS vengono eseguite con lo stesso ID di test. Se le attività sono ancora in fase di provisioning, attende un minuto e verifica nuovamente. Una volta eseguite tutte le attività di Amazon ECS, restituisce l'ID di test, il conteggio delle attività, il tipo di test, tutte le attività IDs e il prefisso e li passa alla funzione task-runner.
8. La funzione task-runner AWS Lambda viene eseguita nuovamente, questa volta lanciando una singola attività Amazon ECS che funge da nodo leader. Questa attività ECS invia un messaggio di avvio del test a ciascuna delle attività del lavoratore per avviare i test contemporaneamente.
9. La funzione task-status-checker AWS Lambda verifica nuovamente se le attività di Amazon ECS sono in esecuzione con lo stesso ID di test. Se le attività sono ancora in esecuzione, attende un minuto e verifica nuovamente. Quando non ci sono attività Amazon ECS in esecuzione, restituisce l'ID di test, il conteggio delle attività, il tipo di test e il prefisso.
10. Quando la funzione task-runner AWS Lambda esegue le attività di Amazon ECS nel cluster AWS Fargate, ogni attività scarica la configurazione di test da Amazon S3 e avvia il test.
11. Una volta eseguiti i test, il tempo di risposta medio, il numero di utenti simultanei, il numero di richieste riuscite e il numero di richieste non riuscite per ogni attività vengono registrati in Amazon CloudWatch e possono essere visualizzati in una CloudWatch dashboard.
12. Se hai incluso dati in tempo reale nel test, la soluzione filtra i risultati dei test in tempo reale CloudWatch utilizzando un filtro di abbonamento. Quindi la soluzione passa i dati a una funzione Lambda.
13. La funzione Lambda struttura quindi i dati ricevuti e li pubblica su un argomento di AWS IoT Core.
14. La console Web sottoscrive l'argomento AWS IoT Core per il test e riceve i dati pubblicati sull'argomento per rappresentare graficamente i dati in tempo reale durante l'esecuzione del test.
15. Al termine del test, le immagini del contenitore esportano un report dettagliato come file XML in Amazon S3. A ogni file viene assegnato un UUID per il nome del file. Ad esempio, s3://dlte-bucket/test-scenarios/ <$TEST_ID> /results/ <$UUID> .json.
16. Quando i file XML vengono caricati su Amazon S3, la funzione results-parser AWS Lambda legge i risultati nei file XML a partire dal prefisso e analizza e aggrega tutti i risultati in un unico risultato riepilogativo.
17. La funzione results-parser AWS Lambda scrive il risultato aggregato in una tabella Amazon DynamoDB.

Flusso di lavoro del server MCP (opzionale)

Se implementate l'integrazione opzionale con MCP Server, gli agenti AI possono accedere e analizzare i dati dei test di carico attraverso il seguente flusso di lavoro:

Architettura del server MCP



1. Interazione con il cliente: il cliente interagisce con l'MCP di DLT tramite l'endpoint MCP ospitato da AWS Gateway. AgentCore Gli agenti AI si connettono a questo endpoint per richiedere l'accesso ai dati dei test di carico.

2. Autorizzazione: AgentCore Gateway gestisce l'autorizzazione sul client dell'applicazione del pool di utenti di Solution Cognito. Il gateway convalida il token Cognito dell'utente per garantire che abbia l'autorizzazione ad accedere al server DLT MCP. Agli utenti autorizzati viene concesso l'accesso con accesso agli strumenti dell'agente limitato alle operazioni di sola lettura.
3. Specifiche dello strumento: il AgentCore gateway si connette alla funzione Lambda del server DLT MCP. Una specifica dello strumento definisce gli strumenti disponibili che gli agenti AI possono utilizzare per interagire con i dati dei test di carico.
4. Accesso API in sola lettura: la funzione Lambda è destinata all'accesso API in sola lettura tramite gli endpoint DLT API Gateway esistenti. La funzione fornisce quattro operazioni principali:
   - Elenca scenari: recupera un elenco di scenari di test dalla tabella degli scenari di DynamoDB
   - Ottieni i risultati dei test degli scenari: accedi ai risultati dettagliati dei test per scenari specifici di DynamoDB e S3
   - Get Fargate load test runner: richiedi informazioni sull'esecuzione delle attività Fargate nel cluster ECS
   - Ottieni stack regionali disponibili: recupera informazioni sull'infrastruttura regionale distribuita da CloudFormation

L'integrazione con MCP Server sfrutta l'infrastruttura DLT esistente (API Gateway, Cognito, DynamoDB, S3) per fornire un accesso sicuro e in sola lettura ai dati di test per analisi e approfondimenti basati sull'intelligenza artificiale.

Considerazioni di natura progettuale

Questa sezione descrive importanti decisioni di progettazione e opzioni di configurazione per la soluzione Distributed Load Testing on AWS, incluse le applicazioni supportate, i tipi di test, le opzioni di pianificazione e le considerazioni sulla distribuzione.

Applicazioni supportate

Questa soluzione supporta il test di applicazioni basate su cloud e applicazioni locali purché sia disponibile la connettività di rete dal tuo account AWS all'applicazione. La soluzione supporta l'utilizzo APIs di protocolli HTTP o HTTPS.

Tipi di test

Distributed Load Testing su AWS supporta diversi tipi di test: semplici test degli endpoint HTTP JMeter, K6 e Locust.

Semplici test degli endpoint HTTP

La console Web fornisce un'interfaccia di configurazione degli endpoint HTTP che consente di testare qualsiasi endpoint HTTP o HTTPS senza scrivere script personalizzati. È possibile definire l'URL dell'endpoint, selezionare il metodo HTTP (GET, POST, PUT, DELETE, ecc.) da un menu a discesa e, facoltativamente, aggiungere intestazioni di richiesta e payload body personalizzati. Questa configurazione consente di eseguire test APIs con token di autorizzazione personalizzati, tipi di contenuto o qualsiasi altra intestazione HTTP e corpo di richiesta richiesto dall'applicazione.

JMeter test

Quando si crea uno scenario di test utilizzando la console Web, è possibile caricare uno script JMeter di test. La soluzione carica lo script nel bucket S3 degli scenari. Quando le attività di Amazon ECS vengono eseguite, scaricano lo JMeter script da S3 ed eseguono il test.

> **Important**
>
> Sebbene JMeter lo script possa definire concorrenza (utenti virtuali), tassi di transazione (TPS), tempi di accelerazione e altri parametri di caricamento, la soluzione sostituirà queste configurazioni con i valori specificati nella schermata Traffic Shape durante la creazione del test. La configurazione Traffic Shape controlla il numero di attività, la concorrenza (utenti virtuali per attività), la durata dell'accelerazione e la durata di attesa per l'esecuzione del test.

Se disponi JMeter di file di input, puoi comprimere i file di input insieme allo script. JMeter È possibile scegliere il file zip quando si crea uno scenario di test.

Se desideri includere dei plugin, tutti i file.jar inclusi in una sottodirectory /plugins nel file zip fornito in bundle verranno copiati nella directory delle JMeter estensioni e saranno disponibili per il test di carico.

> **Note**
>
> Se JMeter includi file di input nel file di JMeter script, devi includere il percorso relativo dei file di input nel file di script. JMeter Inoltre, i file di input devono trovarsi nel percorso

relativo. Ad esempio, se i file JMeter di input e il file di script si trovano invece in/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, il test avrà esito negativo perché non sarà in grado di trovare i file di input.

Se includete JMeter dei plugin, i file.jar devono essere raggruppati in una sottodirectory denominata / plugins all'interno della radice del file zip. Rispetto alla radice del file zip, il percorso dei file jar deve essere. /plugins/bundled_plugin.jar.

Per ulteriori informazioni su come utilizzare gli script, consulta il Manuale dell'utente. JMeter JMeter

Test K6

La soluzione supporta i test basati sul framework K6. K6 è rilasciato con licenza AGPL-3.0. La soluzione visualizza un messaggio di conferma della licenza durante la creazione di un nuovo test K6. È possibile caricare il file di test K6 insieme a tutti i file di input necessari in un file di archivio.

Important

Sebbene lo script K6 possa definire concorrenza (utenti virtuali), fasi, soglie e altri parametri di carico, la soluzione sostituirà queste configurazioni con i valori specificati nella schermata Traffic Shape durante la creazione del test. La configurazione Traffic Shape controlla il numero di attività, la concorrenza (utenti virtuali per attività), la durata di avvio e la durata di attesa per l'esecuzione del test.

Test Locust

La soluzione supporta i test basati sul framework Locust. È possibile caricare il file di test Locust insieme a tutti i file di input necessari in un file di archivio.

Important

Sebbene lo script Locust possa definire la concorrenza (conteggio utenti), la frequenza di spawn e altri parametri di caricamento, la soluzione sostituirà queste configurazioni con i valori specificati nella schermata Traffic Shape durante la creazione del test. La

configurazione Traffic Shape controlla il numero di attività, la concorrenza (utenti virtuali per attività), la durata di avvio e la durata di attesa per l'esecuzione del test.

Pianificazione dei test

La soluzione offre tre opzioni di tempistica di esecuzione per l'esecuzione dei test di carico:

- Esegui ora: esegue il test di carico subito dopo la creazione
- Esegui una volta: esegui il test in una data e un'ora specifiche future
- Esegui secondo una pianificazione: crea test ricorrenti utilizzando le espressioni cron per definire la pianificazione

Quando si seleziona Esegui una volta, si specifica il tempo di esecuzione nel formato a 24 ore e la data di esecuzione in cui deve iniziare l'esecuzione del test di carico.

Quando selezioni Esegui su una pianificazione, puoi inserire manualmente un'espressione cron o selezionare uno dei modelli cron più comuni (ad esempio ogni ora, ogni giorno a un'ora specifica, nei giorni feriali o mensili). L'espressione cron utilizza un formato di pianificazione preciso con campi per minuti, ore, giorno del mese, mese, giorno della settimana e anno. È inoltre necessario specificare una data di scadenza, che definisce quando l'esecuzione del test programmato deve cessare. Per ulteriori informazioni su come funziona la pianificazione, consulta la sezione Flusso di lavoro di pianificazione dei test di questa guida.

> **Note**
>
> - Durata del test: durante la pianificazione, considera la durata totale dei test. Ad esempio, il completamento di un test con un tempo di accelerazione di 10 minuti e un tempo di attesa di 40 minuti richiederà circa 80 minuti.
> - Intervallo minimo: assicurati che l'intervallo tra i test programmati sia più lungo della durata stimata del test. Ad esempio, se il test dura circa 80 minuti, programmalo in modo che venga eseguito con una frequenza non superiore a ogni 3 ore.
> - Limitazione oraria: il sistema non consente di programmare i test con una differenza di solo un'ora, anche se la durata stimata del test è inferiore a un'ora.

Test simultanei

Questa soluzione crea una CloudWatch dashboard Amazon per ogni test che mostra l'output combinato di tutte le attività in esecuzione nel cluster Amazon ECS in tempo reale. La CloudWatch dashboard mostra il tempo di risposta medio, il numero di utenti simultanei, il numero di richieste riuscite e il numero di richieste non riuscite. La soluzione aggrega ogni metrica al secondo e aggiorna la dashboard ogni minuto.

Gestione degli utenti

Durante la configurazione iniziale, fornisci un nome utente e un indirizzo e-mail che Amazon Cognito utilizza per concederti l'accesso alla console web della soluzione. La console non fornisce l'amministrazione degli utenti. Per aggiungere altri utenti, devi utilizzare la console Amazon Cognito. Per ulteriori informazioni, consulta la sezione Gestione degli utenti nei pool di utenti nella Amazon Cognito Developer Guide.

Per la migrazione degli utenti esistenti ai pool di utenti di Amazon Cognito, consulta il blog di AWS Approaches for migrating users to Amazon Cognito user pool.

Implementazione regionale

Questa soluzione utilizza Amazon Cognito, disponibile solo in regioni AWS specifiche. Pertanto, è necessario distribuire questa soluzione in una regione in cui è disponibile Amazon Cognito. Per la disponibilità dei servizi più aggiornata per regione, consulta l'AWS Regional Services List.

Pianifica la tua implementazione

Questa sezione descrive costi, sicurezza, regioni supportate, quote e altre considerazioni da esaminare prima di distribuire la soluzione.

Costo

Sei responsabile del costo dei servizi AWS utilizzati durante l'esecuzione di questa soluzione. Il costo totale dipende dal numero di test di carico eseguiti, dalla durata di tali test e dalla quantità di dati generati. A partire da questa revisione, il costo stimato per l'esecuzione di questa soluzione con le impostazioni predefinite nella regione Stati Uniti orientali (Virginia settentrionale) è di circa 30,90 USD al mese.

La tabella seguente fornisce un esempio di ripartizione dei costi per l'implementazione di questa soluzione con i parametri predefiniti nella regione Stati Uniti orientali (Virginia settentrionale) per un mese.

Servizio AWS	Dimensioni	Costo [USD]
AWS Fargate	10 attività su richiesta (utilizzando due v CPUs e 4 GB di memoria) in esecuzione per 30 ore	29,62 USD
Amazon DynamoDB	1.000 unità con capacità di scrittura su richiesta 1.000 unità con capacità di lettura su richiesta	0,0015 USD
AWS Lambda	1.000 richieste Durata totale di 10 minuti	$1,25
AWS Step Functions	1.000 transizioni di stato	0,025 USD
Totale:		$30,90 al mese

Le risorse della soluzione sono contrassegnate con Key= e Value=SolutionId . SO0062 È possibile attivare la chiave del tag SolutionId seguendo la documentazione activating-tags. Una volta attivato il tag, puoi creare una regola per la categoria di costo seguendo la documentazione per creare le categorie di costo. È possibile visualizzare i costi sostenuti per la soluzione monitorando la console delle categorie di costi e selezionando il nome della categoria di costo.

Ti consigliamo di creare un budget tramite AWS Cost Explorer per gestire i costi. I prezzi sono soggetti a modifiche. Per tutti i dettagli, consulta la pagina web dei prezzi per ogni servizio AWS utilizzato in questa soluzione.

> **Note**
>
> La configurazione predefinita delle attività utilizza 2 v CPUs e 4 GB di memoria per attività. Se i test di carico non richiedono queste risorse, è possibile ridurle per ridurre i costi. Al contrario, è possibile aumentare le risorse per supportare una maggiore concorrenza per attività. Per ulteriori informazioni, consulta la sezione Aumentare le risorse del contenitore di questa guida.

> **Note**
>
> Questa soluzione offre la possibilità di includere dati in tempo reale durante l'esecuzione di un test. Questa funzionalità richiede una funzione AWS Lambda aggiuntiva e un argomento AWS IoT Core che comporta costi aggiuntivi.

Costi aggiuntivi del server MCP (opzionale)

La tabella seguente fornisce una ripartizione dei costi per l'integrazione di MCP Server con i prezzi nella regione Stati Uniti orientali (Virginia settentrionale) per un mese.

Componente di servizio	Dimensioni	Costo [USD]
AgentCore Gateway - Indicizzazione degli strumenti	10 strumenti × 0,02 USD per 100 strumenti	0,002$
AgentCore Gateway - API di ricerca	10.000 interazioni × 0,025 USD per 1.000	0,25$

Componente di servizio	Dimensioni	Costo [USD]
AgentCore Gateway - Richiamazioni API	50.000 invocazioni × 0,005 USD per 1.000	0,25$
Funzione AWS Lambda	Variabile in base all'utilizzo (carichi di lavoro tipici)	$5,00 - $20,00
Costo aggiuntivo totale stimato:		$5,50 - $20,50 al mese

I prezzi sono soggetti a modifiche. Per tutti i dettagli sui prezzi di AgentCore Gateway, consulta la sezione Prezzi di Amazon Bedrock (sezione AgentCore Gateway). Per i prezzi Lambda, consulta i prezzi di AWS Lambda.

Sicurezza

Quando crei sistemi sull'infrastruttura AWS, le responsabilità di sicurezza vengono condivise tra te e AWS. Questo modello di responsabilità condivisa riduce il carico operativo perché AWS gestisce, gestisce e controlla i componenti, tra cui il sistema operativo host, il livello di virtualizzazione e la sicurezza fisica delle strutture in cui operano i servizi. Per ulteriori informazioni sulla sicurezza di AWS, visita AWS Cloud Security.

Ruoli IAM

I ruoli di AWS Identity and Access Management (IAM) consentono ai clienti di assegnare policy e autorizzazioni di accesso granulari a servizi e utenti sul cloud AWS. Questa soluzione crea ruoli IAM che garantiscono l'accesso alle funzioni AWS Lambda della soluzione per creare risorse regionali.

Amazon CloudFront

Questa soluzione implementa un'interfaccia utente Web ospitata in un bucket Amazon S3, distribuito da Amazon. CloudFront Per contribuire a ridurre la latenza e migliorare la sicurezza, questa soluzione include una CloudFront distribuzione con un'identità di accesso di origine, ovvero un CloudFront utente che fornisce l'accesso pubblico ai contenuti del bucket del sito Web della soluzione. Per impostazione predefinita, la CloudFront distribuzione utilizza TLS 1.2 per applicare il più alto livello di protocollo di sicurezza. Per ulteriori informazioni, consulta la sezione Limitazione dell'accesso a un'origine Amazon S3 nella CloudFront Amazon Developer Guide.

CloudFront attiva ulteriori mitigazioni di sicurezza per aggiungere intestazioni di sicurezza HTTP a ciascuna risposta del visualizzatore. Per ulteriori informazioni, consulta Aggiungere o rimuovere intestazioni HTTP nelle risposte. CloudFront

Questa soluzione utilizza il CloudFront certificato predefinito, che ha un protocollo di sicurezza minimo supportato di TLS v1.0. Per imporre l'uso di TLS v1.2 o TLS v1.3, è necessario utilizzare un certificato SSL personalizzato anziché il certificato predefinito. CloudFront Per ulteriori informazioni, consulta Come posso configurare la mia CloudFront distribuzione per utilizzare un certificato. SSL/TLS

Gateway Amazon API

Questa soluzione implementa endpoint Amazon API Gateway ottimizzati per i dispositivi perimetrali RESTful APIs per fornire la funzionalità di test di carico utilizzando l'endpoint API Gateway predefinito anziché un dominio personalizzato. Per l'ottimizzazione dell'edge APIs utilizzando l'endpoint predefinito, API Gateway utilizza la politica di sicurezza TLS-1-0. Per ulteriori informazioni, consulta Working with REST APIs nella Amazon API Gateway Developer Guide.

Questa soluzione utilizza il certificato API Gateway predefinito, che ha un protocollo di sicurezza minimo supportato di TLS v1.0. Per imporre l'uso di TLS v1.2 o TLS v1.3, è necessario utilizzare un dominio personalizzato con un certificato SSL personalizzato anziché il certificato API Gateway predefinito. Per ulteriori informazioni, consulta Configurazione di nomi di dominio personalizzati per REST. APIs

Gruppo di sicurezza AWS Fargate

Per impostazione predefinita, questa soluzione apre al pubblico la regola in uscita del gruppo di sicurezza AWS Fargate. Se desideri impedire ad AWS Fargate di inviare traffico ovunque, modifica la regola in uscita con uno specifico Classless Inter-Domain Routing (CIDR).

Questo gruppo di sicurezza include anche una regola in entrata che consente il traffico locale sulla porta 50.000 verso qualsiasi fonte appartenente allo stesso gruppo di sicurezza. Viene utilizzato per consentire ai contenitori di comunicare tra loro.

Amazon VPC

VPC: un cloud privato virtuale (VPC) basato sul servizio Amazon VPC offre una rete privata e logicamente isolata nel cloud AWS.

Puoi specificare il tuo VPC nei CloudFormation parametri AWS durante la distribuzione. Il VPC viene utilizzato esclusivamente dalle attività ECS che generano carico; la console Web e l'API non vengono distribuite all'interno di questo VPC. Se non si specifica un VPC esistente, la soluzione creerà un nuovo VPC con la configurazione di rete richiesta. Se scegli di utilizzare un VPC esistente, questo deve soddisfare i seguenti requisiti per eseguire correttamente le attività di test di carico.

Requisiti VPC

I requisiti minimi per un VPC da utilizzare con Distributed Load Testing on AWS sono elencati di seguito.

- Il VPC deve contenere almeno due AZs
- Il VPC deve contenere almeno due sottoreti, ciascuna in una AZ separata
- Le sottoreti VPC possono essere pubbliche o private, ma devono utilizzare la stessa configurazione (sia pubblica che privata)
- Il VPC deve fornire l'accesso agli endpoint per ECR, CloudWatch Logs, S3 e IoT Core.
- Il VPC deve fornire l'accesso ai servizi oggetto dei test di carico.

> **Note**
>
> Se non disponi di un VPC che soddisfi questi criteri, puoi creare rapidamente un VPC con la procedura guidata VPC. Per ulteriori informazioni, consulta la sezione Creazione di un VPC.

Le sottoreti pubbliche possono soddisfare questi requisiti includendo quanto segue:

- Un gateway Internet collegato al VPC
- Un percorso verso il gateway Internet (0.0.0.0/0)

Le sottoreti private possono soddisfare questi requisiti tramite l'uso di gateway NAT o endpoint VPC, come descritto di seguito.

Opzione 1: NAT Gateway

- Implementa un gateway NAT in ogni AZ con sottoreti private
- Configura le tabelle di routing per instradare il traffico diretto a Internet (0.0.0.0/0) attraverso il NAT Gateway

Opzione 2: endpoint VPC

Crea i seguenti endpoint VPC nel tuo VPC:

- Endpoint dell'API Amazon ECR: `com.amazonaws.<region>.ecr.api`
- Endpoint Amazon ECR DKR: `com.amazonaws.<region>.ecr.dkr`
- Endpoint Amazon CloudWatch Logs: `com.amazonaws.<region>.logs`
- Endpoint Amazon S3 Gateway: `com.amazonaws.<region>.s3`
- Endpoint AWS IoT Core (richiesto se si utilizzano i grafici di dati in tempo reale) `com.amazonaws.<region>.iot.data`

Potrebbero funzionare anche altre configurazioni VPC.

> **Important**
>
> Il gruppo di sicurezza collegato a ciascuna interfaccia endpoint VPC deve consentire il traffico TCP in entrata sulla porta 443 dal gruppo di sicurezza delle attività ECS.

Configurazione del gruppo di sicurezza

Durante l'implementazione, la soluzione creerà un gruppo di sicurezza all'interno del VPC per consentire il seguente traffico con attività nel cluster ECS:

- Tutto il traffico in uscita
- Traffico in entrata sulla porta 50000 proveniente da altre attività dello stesso gruppo di sicurezza, per facilitare il coordinamento tra le attività dei dipendenti e quelle dei dirigenti.

Test di stress della rete

L'utente è responsabile dell'utilizzo di questa soluzione in base alla politica di Network Stress Test. Questa policy copre situazioni come quando prevedi di eseguire test di rete ad alto volume direttamente dalle tue istanze Amazon EC2 su altre posizioni come altre istanze Amazon EC2, proprietà/servizi AWS o endpoint esterni. Questi test sono talvolta chiamati stress test, test di carico o test del giorno di gioco. La maggior parte dei test effettuati dai clienti non rientra in questa politica; tuttavia, fai riferimento a questa politica se ritieni che genererai traffico che durerà, in aggregato, per

più di 1 minuto, oltre 1 Gbps (1 miliardo di bit al secondo) o oltre 1 Gpps (1 miliardo di pacchetti al secondo).

Limitazione dell'accesso all'interfaccia utente pubblica

Per limitare l'accesso all'interfaccia utente pubblica oltre ai meccanismi di autenticazione e autorizzazione forniti da IAM e Amazon Cognito, utilizza la soluzione AWS WAF (web application firewall) Security Automations.

Questa soluzione implementa automaticamente una serie di regole AWS WAF che filtrano i comuni attacchi basati sul Web. Gli utenti possono scegliere tra funzionalità di protezione preconfigurate che definiscono le regole incluse in una lista di controllo degli accessi Web AWS WAF (Web ACL).

Sicurezza del server MCP (opzionale)

Se distribuisci l'integrazione opzionale con MCP Server, la soluzione utilizza AWS AgentCore Gateway per fornire un accesso sicuro ai dati di test di carico per gli agenti AI. AgentCore Gateway convalida i token di autenticazione Amazon Cognito per ogni richiesta, garantendo che solo gli utenti autorizzati possano accedere al server MCP. La funzione MCP Server Lambda implementa modelli di accesso di sola lettura, impedendo agli agenti AI di modificare le configurazioni o i risultati dei test. Tutte le interazioni con il server MCP utilizzano gli stessi limiti di autorizzazione e gli stessi controlli di accesso della console web.

Regioni AWS supportate

Questa soluzione utilizza il servizio Amazon Cognito, che attualmente non è disponibile in tutte le regioni AWS. Per la disponibilità più aggiornata dei servizi AWS per regione, consulta l'AWS Regional Services List.

Il test di carico distribuito su AWS è disponibile nelle seguenti regioni AWS:

Nome della Regione	
Stati Uniti orientali (Ohio)	Asia Pacifico (Tokyo)
Stati Uniti orientali (Virginia settentrionale)	Canada (Centrale)
Stati Uniti occidentali (California settentrionale)	Europa (Francoforte)

Nome della Regione	
Stati Uniti occidentali (Oregon)	Europa (Irlanda)
Asia Pacifico (Mumbai)	Europa (Londra)
Asia Pacifico (Seul)	Europa (Parigi)
Asia Pacifico (Singapore)	Europa (Stoccolma)
Asia Pacifico (Sydney)	Sud America (San Paolo)

Regioni AWS supportate da MCP Server (opzionale)

Se prevedi di implementare l'integrazione opzionale con MCP Server, devi distribuire la soluzione in una regione AWS in cui è disponibile AWS AgentCore Gateway. La funzionalità MCP Server è disponibile solo nelle seguenti regioni AWS:

Nome della Regione	Codice regione
Stati Uniti orientali (Virginia settentrionale)	us-east-1
Stati Uniti occidentali (Oregon)	us-west-2
Asia Pacifico (Singapore)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Europa (Francoforte)	eu-central-1
Europe (Ireland)	eu-west-1
Europe (London)	eu-west-2
Europe (Paris)	eu-west-3

Per conoscere la disponibilità più aggiornata di AWS AgentCore Gateway per regione, consulta gli endpoint e le quote di AWS AgentCore Gateway nella AWS AgentCore Gateway Developer Guide.

Quote

Le quote di servizio, anche denominate limiti, rappresentano il numero massimo di risorse di servizio o operazioni per l'account AWS.

Quote per i servizi AWS in questa soluzione

Assicurati di disporre di una quota sufficiente per ciascuno dei servizi implementati in questa soluzione. Per ulteriori informazioni, consulta Quote di servizio AWS.

Utilizza i seguenti collegamenti per accedere alla pagina relativa al servizio. Per visualizzare le quote di servizio per tutti i servizi AWS nella documentazione senza cambiare pagina, visualizza invece le informazioni nella pagina Endpoint e quote del servizio nel PDF.

CloudFormation Quote AWS

Il tuo account AWS ha CloudFormation quote AWS di cui dovresti essere a conoscenza quando avvii lo stack di questa soluzione. Comprendendo queste quote, puoi evitare errori di limitazione che potrebbero impedirti di implementare questa soluzione con successo. Per ulteriori informazioni, consulta le CloudFormation quote AWS nella AWS CloudFormation User's Guide.

Quote di test di carico

Il numero massimo di attività che possono essere eseguite in Amazon ECS utilizzando il tipo di avvio AWS Fargate si basa sulla dimensione della vCPU delle attività. La dimensione predefinita dell'attività in Distributed Load Testing on AWS è di 2 vCPU. Per visualizzare le quote predefinite correnti, consulta le quote dei servizi Amazon ECS. Le quote dei conti correnti possono differire dalle quote elencate. Per controllare le quote specifiche di un account, controlla la quota di servizio per il conteggio delle risorse vCPU on-demand di Fargate nella Console di gestione AWS. Per istruzioni su come richiedere un aumento, consulta le quote dei servizi AWS nella AWS General Reference Guide.

L'immagine del contenitore Amazon Linux 2023 (con Taurus installato) non limita le connessioni simultanee per attività, ma ciò non significa che possa supportare un numero illimitato di utenti. Per determinare il numero di utenti simultanei che i contenitori possono generare per un test, consulta la sezione Determinare il numero di utenti di questa guida.

> **Note**
>
> Il limite consigliato per gli utenti simultanei in base alle impostazioni predefinite è di 200 utenti.

Test simultanei

Questa soluzione crea una CloudWatch dashboard Amazon per ogni test che mostra l'output combinato di tutte le attività in esecuzione nel cluster Amazon ECS in tempo reale. La CloudWatch dashboard mostra il tempo di risposta medio, il numero di utenti simultanei, il numero di richieste riuscite e il numero di richieste non riuscite. La soluzione aggrega ogni metrica al secondo e aggiorna la dashboard ogni minuto.

Politica di test di Amazon EC2

Non è necessaria l'approvazione di AWS per eseguire test di carico utilizzando questa soluzione purché il traffico di rete rimanga inferiore a 1 Gbps. Se il test genererà più di 1 Gbps, contatta AWS. Per ulteriori informazioni, consulta la policy di test di Amazon EC2.

Politica di test CloudFront di carico di Amazon

Se prevedi di testare il carico di un CloudFront endpoint, consulta le linee guida per i test di carico nell'Amazon CloudFront Developer Guide. Consigliamo inoltre di distribuire il traffico tra più attività e regioni. Fornisci almeno 30 minuti di accelerazione per il test di carico. Per i test di carico che inviano più di 500.000 richieste al secondo o richiedono dati superiori a 300 Gbps, consigliamo di ottenere prima un'approvazione preventiva per l'invio del traffico. CloudFront può limitare il traffico di test di carico non approvato che influisce sulla disponibilità del servizio. CloudFront

Monitoraggio della soluzione dopo l'implementazione

Dopo aver distribuito la soluzione, consigliamo di monitorare continuamente le risorse della soluzione utilizzando CloudWatch allarmi e metriche di Amazon.

Configurazione degli allarmi CloudWatch

Puoi configurare CloudWatch allarmi per monitorare le metriche chiave e ricevere notifiche quando vengono superate le soglie. Valuta la possibilità di configurare allarmi per le seguenti risorse:

Metriche CloudFront di distribuzione di Amazon

Monitora le prestazioni e gli errori di CloudFront distribuzione. Per ulteriori informazioni, consulta i parametri CloudFront di distribuzione nell'Amazon CloudFront Developer Guide.

Parametri per Gateway Amazon API

Monitora i tassi di richiesta, la latenza e gli errori delle API. Per ulteriori informazioni, consulta le dimensioni e i parametri di Amazon API Gateway nella Amazon API Gateway Developer Guide.

Metriche delle funzioni AWS Lambda

Monitora le chiamate, la durata, gli errori e le accelerazioni delle funzioni Lambda per i microservizi della soluzione.

Metriche di Amazon ECS e AWS Fargate

Monitora l'utilizzo della CPU e della memoria delle attività durante i test di carico per garantire risorse adeguate.

Parametri di Amazon DynamoDB

Monitora il consumo di capacità di lettura e scrittura, le richieste limitate e la latenza.

Rivolgiti a un esperto

Impegni a breve termine AWS Countdown Premium per test di carico distribuiti su AWS

I nostri ingegneri AWS forniscono una guida esperta sui fondamenti dei test delle prestazioni, sullo sviluppo di script e sull'analisi dei risultati. Registrati ora.

Panoramica

AWS Countdown Premium (CDP) Short Term Engagements fornisce una guida esperta per le organizzazioni che effettuano test delle prestazioni su larga scala. Attraverso un modello "do-it-yourself" collaborativo, gli ingegneri AWS offrono supervisione strategica e competenze tecniche mentre il team mantiene la responsabilità dell'esecuzione. I tecnici esperti di AWS sono disponibili entro una settimana dall'iscrizione senza bisogno di contratti a lungo termine.

Modello di servizio

Gli ingegneri CDP collaborano con il tuo team per fornire indicazioni e supervisione durante l'implementazione dei test delle prestazioni. Questo approccio pratico ti assicura di ricevere la guida di un esperto mentre sviluppi le capacità interne. Il servizio è ideale per le organizzazioni con capacità di test esistenti che necessitano di competenze AWS specializzate per implementare Distributed Load Testing su AWS in modo efficace.

Cosa forniscono gli ingegneri CDP

Gli ingegneri CDP ti guidano attraverso i fondamentali dei test delle prestazioni e il Distributed Load Testing sull'architettura AWS. Forniscono indicazioni sulla JMeter struttura degli script K6 e Locust e sullo sviluppo degli script di test, forniscono assistenza nella distribuzione dei CloudFormation modelli e valutano i risultati dei test con consigli per l'ottimizzazione delle prestazioni. Il supporto include l'analisi dell'utilizzo delle risorse, l'allineamento delle best practice e end-to-end la guida dalla configurazione iniziale all'analisi dei risultati, per consentire il trasferimento delle conoscenze al team.

Responsabilità del cliente

Il tuo team gestisce le configurazioni a livello di applicazione, lo sviluppo di script di test e la verifica degli scenari di test. Sei responsabile dell'esecuzione e delle operazioni effettive dei test, comprese tutte le attività di test prima, durante e dopo gli eventi di test delle prestazioni.

Vantaggi principali

I CDP Short Term Engagements riducono i rischi grazie alla supervisione di esperti, alla guida contestuale specifica per il carico di lavoro, ai consigli per l'ottimizzazione delle prestazioni, una risoluzione più rapida dei problemi, l'allineamento delle best practice e un supporto completo, mantenendo al contempo la titolarità e lo sviluppo delle capacità del team.

Architetture supportate

Distributed Load Testing on AWS supporta test per applicazioni Web APIs, microservizi e architetture serverless su larga scala, sfruttando la soluzione Distributed Load Testing on AWS. Le funzionalità di test vanno ben oltre questi casi d'uso comuni e includono database, TCP/UDP protocolli, directory LDAP, server di posta SMTP e molti altri sistemi e protocolli che richiedono la convalida delle prestazioni sotto carico.

Nozioni di base

Organizations interessate a CDP Short Term Engagements for Distributed Load Testing on AWS possono registrarsi direttamente tramite il sito Web di AWS qui e selezionare «Use Case Implementation» come area di interesse.

Fuori ambito

CDP non fornisce lo sviluppo di script di test personalizzati (solo linee guida), gestisce le operazioni di esecuzione dei test né crea laboratori o workshop pratici personalizzati. Anche il supporto in loco è escluso.

Implementazione della soluzione

AWS Launch Wizard è il metodo di distribuzione consigliato per questa soluzione. Offre:

- Un'esperienza di configurazione guidata con pannelli di aiuto dettagliati in ogni fase
- Una pagina centralizzata per monitorare lo stato di tutte le implementazioni
- Indicazione quando è disponibile una versione più recente della soluzione per la distribuzione o l'aggiornamento

In alternativa, puoi distribuire la soluzione direttamente utilizzando un CloudFormation modello AWS.

Panoramica del processo di distribuzione

Prima di implementare la soluzione, esamina i costi, l'architettura, la sicurezza e altre considerazioni discusse in precedenza in questa guida.

Tempo di implementazione: circa 15 minuti per lo stack principale, più 5 minuti per ogni regione aggiuntiva

> **Note**
>
> Questa soluzione include parametri di raccolta dati per AWS. Utilizziamo questi dati per comprendere meglio come i clienti utilizzano questa soluzione e i servizi e i prodotti correlati. AWS possiede i dati raccolti attraverso questo sondaggio. La raccolta dei dati è soggetta all'Informativa sulla privacy di AWS.

> **Note**
>
> Sei responsabile del costo dei servizi AWS utilizzati durante l'esecuzione di questa soluzione. Per ulteriori dettagli, visita la sezione Costo di questa guida e consulta la pagina web dei prezzi per ogni servizio AWS utilizzato in questa soluzione.

Implementa con AWS Launch Wizard

Questa soluzione prevede un processo di distribuzione guidato utilizzando AWS Launch Wizard. Segui questi passaggi per distribuire Distributed Load Testing on AWS nel tuo account.

1. Accedi alla Console di gestione AWS e seleziona il pulsante in basso per avviare il processo di distribuzione.

2. Se sono disponibili più modelli di distribuzione per la soluzione, seleziona quello più adatto al tuo caso d'uso.
3. Seleziona una versione da distribuire. È consigliata la versione più recente.
4. Fai clic sul pulsante Avvia la procedura guidata di distribuzione.

Seguirai quindi una serie di passaggi per raccogliere le informazioni necessarie per implementare la soluzione. Saranno necessari circa 15 minuti per fornire le risorse necessarie.

Seleziona la tua distribuzione dall'elenco delle distribuzioni per visualizzarne lo stato.

Implementa con AWS CloudFormation

Questa soluzione utilizza CloudFormation modelli e stack AWS per automatizzarne l'implementazione. I CloudFormation modelli specificano le risorse AWS incluse in questa soluzione e le relative proprietà. Lo CloudFormation stack fornisce le risorse descritte nei modelli.

CloudFormation Modello AWS

Puoi scaricare il CloudFormation modello per questa soluzione prima di distribuirla. Questa soluzione utilizza AWS CloudFormation per automatizzare l'implementazione di Distributed Load Testing su AWS. Include il seguente CloudFormation modello AWS, che puoi scaricare prima della distribuzione:

distribu

load-testing-on-aws.template: utilizza questo modello per avviare la soluzione e tutti i componenti associati. La configurazione predefinita distribuisce i servizi di base e di supporto disponibili nei

servizi AWS in questa sezione della soluzione, ma puoi personalizzare il modello per soddisfare le tue esigenze specifiche.

> **Note**
>
> Le CloudFormation risorse AWS vengono create a partire da costrutti di AWS Cloud Development Kit (AWS CDK). Se hai già distribuito questa soluzione, consulta Aggiornare la soluzione per le istruzioni di aggiornamento.

Avvio dello stack

Segui questi passaggi per distribuire la soluzione Distributed Load Testing on AWS nel tuo account. Questo CloudFormation modello AWS automatizzato implementa Distributed Load Testing su AWS.

1. Accedi alla Console di gestione AWS e seleziona il pulsante per avviare il CloudFormation modello.

   In alternativa, puoi scaricare il modello come punto di partenza per la tua implementazione.
2. Questo modello viene avviato nella regione Stati Uniti orientali (Virginia settentrionale), per impostazione predefinita. Per avviare questa soluzione in un'altra regione AWS, utilizza il selettore di regione nella barra di navigazione della console.

   > **Note**
   >
   > Questa soluzione utilizza Amazon Cognito, attualmente disponibile solo in regioni AWS specifiche. Pertanto, è necessario avviare questa soluzione in una regione AWS in cui è disponibile Amazon Cognito. Per la disponibilità dei servizi più aggiornata per regione, consulta l'AWS Regional Services List.
3. Nella pagina Create stack, verifica che l'URL del modello corretto sia visualizzato nella casella di testo URL Amazon S3 e scegli Avanti.
4. Nella pagina Specificare i dettagli dello stack, assegna un nome allo stack di soluzioni.
5. In Parametri, esaminate i parametri per il modello e modificateli se necessario. Questa soluzione utilizza i seguenti valori predefiniti.

Parametro	Predefinita	Description
Nome dell'amministratore	<Da configurare da parte dell'utente>	Nome utente per l'amministratore della soluzione iniziale.
Email dell'amministratore	*<Requires input>*	Indirizzo e-mail dell'utente amministratore. Dopo il lancio, verrà inviata un'e-mail a questo indirizzo con le istruzioni per l'accesso alla console.
ID VPC esistente	<Optional input>	Se hai un VPC che desideri utilizzare ed è già stato creato, inserisci l'ID di un VPC esistente nella stessa regione in cui è stato distribuito lo stack. Ad esempio, vpc-1a2b3c4d5e6f.
Prima sottorete esistente	<Optional input>	L'ID della prima sottorete all'interno del VPC esistente. Questa sottorete necessita di un percorso verso Internet per recuperare l'immagine del contenitore per eseguire i test. Ad esempio, subnet-7h8i9j0k.

Parametro	Predefinita	Description
Seconda sottorete esistente	<Optional input>	L'ID della seconda sottorete all'interno del VPC esistente . Questa sottorete necessita di un percorso verso Internet per recuperare l'immagine del contenitore per eseguire i test. Ad esempio, subnet-1x 2y3z.
Fornisci un blocco CIDR valido per la soluzione per creare VPC	192.168.0.0/16	È possibile lasciare vuoto questo parametro se si utilizza un VPC esistente
Fornisci un blocco CIDR valido per la sottorete A per la soluzione per creare VPC	192.168.0.0/20	Blocco CIDR per la sottorete A del VPC AWS Fargate
Fornisci un blocco CIDR valido per la sottorete B per la soluzione per creare VPC	192,168,160/20	Blocco CIDR per la sottorete B del VPC AWS Fargate
Fornisci il blocco CIDR per consentire il traffico in uscita delle attività di Fargate	0.0.0.0/0	Blocco CIDR che limita l'accesso in uscita ai container Amazon ECS.
Aggiornamento automatico dell'immagine del contenitore	No	Usa automaticamente l'immagine più aggiornata e sicura fino alla prossima versione secondaria. La selezione No riporterà l'immagine così come è stata rilasciata originariamente, senza aggiornamenti di sicurezza.

Parametro	Predefinita	Description
Implementa un server MCP opzionale	No	Implementa il server MCP remoto opzionale, utilizzando AgentCore Gateway per connettere le applicazioni AI a Distributed Load Testing su AWS.

6. Scegli Next (Successivo).
7. Nella pagina Configure stack options (Configura opzioni pila), scegliere Next (Successivo).
8. Nella pagina Rivedi, verifica e conferma le impostazioni. Seleziona la casella per confermare che il modello creerà risorse AWS Identity and Access Management (IAM).
9. Seleziona Create (Crea) per implementare lo stack.

Puoi visualizzare lo stato dello stack nella CloudFormation console AWS nella colonna Status. Dovresti ricevere lo status CREATE_COMPLETE in circa 15 minuti.

> **Note**
>
> Oltre alla funzione principale di AWS Lambda, questa soluzione include la funzione Lambda con risorse personalizzate, che viene eseguita solo durante la configurazione iniziale o quando le risorse vengono aggiornate o eliminate.

Quando si esegue questa soluzione, la funzione Lambda della risorsa personalizzata è inattiva. Tuttavia, non eliminate questa funzione poiché è necessaria per gestire le risorse associate.

Implementazione in più regioni

Tempo di implementazione: circa 5 minuti per regione

È possibile eseguire test in più regioni.

Quando si implementa la soluzione Distributed Load Testing, viene creato un CloudFormation modello regionale nel bucket S3 degli scenari. L'URL di questo modello è elencato negli CloudFormation output dello stack principale sotto la chiave «Regionale». CFTemplate

Per eseguire un test multiregionale, è necessario distribuire il CloudFormation modello regionale in ogni regione in cui si desidera eseguire il test.

> **Note**
>
> Ogni account AWS può utilizzare solo uno stack regionale per regione. Inoltre, lo stack regionale non può essere utilizzato nella stessa regione dello stack principale.

È possibile installare il modello regionale come segue:

1. Nella console web della soluzione, accedi a Dashboard nel menu a sinistra.
2. Usa l'icona degli appunti per copiare il link del CloudFormation modello in Amazon S3.
3. Accedi alla CloudFormation console AWS e seleziona la regione corretta.
4. Nella pagina Create stack, verifica che l'URL del modello corretto sia visualizzato nella casella di testo URL Amazon S3 e scegli Avanti.
5. Nella pagina Specificare i dettagli dello stack, assegna un nome allo stack di soluzioni.
6. In Parametri, esaminate i parametri per il modello e modificateli se necessario. Questa soluzione utilizza i seguenti valori predefiniti.

Parametro	Predefinita	Description
ID VPC esistente	<Optional input>	Se hai un VPC che desideri utilizzare ed è già stato creato, inserisci l'ID di un VPC esistente nella stessa regione in cui è stato distribui to lo stack. Ad esempio, vpc-1a2b3c4d5e6f.
Prima sottorete esistente	<Optional input>	L'ID della prima sottorete all'interno del VPC esistente . Questa sottorete necessita di un percorso verso Internet per recuperare l'immagine del contenitore per eseguire i

Parametro	Predefinita	Description
		test. Ad esempio, subnet-7h 8i9j0k.
Seconda sottorete esistente	<Optional input>	L'ID della seconda sottorete all'interno del VPC esistente . Questa sottorete necessita di un percorso verso Internet per recuperare l'immagine del contenitore per eseguire i test. Ad esempio, subnet-1x 2y3z.
Fornisci un blocco CIDR valido per la soluzione per creare VPC	192.168.0.0/16	Se non fornisci valori per un VPC esistente, il blocco CIDR per Amazon VPC creato dalla soluzione contiene l'indirizzo IP per AWS Fargate.
Fornisci il blocco CIDR per consentire il traffico in uscita delle attività di Fargate	0.0.0.0/0	Blocco CIDR che limita l'accesso in uscita ai container Amazon ECS.

7. Scegli Next (Successivo).
8. Nella pagina Configure stack options (Configura opzioni pila), scegliere Next (Successivo).
9. Nella pagina Rivedi, verifica e conferma le impostazioni. Assicurati di selezionare la casella per confermare che il modello creerà risorse AWS Identity and Access Management (IAM).
10. Seleziona Create (Crea) per implementare lo stack.

    Puoi visualizzare lo stato dello stack nella CloudFormation console AWS nella colonna Status. Dovresti ricevere lo status CREATE_COMPLETE in circa cinque minuti.

Quando le regioni sono state distribuite correttamente, vengono visualizzate nella console Web. Quando si crea un test, tutte le regioni disponibili vengono elencate nella dashboard e in Creazione dello scenario. Puoi aggiungere una regione a un test nella fase Traffic Shape della creazione dello scenario.

La soluzione crea un elemento DynamoDB per ogni regione distribuita nella tabella degli scenari, che contiene le informazioni necessarie sulle risorse di test in quella regione. È possibile ordinare i risultati dei test nella console Web per regione. Per visualizzare i risultati aggregati in tutte le regioni in un test multiregionale, utilizza i parametri di Amazon CloudWatch . Puoi trovare il codice sorgente del grafico nei risultati del test dopo il completamento del test.

> **Note**
>
> Puoi avviare lo stack regionale senza la console web. Ottieni un link al modello regionale nel bucket di scenari Amazon S3 e forniscilo come origine quando avvii lo stack regionale nella regione richiesta. In alternativa, puoi scaricare il modello e caricarlo come origine per la regione che desideri.

Aggiorna la soluzione

L'aggiornamento della soluzione applica le funzionalità, le patch di sicurezza e le correzioni di bug più recenti alla distribuzione. Per eseguire l'aggiornamento alla versione più recente, consulta la sezione appropriata in base al metodo di distribuzione originale: AWS Launch Wizard o AWS. CloudFormation

> **Important**
>
> Prima dell'aggiornamento, assicurati che non siano attualmente in esecuzione test di carico. Il processo di aggiornamento potrebbe compromettere temporaneamente la disponibilità della soluzione.

Aggiornamento tramite AWS Launch Wizard

La console mostra automaticamente la versione più recente disponibile della soluzione nel menu a discesa della versione di distribuzione. Se hai già distribuito la soluzione, segui questa procedura per aggiornare la distribuzione alla versione più recente.

1. Vai a Launch Wizard Deployments.
2. Seleziona la distribuzione che desideri aggiornare.
3. Scegli Azioni, quindi Aggiorna la versione di distribuzione.
4. Seleziona la versione più recente tra le versioni di distribuzione disponibili.
5. Rivedi la configurazione.
6. Apporta le modifiche necessarie in ogni passaggio.
7. Conferma l'aggiornamento.

Aggiornamento tramite AWS CloudFormation

Se hai già distribuito la soluzione, segui questa procedura per aggiornare lo CloudFormation stack alla versione più recente.

1. Accedi alla CloudFormation console, seleziona lo stack esistente e seleziona CloudFormation Aggiorna stack.

2. Seleziona Effettua un aggiornamento diretto.
3. Seleziona Sostituisci modello esistente.
4. In Specificare il modello:
   a. Seleziona l'URL di Amazon S3.
   b. Copia il link del modello più recente.
   c. Incolla il link nella casella dell'URL di Amazon S3.
   d. Verifica che l'URL del modello corretto sia visualizzato nella casella di testo dell'URL di Amazon S3.
   e. Scegli Next (Successivo).
   f. Scegliere Next (Successivo) di nuovo.
5. In Parametri, esamina i parametri del modello e modificali se necessario. Per informazioni dettagliate sui parametri, consulta Launch the stack.
6. Scegli Next (Successivo).
7. Nella pagina Configure stack options (Configura opzioni pila), scegliere Next (Successivo).
8. Nella pagina Rivedi, verifica e conferma le impostazioni.
9. Seleziona la casella riconoscendo che il modello potrebbe creare risorse IAM.
10. Scegli Visualizza set di modifiche e verifica le modifiche.
11. Scegli Aggiorna stack per distribuire lo stack.

Puoi visualizzare lo stato dello stack nella CloudFormation console AWS nella colonna Status. Dovresti ricevere uno `UPDATE_COMPLETE` status in circa 15 minuti.

> **Note**
>
> Se riscontri problemi di autenticazione di Amazon Cognito durante l'accesso dal browser dopo l'aggiornamento dello stack, aggiorna il browser (Ctrl+Shift+R su Windows/Linux o Cmd +Shift+R su Mac) per cancellare i dati memorizzati nella cache e riprova.

Risoluzione dei problemi relativi agli aggiornamenti delle versioni precedenti alla v3.3.0

> **Note**
>
> Questa sezione si applica solo agli aggiornamenti delle versioni precedenti alla v3.3.0. Se stai eseguendo l'aggiornamento dalla versione 3.3.0 o successiva, segui la procedura di aggiornamento standard tramite AWS Launch Wizard o AWS. CloudFormation

1. Scarica il -aws.template. distributed-load-testing-on
2. Apri il modello, vai a `Conditions:` e cerca. `DLTCommonResourcesAppRegistryCondition`
3. Dovrebbe essere visualizzato un output simile al seguente:

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

4. Cambia il secondo `true` valore in`false`:

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

5. Utilizza il modello personalizzato per aggiornare il tuo stack seguendo i passaggi descritti in Update using AWS CloudFormation.
6. Questo aggiornamento rimuove le risorse relative al registro delle app dallo stack, consentendo il corretto completamento dell'aggiornamento.
7. Eseguite un altro aggiornamento dello stack utilizzando l'URL del modello più recente.

Aggiornamento degli stack regionali

Se la soluzione è stata distribuita in più regioni, è necessario aggiornare ogni stack regionale separatamente. Segui la procedura di aggiornamento standard per ogni CloudFormation stack regionale nelle regioni in cui hai distribuito l'infrastruttura di test.

Gestore di applicazioni AWS Systems Manager

Dopo aver aggiornato la soluzione, AWS Systems Manager Application Manager fornisce una visione a livello di applicazione della soluzione e delle sue risorse. Puoi usare Application Manager per:

- Monitora le risorse, i costi delle risorse distribuite su stack e account AWS e i log da una posizione centrale.
- Visualizza i dati operativi per le risorse della soluzione nel contesto di un'applicazione, come lo stato della distribuzione, gli CloudWatch allarmi, le configurazioni delle risorse e i problemi operativi.

Risoluzione dei problemi

La risoluzione dei problemi noti fornisce istruzioni per mitigare gli errori noti. Se queste istruzioni non risolvono il problema, Contatta AWS Support fornisce istruzioni per aprire un caso AWS Support per questa soluzione.

Risoluzione di problemi noti

Problema: stai utilizzando un VPC esistente e i tuoi test hanno esito negativo con lo stato Fallito, che genera il seguente messaggio di errore:

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

- Risoluzione:

Assicurati che le sottoreti esistano nel VPC specificato e che abbiano un percorso verso Internet con un gateway Internet o un gateway NAT. AWS Fargate necessita dell'accesso per estrarre l'immagine del contenitore dall'archivio pubblico per eseguire correttamente i test.

Problema: i test richiedono troppo tempo per essere eseguiti o sono bloccati a tempo indeterminato

- Risoluzione:

Annulla il test e controlla AWS Fargate per assicurarti che tutte le attività siano state interrotte. Se non si sono interrotti, interrompi manualmente tutte le attività di Fargate. Controlla i limiti delle attività Fargate su richiesta sul tuo account per assicurarti di poter avviare il numero di attività desiderato. Puoi anche controllare CloudWatch i log della funzione Lambda task-runner per maggiori informazioni sugli errori durante l'avvio delle attività di Fargate. Controlla i log CloudWatch ECS per i dettagli su ciò che accade nei container Fargate in esecuzione.

Problema: i test vengono avviati ma non vengono completati o lo stato delle attività ECS è sconosciuto

- Risoluzione:

Se hai selezionato l'opzione per fornire un VPC esistente nell'account in cui è stata distribuita la soluzione, assicurati che il VPC utilizzato da ECS Tasks disponga di indirizzi IP liberi sufficienti per

avviare il numero di attività fornite nell'input del test. La definizione dell'attività ECS utilizza l'immagine ECR che richiede un gateway Internet o un percorso verso Internet in modo che il servizio ECS possa fornire le attività scaricando l'immagine ECR della soluzione da aws-solutions/ distributed-load-testing-on - aws-load-tester. Se non riesci a fornire un percorso verso Internet poiché tutte le sottoreti nel VPC sono private, puoi ospitare l'immagine ECR nel tuo account utilizzando la cache pull through ECR. Aggiorna la definizione dell'attività con il nuovo URI dell'immagine ECR e crea una nuova revisione. Una volta aggiornata la definizione del task, è necessario aggiornare la configurazione della soluzione nella tabella DynamoDB per utilizzare la nuova revisione. Il nome della tabella DynamoDB si trova nella scheda stack CloudFormation outputs sotto la chiave. ScenariosTable Aggiorna l'attributo taskDefinition per l'elemento con la chiave testID e il valore region- [SOLUTION-DEPLOYED-REGION].

Problema: i test devono utilizzare un endpoint privato o non disponibile tramite il gateway Internet

- Risoluzione:

Quando testate endpoint API privati che non sono accessibili tramite il gateway Internet, prendete in considerazione i seguenti approcci:

1. Configurazione di rete: assicurati che le tabelle di routing della sottorete utilizzate dalle attività ECS siano aggiornate con un percorso verso l'intervallo di indirizzi IP dell'endpoint privato oggetto del test. Ciò consente al traffico di test di raggiungere l'endpoint privato all'interno del tuo VPC.
2. Risoluzione DNS: per i domini personalizzati, configura le impostazioni DNS nel tuo VPC per risolvere il nome di dominio dell'endpoint privato. Per istruzioni dettagliate, consulta la documentazione DNS di VPC.
3. Endpoint VPC: se stai testando i servizi AWS, prendi in considerazione l'utilizzo di endpoint VPC (PrivateLinkAWS) per stabilire una connettività privata. Ad esempio, per testare un gateway API privato, puoi creare un endpoint VPC per API Gateway. Consulta la documentazione di Private API Gateway.
4. Peering VPC: se l'endpoint privato si trova in un VPC diverso, stabilisci il peering VPC tra il VPC in cui è implementata la soluzione e il VPC contenente l'endpoint privato. Configura VPCs le tabelle di routing appropriate in entrambi. Consulta la documentazione del peering VPC.
5. Transit Gateway: per scenari di rete più complessi che coinvolgono più utenti VPCs, prendi in considerazione l'utilizzo di AWS Transit Gateway per instradare il traffico tra il VPC della soluzione e il VPC contenente l'endpoint privato. Consultate la documentazione di Transit Gateway.

6. Gruppi di sicurezza: assicurati che i gruppi di sicurezza associati alle tue attività ECS consentano il traffico in uscita verso l'endpoint privato e che i gruppi di sicurezza dell'endpoint privato consentano il traffico in entrata dalle attività ECS.

Per testare gli Application Load Balancer interni o le istanze EC2, assicurati che gli intervalli CIDR VPC non si sovrappongano e che le rotte necessarie siano configurate nelle tabelle di routing.

Problema: i test vengono completati ma i risultati non sono disponibili nell'interfaccia utente

- Risoluzione:

Se il test è stato completato ma i risultati non sono disponibili nell'interfaccia utente, i file dei risultati dovrebbero essere ancora disponibili nel bucket S3 delle attività ECS che hanno eseguito i test. Questa è una limitazione nota della soluzione. Nell'architettura attuale, la soluzione utilizza una funzione Lambda di analisi dei risultati per riepilogare i risultati di più attività ECS, che vengono quindi archiviate come elemento nella tabella DynamoDB. La tabella DynamoDB ha un limite di dimensione massima dell'elemento di 400 KB. Questa limitazione viene raggiunta in base alla complessità dello script di test, alla concorrenza e al numero di attività utilizzate. L'errore non significa che il test stia fallendo; indica che il processo di riepilogo dei risultati e di memorizzazione nella tabella DynamoDB per le operazioni CRUD non è riuscito. I risultati sono ancora disponibili nel bucket S3 per lo scenario di test.

Contattare AWS Support

Se disponi di AWS Business Support+, AWS Enterprise Support o Unified Operations, puoi utilizzare AWS Support Center per ottenere l'assistenza di esperti con questa soluzione. Le istruzioni per eseguire tali operazioni sono fornite nelle sezioni seguenti.

Crea un caso

1. Accedi al Support Center.
2. Scegli Crea caso.

Come possiamo aiutarti?

1. Scegli Technical

2. Per Assistenza, seleziona Soluzioni.
3. Per Categoria, seleziona Distributed Load Testing on AWS.
4. Per Severity, seleziona l'opzione più adatta al tuo caso d'uso.
5. Quando si inseriscono i campi Servizio, Categoria e Severità, l'interfaccia compila i collegamenti alle domande più comuni per la risoluzione dei problemi. Se non riesci a risolvere le tue domande con questi link, scegli Passaggio successivo: Informazioni aggiuntive.

Informazioni aggiuntive

1. In Oggetto, inserisci il testo che riassume la domanda o il problema.
2. Per Descrizione, descrivi il problema in dettaglio, includendo il nome di questo prodotto e la versione che stai utilizzando, ad esempio: Distributed Load Testing on AWS Vx.y.z.
3. Scegli Allega file.
4. Allega le informazioni di cui AWS Support ha bisogno per elaborare la richiesta.

Aiutaci a risolvere il tuo caso più velocemente

1. Inserisci le informazioni richieste.
2. Scegli Passaggio successivo: risolvi ora o contattaci.

Risolvi subito o contattaci

1. Rivedi le soluzioni Solve now.
2. Se non riesci a risolvere il problema con queste soluzioni, scegli Contattaci, inserisci le informazioni richieste e scegli Invia.

Disinstalla la soluzione

Puoi disinstallare la soluzione Distributed Load Testing on AWS dalla Console di gestione AWS o utilizzando l'interfaccia a riga di comando AWS. È necessario eliminare manualmente la console, lo scenario e i bucket di registrazione di Amazon Simple Storage Service (Amazon S3) creati da questa soluzione. Le implementazioni delle soluzioni AWS non le eliminano automaticamente nel caso in cui ci siano dati da conservare.

> **Note**
>
> Se hai distribuito stack regionali, devi eliminare gli stack in quelle regioni prima di eliminare lo stack principale.

Utilizzando la Console di gestione AWS

AWS CloudFormation

1. Accedi alla CloudFormation console AWS.
2. Nella pagina Stacks, seleziona lo stack di installazione di questa soluzione.
3. Scegli Elimina.

AWS Launch Wizard

1. Accedi alla console AWS Launch Wizard.
2. Nella pagina Launch Wizard Deployments, seleziona la distribuzione di questa soluzione.
3. Scegli Operazioni, quindi Elimina.
4. Conferma l'eliminazione.

Utilizzo dell'interfaccia a riga di comando AWS

Determina se l'AWS Command Line Interface (AWS CLI) è disponibile nel tuo ambiente. Per istruzioni di installazione, consulta What Is the AWS Command Line Interface nella AWS CLI User Guide. Dopo aver verificato che la CLI di AWS è disponibile, esegui il comando seguente.

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

Eliminazione dei bucket Amazon S3

Questa soluzione è configurata per conservare i bucket Amazon S3 creati dalla soluzione (per la distribuzione in una regione opt-in) se decidi di eliminare lo stack CloudFormation AWS per prevenire la perdita accidentale di dati. Dopo aver disinstallato la soluzione, puoi eliminare manualmente questo bucket S3 se non hai bisogno di conservare i dati. Segui questi passaggi per eliminare il bucket Amazon S3.

1. Accedere alla console Amazon S3.
2. Scegli Bucket dal riquadro di navigazione a sinistra.
3. Nel campo Trova i bucket per nome, inserisci il nome dello stack di questa soluzione.
4. Seleziona uno dei bucket S3 della soluzione e scegli Empty.
5. Inserisci l'eliminazione definitiva nel campo di verifica e scegli Vuoto.
6. Seleziona il bucket S3 che hai appena svuotato e scegli Elimina.
7. Inserisci il nome del bucket S3 nel campo di verifica e scegli Elimina bucket.

Ripeti i passaggi da 4 a 7 fino a eliminare tutti i bucket S3.

Per eliminare il bucket S3 utilizzando AWS CLI, esegui il seguente comando:

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

Usa la soluzione

Questa sezione fornisce una guida completa all'uso della soluzione Distributed Load Testing on AWS, dalla creazione del primo scenario di test all'analisi dettagliata dei risultati. Il flusso di lavoro include la creazione di uno scenario di test, l'esecuzione di un test e l'analisi dei risultati dei test.

Creare uno scenario di test

La creazione di uno scenario di test prevede quattro passaggi principali: configurazione delle impostazioni generali, definizione dello scenario, definizione dei modelli di traffico e revisione della configurazione.

Fase 1: impostazioni generali

Configura i parametri di base per il test di carico, tra cui il nome del test, la descrizione e le opzioni generali di configurazione.

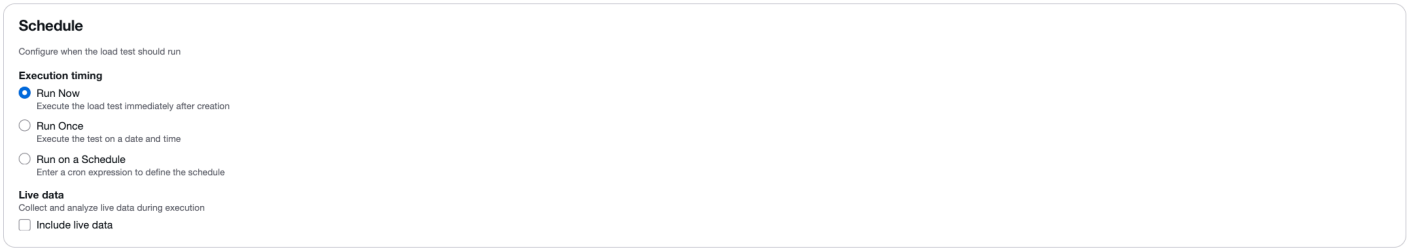
Identificazione del test

- Nome del test (obbligatorio): un nome descrittivo per lo scenario di test
- Descrizione del test (obbligatorio): dettagli aggiuntivi sullo scopo e sulla configurazione del test
- Tag (facoltativo): aggiungi fino a 5 tag per classificare e organizzare gli scenari di test

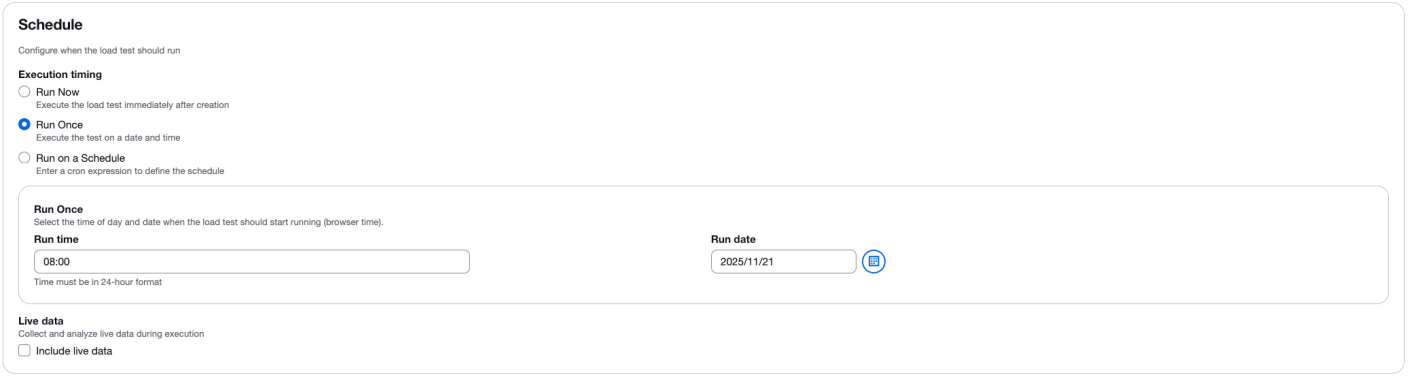
Opzioni di pianificazione

Configura quando deve essere eseguito il test:

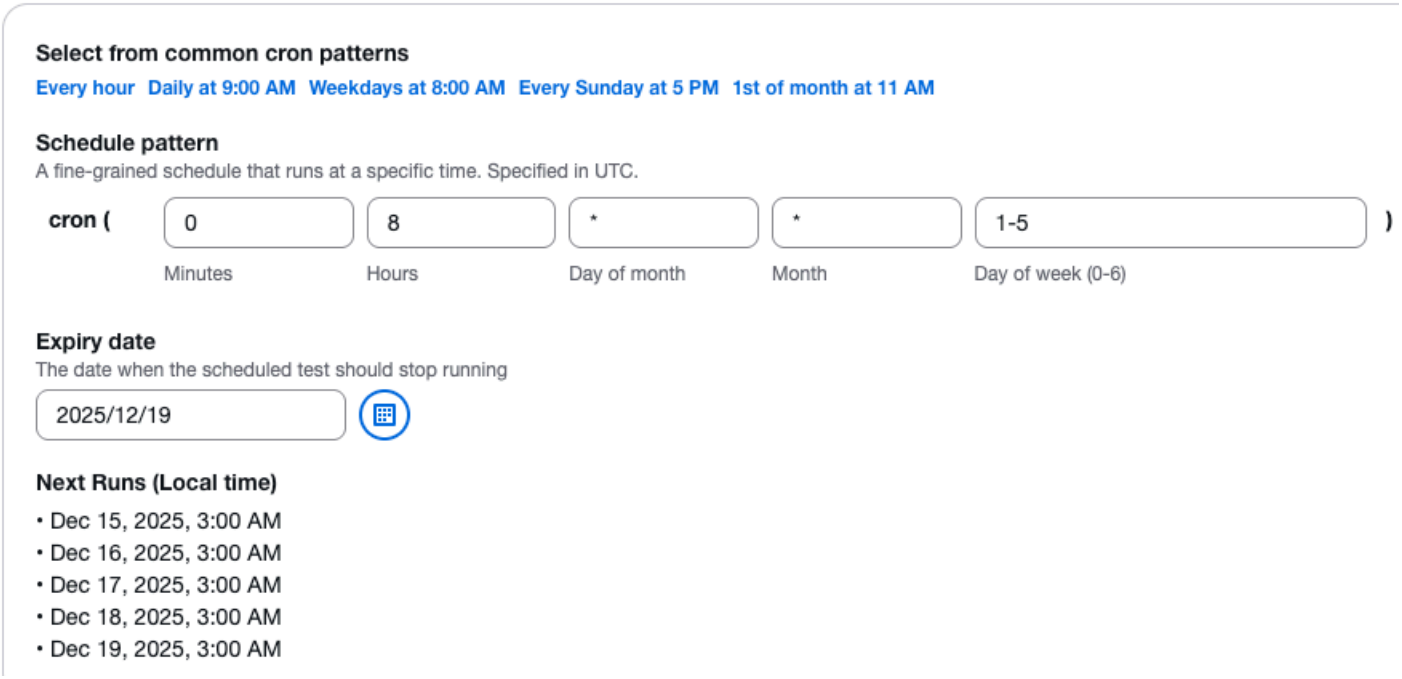
- Esegui ora: esegui il test immediatamente dopo la creazione.



- Esegui una volta: pianifica l'esecuzione del test in una data e un'ora specifiche.



- Esegui in base a una pianificazione: utilizza la pianificazione basata su cronn per eseguire i test automaticamente a intervalli regolari. Puoi scegliere tra modelli comuni (ogni ora, ogni giorno, settimanale) o definire un'espressione cron personalizzata.



Flusso di lavoro di pianificazione

Quando si pianifica un test, si verifica il seguente flusso di lavoro:

- I parametri di pianificazione vengono inviati all'API della soluzione tramite Amazon API Gateway.
- L'API passa i parametri a una funzione Lambda che crea una regola CloudWatch Events programmata per l'esecuzione nella data specificata.
- Per i test una tantum (Run Once), la regola CloudWatch Events viene eseguita nella data specificata e la funzione `api-services` Lambda esegue il test.

- Per i test ricorrenti (Esegui in base a una pianificazione), la regola CloudWatch Events si attiva alla data specificata e la funzione `api-services` Lambda crea una nuova regola che viene eseguita immediatamente e in modo ricorrente in base alla frequenza specificata.

Dati in tempo reale

Seleziona la casella di controllo Includi dati in tempo reale per visualizzare le metriche in tempo reale durante l'esecuzione del test. Se abilitato, puoi monitorare:

- Tempo di risposta medio.
- Conteggi di utenti virtuali.
- Le richieste riuscite contano.
- Le richieste non riuscite contano.

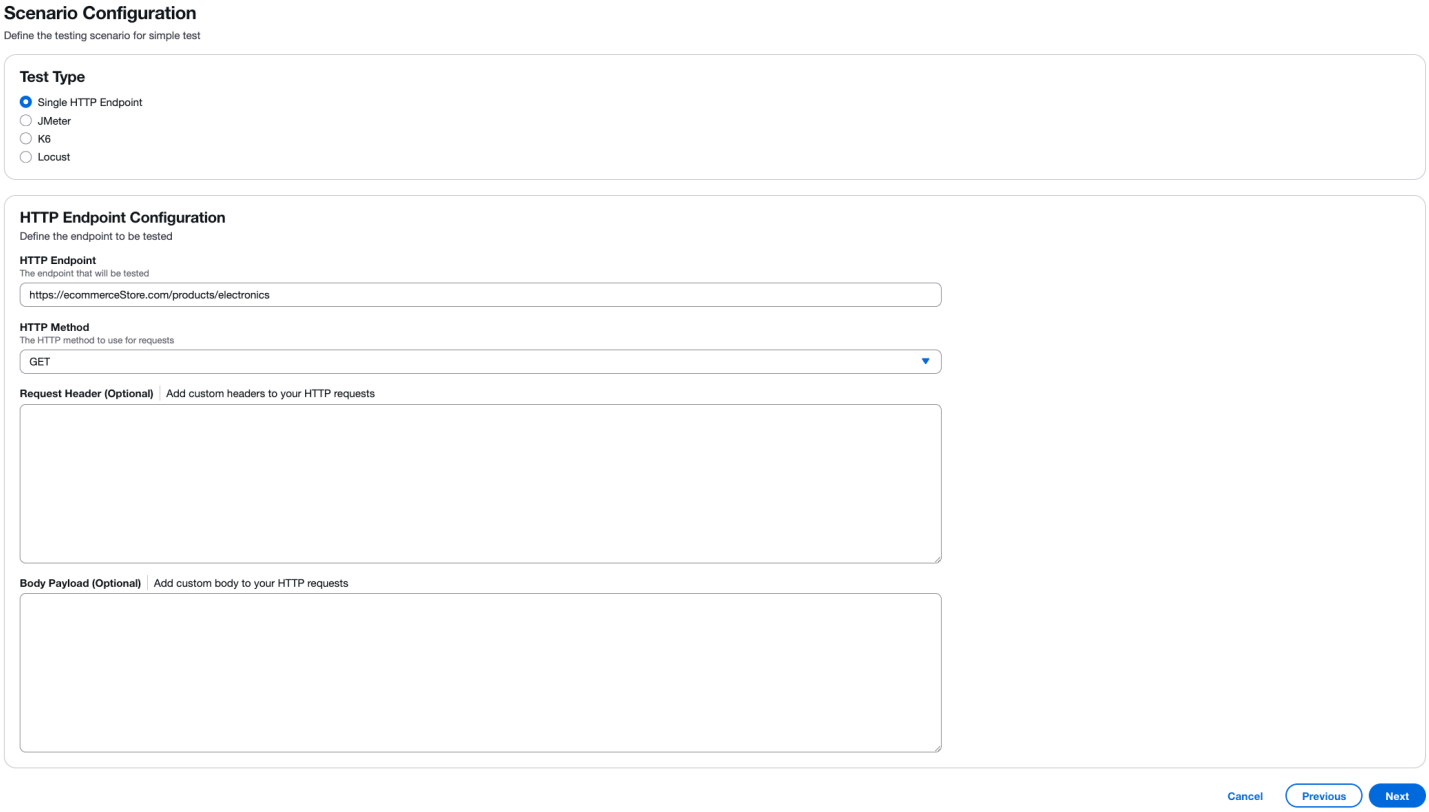
La funzionalità live data fornisce grafici in tempo reale con dati aggregati a intervalli di un secondo. Per ulteriori informazioni, consulta Monitoraggio con dati in tempo reale.

Fase 2: Configurazione dello scenario

Definisci lo scenario di test specifico e seleziona il tuo framework di test preferito.

Selezione del tipo di test

Scegli il tipo di test di carico che desideri eseguire:



- Endpoint HTTP singolo: testa un singolo endpoint API o una pagina Web con una configurazione semplice.
- JMeter- Carica script JMeter di test (file.jmx o archivi.zip).
- K6 - Carica gli script di test K6 (file.js o archivi.zip).
- Locust - Carica gli script di test Locust (file.py o archivi.zip).

Immagine di configurazione dell'endpoint HTTP: :images/test-types.png [Seleziona il tipo di test da eseguire] Quando è selezionato «Single HTTP Endpoint», configura queste impostazioni:

Endpoint HTTP (obbligatorio)

Inserisci l'URL completo dell'endpoint che desideri testare. Ad esempio, `https://api.example.com/users`. Assicurati che l'endpoint sia accessibile dall'infrastruttura AWS.

Metodo HTTP (obbligatorio)

Seleziona il metodo HTTP per le tue richieste. Il valore predefinito è “`GET`”. Altre opzioni includono `POSTPUT`,`DELETE`,`PATCH`,`HEAD`, e`OPTIONS`.

Intestazione della richiesta (opzionale)

Aggiungi intestazioni HTTP personalizzate alle tue richieste. Esempi comuni comprendono:

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

Scegli Aggiungi intestazione per includere più intestazioni.

Body Payload (opzionale)

Aggiungi il contenuto del corpo della richiesta per le richieste POST o PUT. Supporta i formati JSON, XML o testo semplice. Ad esempio: `{"userId": 123, "action": "test"}`.

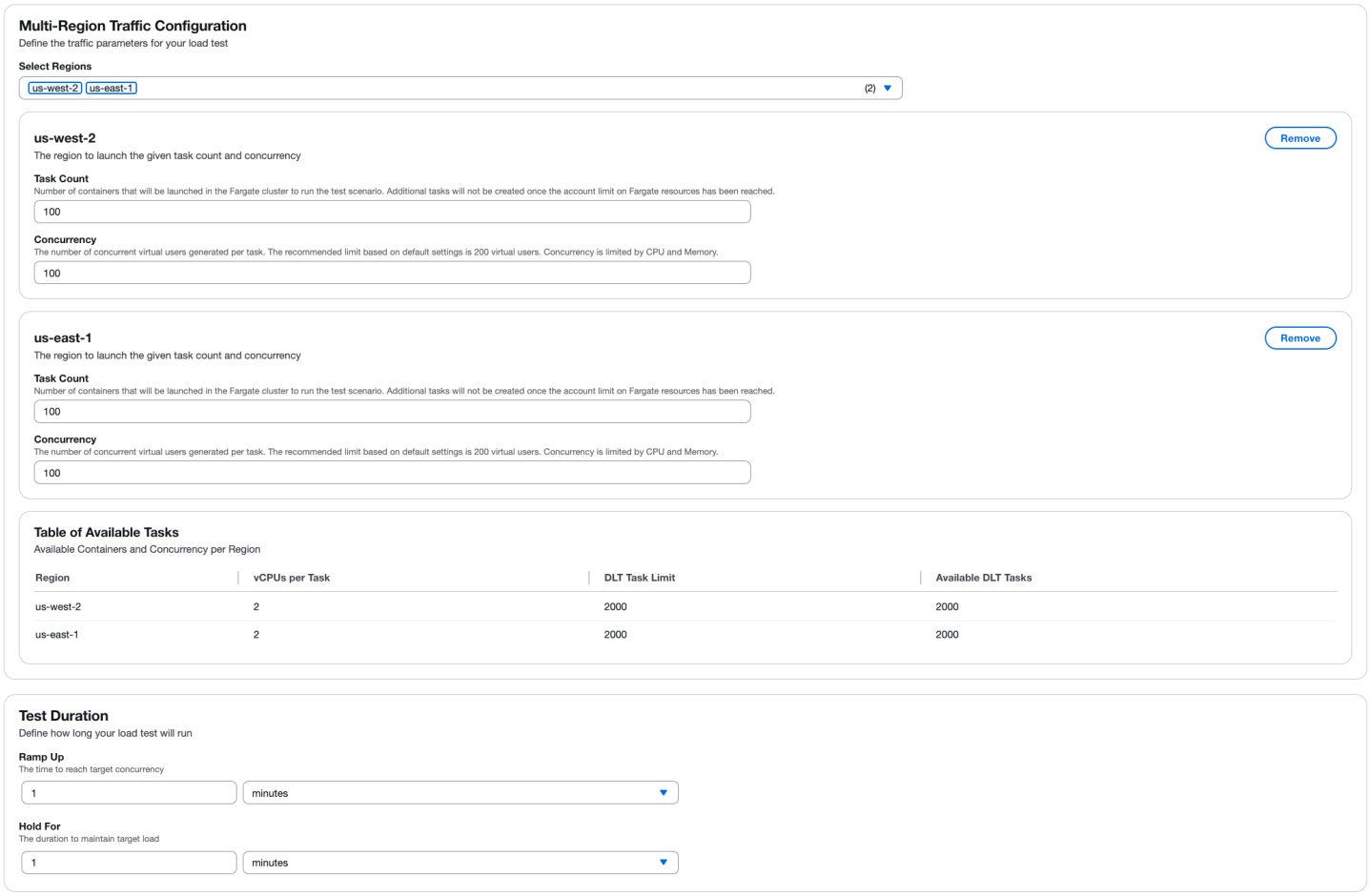
Script del framework di test

Quando usi JMeter K6 o Locust, carica il tuo file di script di test o un archivio.zip contenente lo script di test e i file di supporto. Puoi JMeter infatti includere plugin personalizzati in una `/plugins` cartella all'interno del tuo archivio.zip.

> **Important**
>
> Sebbene lo script di test (JMeter, K6 o Locust) possa definire la concorrenza (utenti virtuali), i tassi di transazione (TPS), i tempi di accelerazione e altri parametri di caricamento, la soluzione sostituirà queste configurazioni con i valori specificati nella schermata Traffic Shape durante la creazione del test. La configurazione Traffic Shape controlla il conteggio delle attività, la concorrenza (utenti virtuali per attività), la durata dell'accelerazione e la durata di attesa per l'esecuzione del test.

Fase 3: Forma del traffico

Configura la modalità di distribuzione del traffico durante il test, incluso il supporto multiregionale.



Configurazione del traffico multiregionale

Seleziona una o più regioni AWS per distribuire geograficamente il test di carico. Per ogni regione selezionata, configura:

Conteggio delle attività

Il numero di contenitori (attività) che verranno avviati nel cluster Fargate per lo scenario di test. Non verranno create attività aggiuntive una volta che l'account avrà raggiunto il limite «La risorsa Fargate è stata raggiunta».

Concurrency (Simultaneità)

Il numero di utenti virtuali simultanei generati per attività. Il limite consigliato si basa sulle impostazioni predefinite di 2 v CPUs per attività. La concorrenza è limitata dalle risorse di CPU e memoria.

Determina il numero di utenti

Il numero di utenti che un container può supportare per un test può essere determinato aumentando gradualmente il numero di utenti e monitorando le prestazioni in Amazon CloudWatch. Una volta notato che le prestazioni della CPU e della memoria si stanno avvicinando ai limiti, hai raggiunto il numero massimo di utenti che un container può supportare per quel test nella sua configurazione predefinita (2 vCPU e 4 GB di memoria).

Processo di calibrazione

È possibile iniziare a determinare i limiti di utenti simultanei per il test utilizzando il seguente esempio:

1. Crea un test con non più di 200 utenti.
2. Durante l'esecuzione del test, monitora la CPU e la memoria utilizzando la CloudWatch console:
   a. Dal riquadro di navigazione a sinistra, in Container Insights, seleziona Performance Monitoring.
   b. Nella pagina di monitoraggio delle prestazioni, dal menu a discesa a sinistra, seleziona ECS Clusters.
   c. Dal menu a discesa a destra, seleziona il tuo cluster Amazon Elastic Container Service (Amazon ECS).
3. Durante il monitoraggio, controlla la CPU e la memoria. Se la CPU non supera il 75% o la memoria non supera l'85% (ignora i picchi occasionali), puoi eseguire un altro test con un numero maggiore di utenti.

Ripetere i passaggi 1-3 se il test non ha superato i limiti di risorse. Facoltativamente, puoi aumentare le risorse del contenitore per consentire un numero maggiore di utenti simultanei. Tuttavia, ciò comporta un costo più elevato. Per i dettagli, consulta la Guida per gli sviluppatori.

> **Note**
>
> Per risultati accurati, esegui solo un test alla volta per determinare i limiti degli utenti simultanei. Tutti i test utilizzano lo stesso cluster e CloudWatch Container Insights aggrega i dati sulle prestazioni in base al cluster. Ciò fa sì che entrambi i test vengano segnalati contemporaneamente a CloudWatch Container Insights, il che si traduce in metriche di utilizzo delle risorse imprecise per un singolo test.

Per ulteriori informazioni sulla calibrazione degli utenti per motore, consulta la sezione Calibrazione di un test Taurus nella documentazione. BlazeMeter

> **Note**
>
> La soluzione mostra le informazioni sulla capacità disponibile per ogni regione, aiutandovi a pianificare la configurazione del test entro i limiti disponibili.

Tabella delle attività disponibili

La tabella delle attività disponibili mostra la disponibilità delle risorse per ogni regione selezionata:

- Regione: il nome della regione AWS.
- v CPUs per Task: il numero di elementi virtuali CPUs assegnati a ciascuna attività (impostazione predefinita: 2).
- Limite attività DLT: il numero massimo di attività che è possibile creare in base ai limiti di Fargate dell'account (impostazione predefinita: 2000).
- Attività DLT disponibili: il numero attuale di attività disponibili per l'uso nella regione (impostazione predefinita: 2000).

Table of Available Tasks

Available Containers and Concurrency per Region

Region	vCPUs per Task	DLT Task Limit	Available DLT Tasks
us-west-2	2	2000	2000
us-east-1	2	2000	2000

Per aumentare il numero di attività disponibili o v CPUs per attività, consulta la Guida per gli sviluppatori.

Durata del test

Definisci per quanto tempo verrà eseguito il test di carico:

Rampa verso l'alto

Il tempo necessario per raggiungere la concorrenza prefissata. In questo periodo, il carico aumenta gradualmente da 0 al livello di concorrenza configurato.

Tieni premuto per

La durata necessaria per mantenere il carico previsto. Il test prosegue in piena concomitanza per questo periodo.

Fase 4: Revisione e creazione

Rivedi tutte le configurazioni prima di creare lo scenario di test. Verifica:

- Impostazioni generali (nome, descrizione, pianificazione).
- Configurazione dello scenario (tipo di test, endpoint o script).
- Forma del traffico (attività, utenti, durata, regioni).

Dopo la revisione, scegli Crea per salvare lo scenario di test.

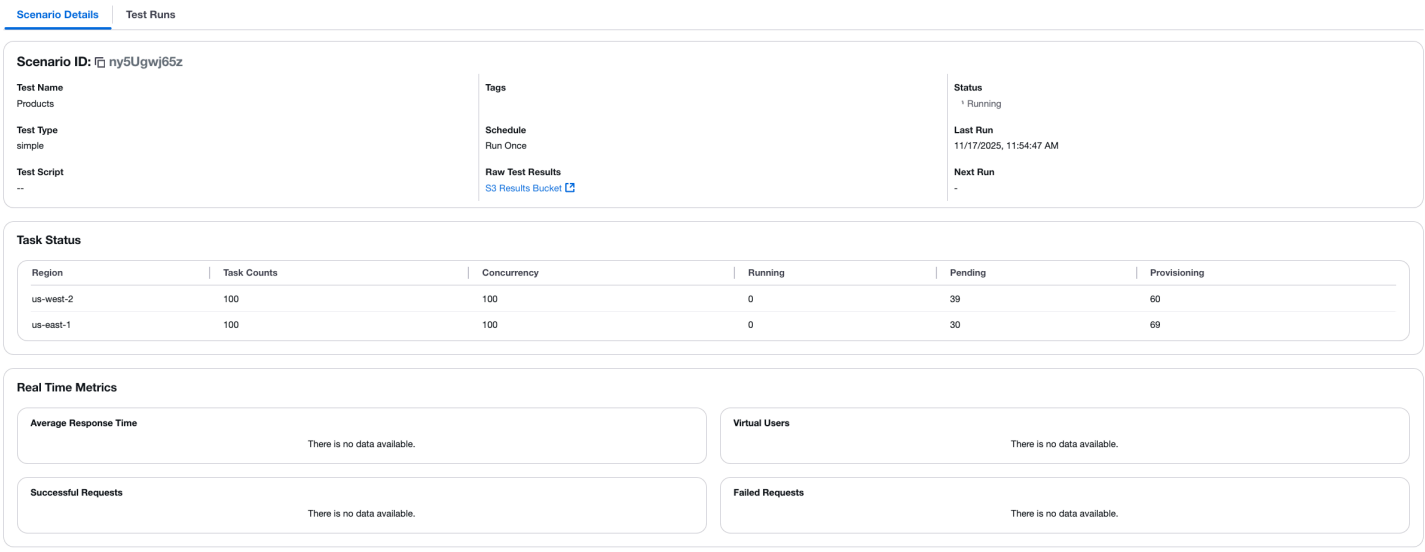
Gestione degli scenari di test

Dopo aver creato uno scenario di test, puoi:

- Modifica: modifica la configurazione del test. Casi di utilizzo comune comprendono:
  - Perfezionamento della forma del traffico per raggiungere la velocità di transazione desiderata.
- Copia: duplica uno scenario di test esistente per creare varianti. Casi di utilizzo comune comprendono:
  - Aggiornamento degli endpoint o aggiunta headers/body di parametri.
  - Aggiungere o modificare script di test.
- Elimina: rimuovi gli scenari di test che non ti servono più.

Esegui uno scenario di test

Dopo aver creato uno scenario di test, è possibile eseguirlo immediatamente o pianificarne l'esecuzione in un momento specifico delle future fasi. Quando accedi a un test in esecuzione, la console visualizza la scheda Dettagli dello scenario con lo stato e le metriche delle attività in tempo reale.



Visualizzazione dei dettagli dello scenario

La scheda Dettagli dello scenario mostra le informazioni chiave sul test. La tabella sullo stato delle attività contiene informazioni in tempo reale per ogni regione.

Tabella dello stato delle attività

La tabella dello stato delle attività mostra informazioni in tempo reale per ogni regione:

- Regione: la regione AWS in cui vengono eseguite le attività
- Task Counts: il numero totale di attività configurate per la regione
- Concorrenza: il numero di utenti virtuali per attività
- In esecuzione: numero di attività attualmente in esecuzione nel test
- In sospeso: numero di attività in attesa di avvio
- Approvvigionamento: numero di attività in fase di assegnazione

Workflow di esecuzione dei test

All'avvio di un test, si verifica il seguente flusso di lavoro:

1. Provisioning delle attività: la soluzione effettua il provisioning dei container (task) nelle regioni AWS specificate. Le attività vengono visualizzate nella colonna «Provisioning».

2. Avvio delle attività: la soluzione continua a fornire le attività fino al raggiungimento del numero di attività previsto in ciascuna regione. Le attività passano da «Fornitura» a «In sospeso» a «In esecuzione».
3. Generazione di traffico: dopo che la soluzione ha effettuato il provisioning di tutte le attività in una regione, inizia a inviare traffico all'endpoint di destinazione.
4. Esecuzione del test: il test viene eseguito per la durata configurata (ramp-up + tempo di attesa).
5. Analisi dei risultati: al termine del test, un processo di analisi in background aggrega ed elabora i risultati di tutte le regioni.

Stati di esecuzione del test

Le esecuzioni dei test possono avere i seguenti stati:

- Pianificato: l'esecuzione del test è programmata per le prossime future.
- In esecuzione: il test è attualmente in corso.
- Annullato: un utente ha annullato un'esecuzione di test in corso.
- Errata: durante l'esecuzione del test si è verificato un errore.
- Completato: l'esecuzione del test è stata completata correttamente e i risultati sono pronti.

Monitoraggio con dati in tempo reale

Se hai abilitato i dati in tempo reale durante la creazione dello scenario di test, puoi visualizzare le metriche in tempo reale mentre il test è in esecuzione. La sezione Metriche in tempo reale mostra quattro grafici che si aggiornano continuamente man mano che il test procede, con dati aggregati a intervalli di un secondo.

Descrizioni dei grafici

Tempo di risposta medio

Visualizza il tempo di risposta medio in secondi per le richieste elaborate da ciascuna regione. L'asse Y mostra il tempo di risposta in secondi e l'asse X mostra l'ora del giorno. Ogni regione è rappresentata da un colore diverso nella legenda.

Utenti virtuali

Mostra il numero di utenti virtuali simultanei che generano attivamente carico in ciascuna regione. Il grafico mostra l'aumento degli utenti virtuali durante il test e mantiene il livello di concorrenza previsto.

Richieste riuscite

Visualizza il conteggio cumulativo delle richieste riuscite nel tempo per ogni regione. Il grafico mostra la velocità con cui vengono elaborate le richieste riuscite.

Richieste non riuscite

Mostra il numero cumulativo di richieste non riuscite nel tempo per ogni regione. Un conteggio basso o nullo indica una corretta esecuzione del test.

Visualizzazione in più regioni

Quando si eseguono test su più regioni, ogni grafico mostra i dati per tutte le regioni contemporaneamente. La legenda nella parte inferiore di ogni grafico identifica il colore che rappresenta ciascuna regione (ad esempio, us-west-2 e us-east-1).

Implementazione tecnica

Il gruppo di CloudWatch log per le attività di Fargate contiene un filtro di sottoscrizione che acquisisce i risultati dei test. Quando viene rilevato il pattern, una funzione Lambda struttura i dati e li pubblica su un argomento di AWS IoT Core. La console web sottoscrive questo argomento e visualizza le metriche in tempo reale.

> **Note**
>
> I dati in tempo reale sono temporanei e disponibili solo durante l'esecuzione del test. La console web mantiene un massimo di 5.000 punti dati, dopodiché i dati più vecchi vengono sostituiti con i più recenti. Se la pagina si aggiorna, i grafici saranno vuoti e inizieranno dal successivo punto dati disponibile. Una volta completato il test, la soluzione archivia i dati dei risultati in DynamoDB e Amazon S3. Se non ci sono ancora dati disponibili, i grafici mostrano «Non ci sono dati disponibili».

Annullamento di un test

È possibile annullare un test in esecuzione dalla console Web. Quando si annulla un test, si verifica il seguente flusso di lavoro:

1. La richiesta di cancellazione viene inviata all'`microservices`API
2. L'`microservices`API richiama la funzione `task-canceler` Lambda, che interrompe tutte le attività attualmente avviate.
3. Se la funzione `task-runner` Lambda continua a essere eseguita dopo la chiamata di annullamento iniziale, le attività potrebbero continuare ad avviarsi brevemente
4. Al termine della funzione `task-runner` Lambda, AWS Step Functions procede alla `Cancel Test` fase, che esegue nuovamente la funzione `task-canceler` Lambda per interrompere le attività rimanenti

> **Note**
>
> I test annullati richiedono tempo per completare il processo di spegnimento poiché la soluzione chiude tutti i contenitori. Lo stato del test cambierà in «Annullato» una volta ripulite tutte le risorse.

Esplora i risultati dei test

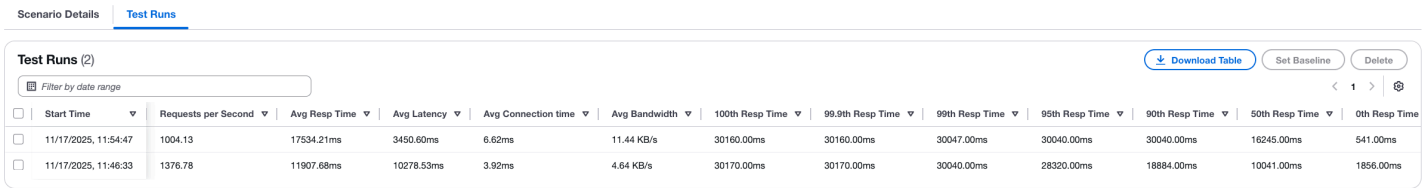
Una volta completato il processo di analisi, i risultati dei test sono disponibili per l'analisi. La soluzione fornisce metriche e strumenti completi per aiutarvi a comprendere le prestazioni dell'applicazione sotto carico.

Metriche di riepilogo dell'esecuzione del test

Al termine di un test, la soluzione genera un riepilogo che include le seguenti metriche:

- Tempo di risposta medio: il tempo di risposta medio, in secondi, per tutte le richieste generate dal test.
- Latenza media: la latenza media, in secondi, per tutte le richieste generate dal test.
- Tempo medio di connessione: il tempo medio, in secondi, necessario per connettersi all'host per tutte le richieste.
- Larghezza di banda media: la larghezza di banda media per tutte le richieste generate dal test.
- Conteggio totale: il numero totale di richieste.
- Numero di richieste riuscite: il numero totale di richieste riuscite.
- Conteggio errori: il numero totale di errori.
- Richieste al secondo: la media delle richieste al secondo per tutte le richieste generate dal test.
- Percentili: percentili del tempo di risposta tra cui p50 (mediana), p90, p95 e p99, che mostrano la distribuzione dei tempi di risposta tra tutte le richieste.

Tabella delle esecuzioni di test



La tabella dei test eseguiti mostra tutte le esecuzioni di test cronologiche per uno scenario. Puoi:

- Visualizza le metriche di riepilogo per ogni esecuzione del test.
- Imposta un test di base per il confronto delle prestazioni.
- Scarica la tabella come file CSV.
- Attiva le colonne per personalizzare la visualizzazione.
- Seleziona un'esecuzione di test per visualizzare i risultati dettagliati.

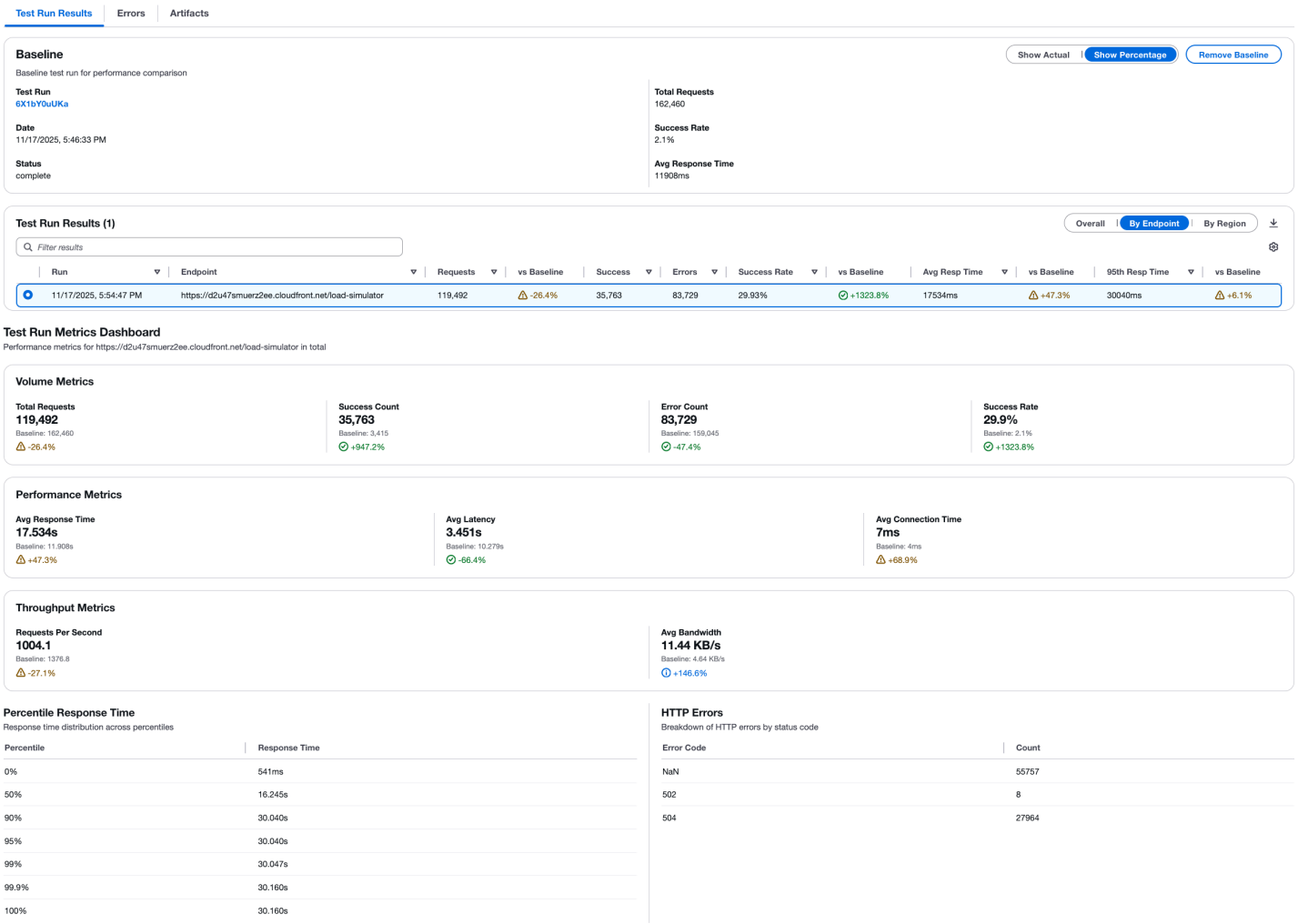
Confronto di base

È possibile designare un'esecuzione di test come base per confrontare le esecuzioni di test future con essa. Quando viene impostata una linea di base:

- La tabella delle esecuzioni dei test mostra le differenze percentuali (+/-%) rispetto alla linea di base per ciascuna metrica.
- L'indicatore di base consente di identificare rapidamente i miglioramenti o le regressioni delle prestazioni.
- È possibile modificare o cancellare la linea di base in qualsiasi momento.

Risultati dettagliati dei test

La selezione di un'esecuzione del test apre la visualizzazione dettagliata dei risultati con tre schede: Risultati dell'esecuzione del test, Errori e Artefatti.



Informazioni di base

Se è impostato un test di base, questo viene visualizzato nella parte superiore della pagina. È possibile scegliere Mostra valore effettivo, Mostra percentuale o Rimuovi linea di base per controllare la modalità di visualizzazione dei confronti di base.

Tabella dei risultati del test Run

La tabella dei risultati fornisce metriche dettagliate con le seguenti funzionalità:

Viste dimensionali

Passa da una visualizzazione all'altra utilizzando i pulsanti dimensionali:

- Complessivamente: risultati aggregati su tutti gli endpoint e le regioni
- Per endpoint: risultati suddivisi per endpoint individuali

- Per regione - Risultati suddivisi per regione AWS

Pulsanti di azione

- Mostra effettivi: mostra i valori metrici effettivi
- Mostra percentuale: mostra le differenze percentuali rispetto alla linea di base
- Rimuovi linea di base: cancella il confronto della linea di base

Esportazione e personalizzazione dei dati

- Scarica la tabella dei risultati come file CSV
- Attiva/disattiva le colonne per personalizzare la visualizzazione
- Filtra e ordina i dati per concentrarti su metriche specifiche
- Filtra e ordina i dati per concentrarti su metriche specifiche.

Scheda Errori

La scheda errori fornisce un'analisi dettagliata degli errori:

- Visualizza il conteggio degli errori per tipo.
- Visualizza gli errori aggregati per test complessivo o per endpoint.
- Identifica i modelli nelle richieste non riuscite.
- Risolvi i problemi con endpoint o regioni specifici.

Scheda Artefatti

La scheda artefatti consente di accedere a tutti i file generati durante l'esecuzione del test:

- Visualizza singoli artefatti (registri, file dei risultati).
- Scarica artefatti specifici per l'analisi offline.
- Scarica tutti gli artefatti del test in un unico archivio.

Struttura dei risultati S3

Nella versione 4.0, la struttura dei risultati di S3 è stata modificata per migliorare l'organizzazione:

- Nuova struttura `-scenario-id/test-run-id/results-files`.
- Struttura precedente: i test eseguiti prima della versione 4.0 mostrano tutti i file dei risultati a livello di ID dello scenario.

> **Note**
>
> I risultati dei test vengono visualizzati nella console. Puoi anche accedere ai risultati grezzi dei test direttamente nel bucket Amazon S3 sotto la cartella. `Results` Per ulteriori informazioni sui risultati dei test Taurus, consulta Generazione di report sui test nel manuale utente di Taurus.

Integrazione con server MCP

Se hai distribuito il componente server MCP opzionale durante l'implementazione della soluzione, puoi integrare la soluzione Distributed Load Testing con strumenti di sviluppo AI che supportano il Model Context Protocol. Il server MCP fornisce l'accesso programmatico per recuperare, gestire e analizzare i test di carico tramite assistenti AI.

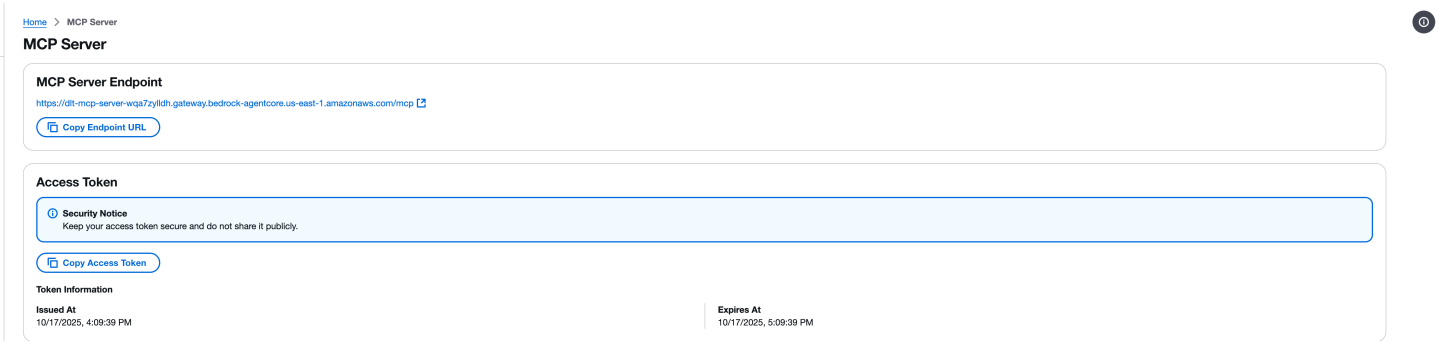
I clienti possono connettersi al server MCP DLT utilizzando il client di loro scelta (Amazon Q, Claude, ecc.), ognuno con istruzioni di configurazione leggermente diverse. Questa sezione fornisce istruzioni di configurazione per MCP Inspector, Amazon Q CLI, Cline e Amazon Q Suite.

Fase 1: Ottieni l'endpoint MCP e il token di accesso

Prima di configurare qualsiasi client MCP, è necessario recuperare l'endpoint del server MCP e il token di accesso dalla console web DLT.

1. Vai alla pagina MCP Server nella console web Distributed Load Testing.
2. Individuate la sezione MCP Server Endpoint.
3. Copia l'URL dell'endpoint utilizzando il pulsante Copia URL dell'endpoint. L'URL dell'endpoint segue il formato: `https://{gateway-id}.gateway.bedrock-agentcore.{region}.amazonaws.com/mcp`
4. Individua la sezione Access Token.
5. Copia il token di accesso utilizzando il pulsante Copia token di accesso.

> **Important**
>
> Mantieni sicuro il tuo token di accesso e non condividerlo pubblicamente. Il token fornisce l'accesso in sola lettura alla soluzione Distributed Load Testing tramite l'interfaccia MCP.



Fase 2: Test con MCP Inspector

Il Model Context Protocol offre MCP Inspector, uno strumento per connettersi direttamente ai server MCP e richiamare strumenti. Ciò fornisce una comoda interfaccia utente e esempi di richieste di rete per testare la connessione al server MCP prima di configurare i client AI.

> **Note**
>
> MCP Inspector richiede la versione 0.17 o successiva. Tutte le richieste possono essere effettuate anche direttamente con JSON RPC, ma MCP Inspector offre un'interfaccia più intuitiva.

Installa e avvia MCP Inspector

1. Installa npm se necessario.
2. Eseguite il seguente comando per avviare MCP Inspector:

```
npx @modelcontextprotocol/inspector
```

Configura la connessione

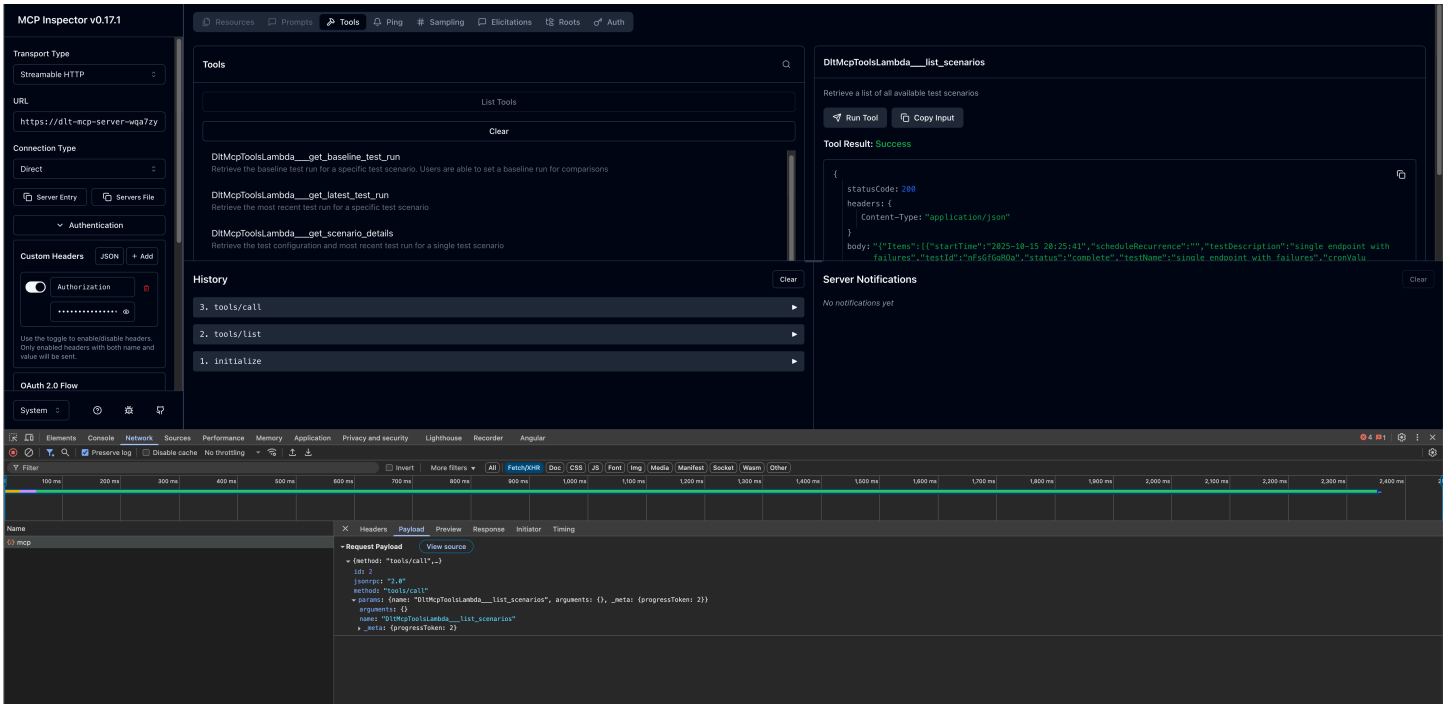
1. Nell'interfaccia MCP Inspector, inserite l'URL dell'endpoint del server MCP.

2. Aggiungi un'intestazione di autorizzazione con il tuo token di accesso.
3. Fate clic su Connect per stabilire la connessione.

Invoca strumenti

Una volta connesso, puoi testare gli strumenti MCP disponibili:

1. Sfoglia l'elenco degli strumenti disponibili nel pannello di sinistra.
2. Seleziona uno strumento (ad esempio,`list_scenarios`).
3. Fornite tutti i parametri richiesti.
4. Fate clic su Invoke per eseguire lo strumento e visualizzare la risposta.



Fase 3: Configurazione dei client di sviluppo AI

Dopo aver verificato la connessione al server MCP con MCP Inspector, puoi configurare il tuo client di sviluppo AI preferito.

CLI di Amazon Q

La CLI di Amazon Q fornisce l'accesso da riga di comando allo sviluppo assistito dall'intelligenza artificiale con l'integrazione di server MCP.

Fasi di configurazione

1. Modifica il file `mcp.json` di configurazione. Per ulteriori informazioni sulla posizione dei file di configurazione, consulta la sezione Configurazione dei server MCP remoti nella Amazon Q Developer User Guide.
2. Aggiungi la configurazione del tuo server DLT MCP:

```
{
  "mcpServers": {
    "dlt-mcp": {
      "type": "http",
      "url": "https://[api-id].execute-api.[region].amazonaws.com/[stage]/gateway/
backend-agent/sse/mcp",
```

```
      "headers": {
        "Authorization": "your_access_token_here"
      }
    }
  }
}
```

Verifica la configurazione

1. In un terminale, digita q per avviare Amazon Q CLI.
2. Digita `/mcp` per vedere tutti i server MCP disponibili.
3. Digita `/tools` per visualizzare gli strumenti disponibili forniti da `dlt-mcp` e altri server MCP configurati.
4. Verificate che l'inizializzazione sia `dlt-mcp` avvenuta correttamente.

Cline

Cline è un assistente di codifica AI che supporta l'integrazione del server MCP.

Fasi di configurazione

1. In Cline, vai a Gestisci server MCP > Configura > Configura server MCP.
2. Aggiorna il file: `cline_mcp_settings.json`

```
{
  "mcpServers": {
    "dlt-mcp": {
      "type": "streamableHttp",
      "url": "https://[api-id].execute-api.[region].amazonaws.com/[stage]/gateway/
backend-agent/sse/mcp",
      "headers": {
        "Authorization": "your_access_token_here"
      }
    }
  }
}
```

3. Salva il file di configurazione.
4. Riavvia Cline per applicare le modifiche.

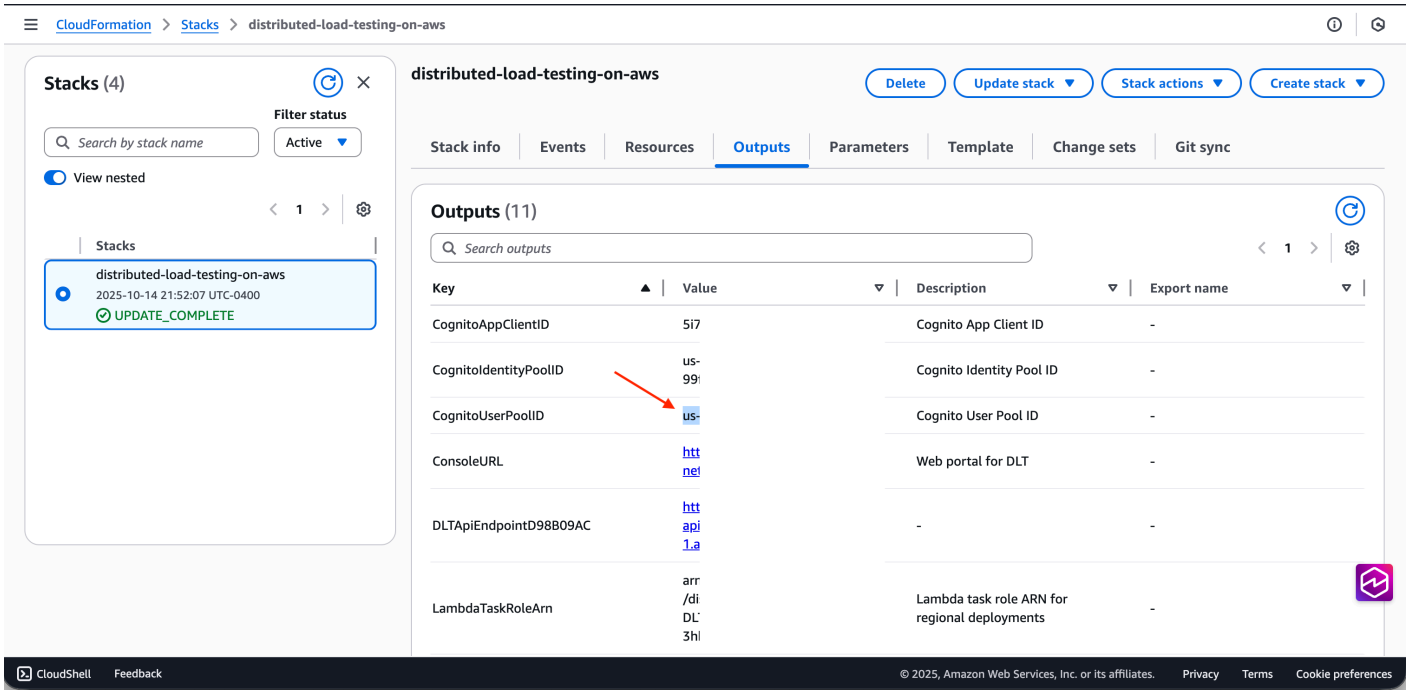
Amazon Q Suite

Amazon Q Suite offre una piattaforma di assistenza AI completa con supporto per le azioni del server MCP.

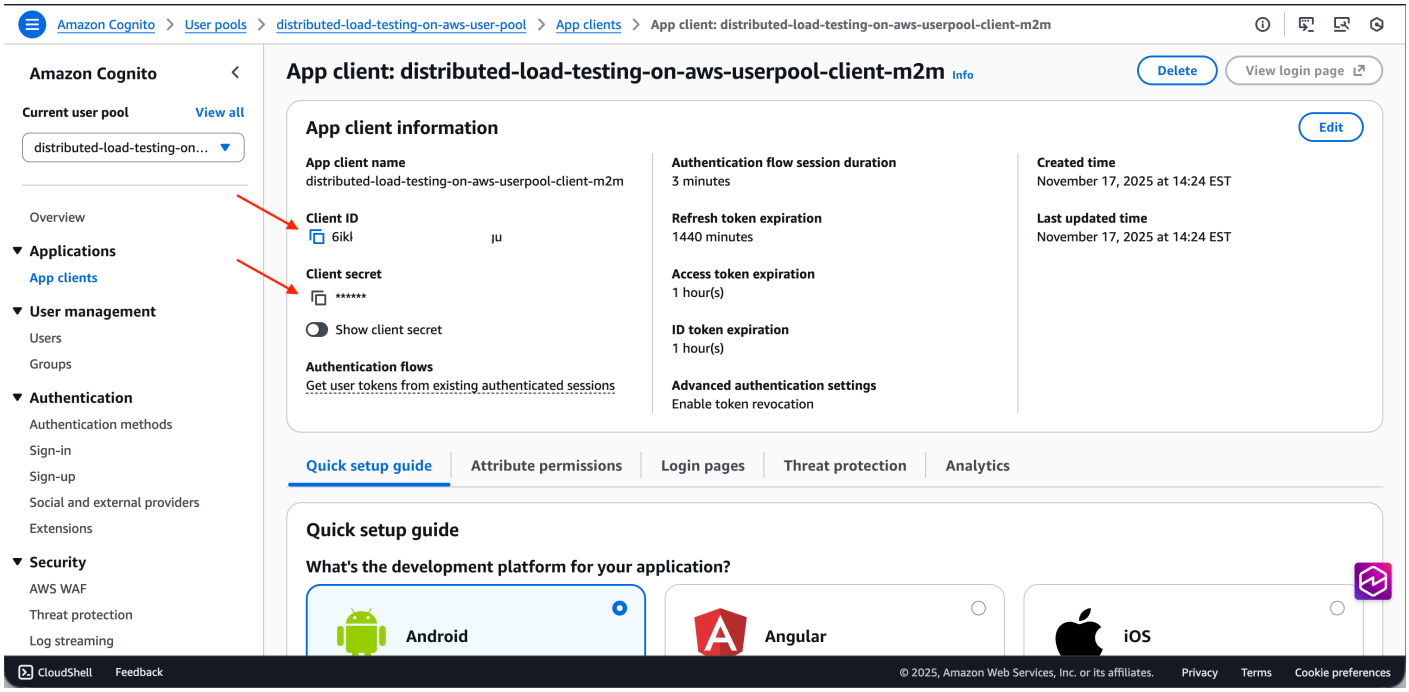
Prerequisiti

Prima di configurare il server MCP in Amazon Q Suite, devi recuperare OAuth le credenziali dal pool di utenti Cognito della distribuzione DLT:

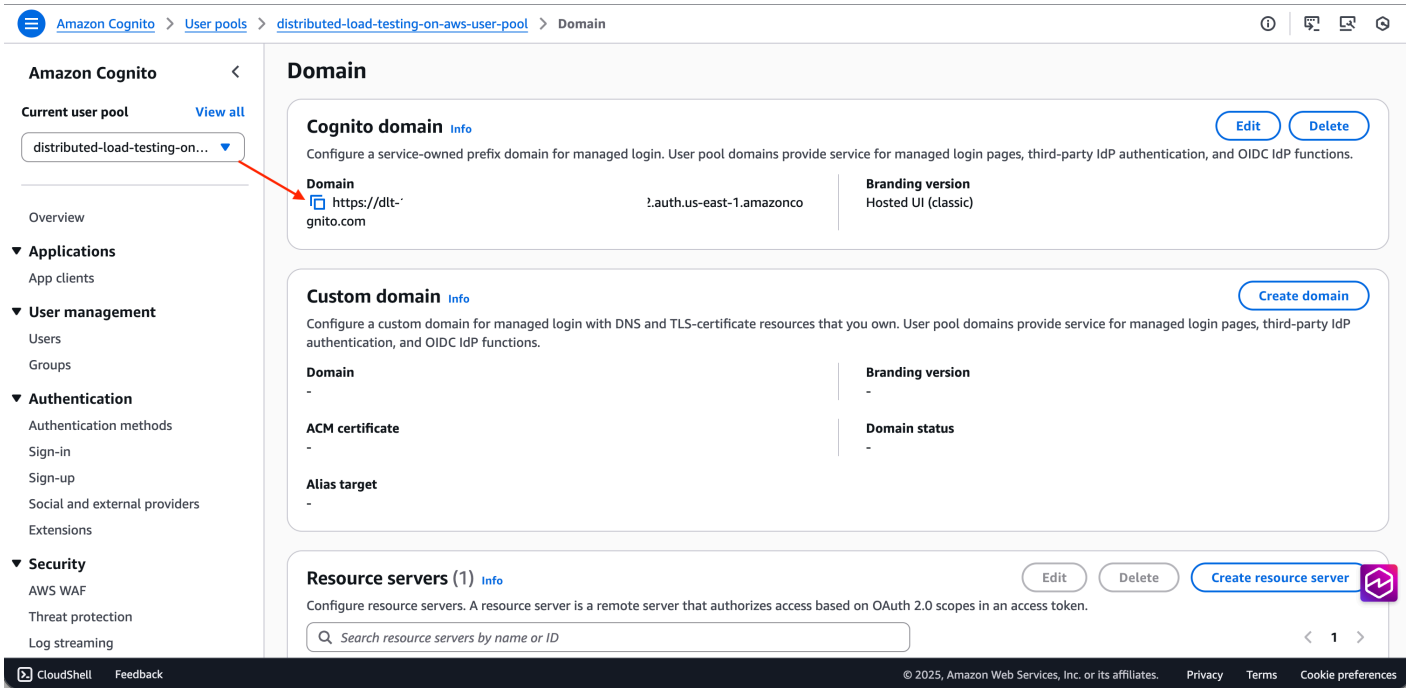
1. Accedi alla CloudFormation console AWS.
2. Seleziona lo stack Distributed Load Testing.
3. Nella scheda Output, individua e copia l'ID del pool di utenti Cognito associato alla distribuzione DLT.



4. Passa alla console di Amazon Cognito.
5. Seleziona il pool di utenti utilizzando l'ID del pool di utenti dagli CloudFormation output.
6. Nella barra di navigazione a sinistra, seleziona App integration > App client.



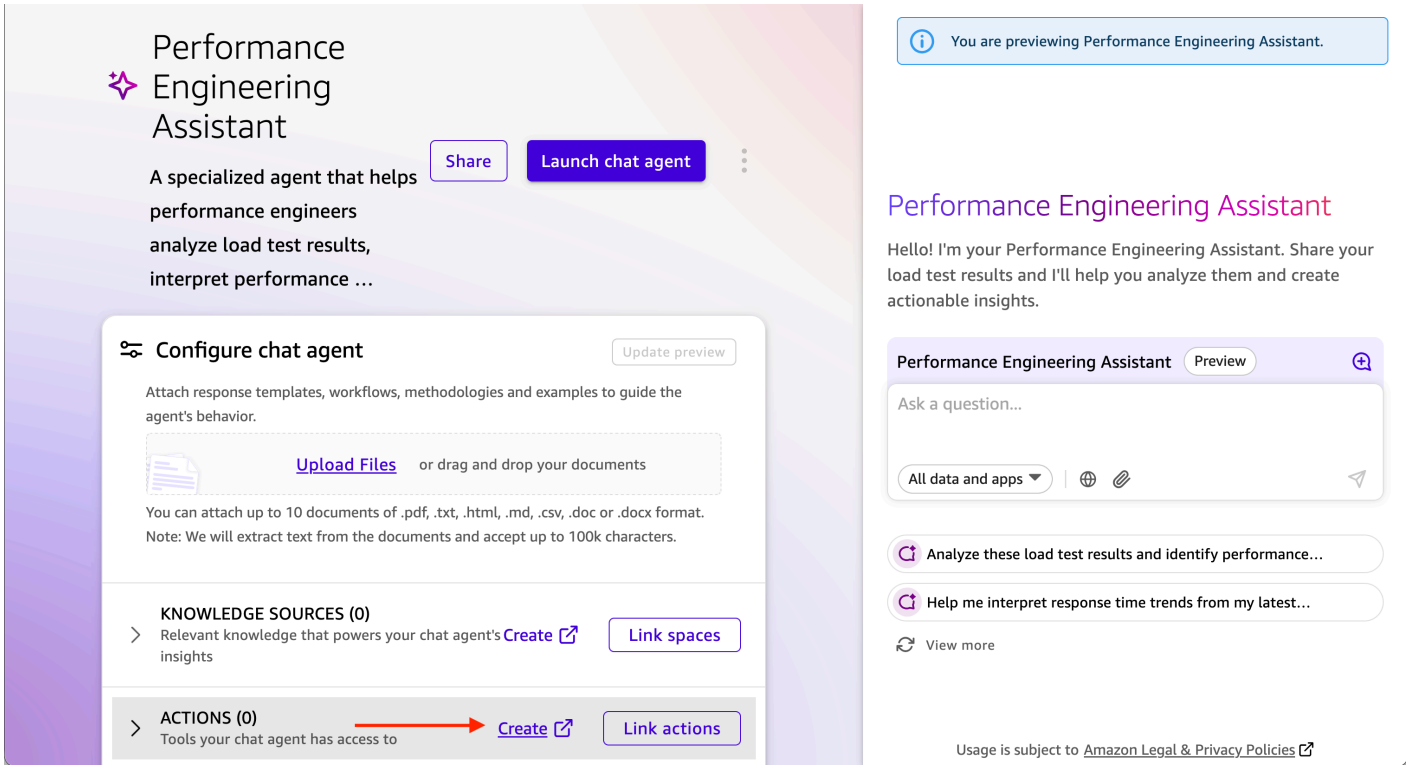
7. Individua il client dell'app con il nome che termina con `m2m` (machine-to-machine).
8. Copia l'ID client e il segreto del cliente.
9. Ottieni il dominio del pool di utenti dalla scheda Dominio.

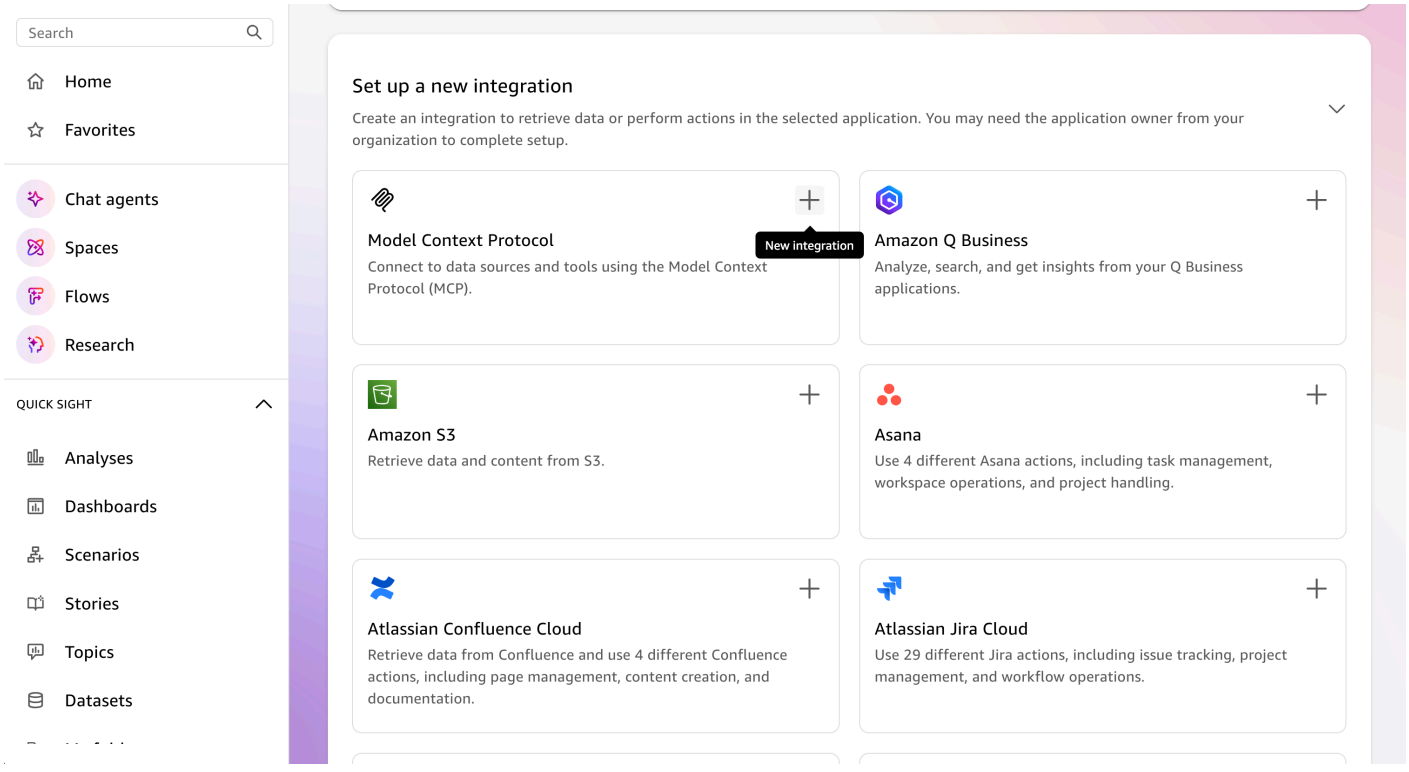


10. Costruisci l'URL dell'endpoint del token aggiungendolo `/oauth2/token` alla fine del dominio.

Fasi di configurazione

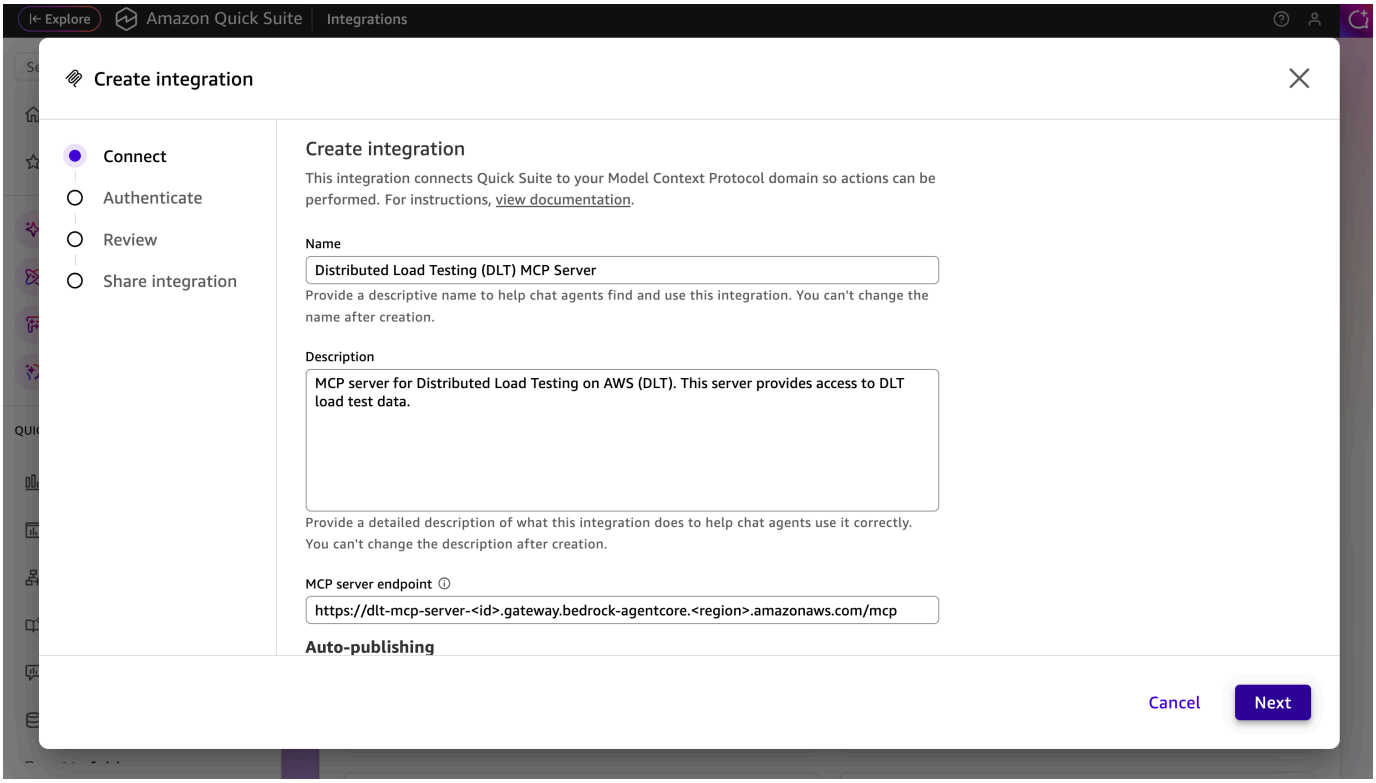
1. In Amazon Q Suite, crea un nuovo agente o seleziona un agente esistente.
2. Aggiungi un prompt all'agente che descrive come interagire con il server DLT MCP.
3. Aggiungi una nuova azione e seleziona l'azione del server MCP.



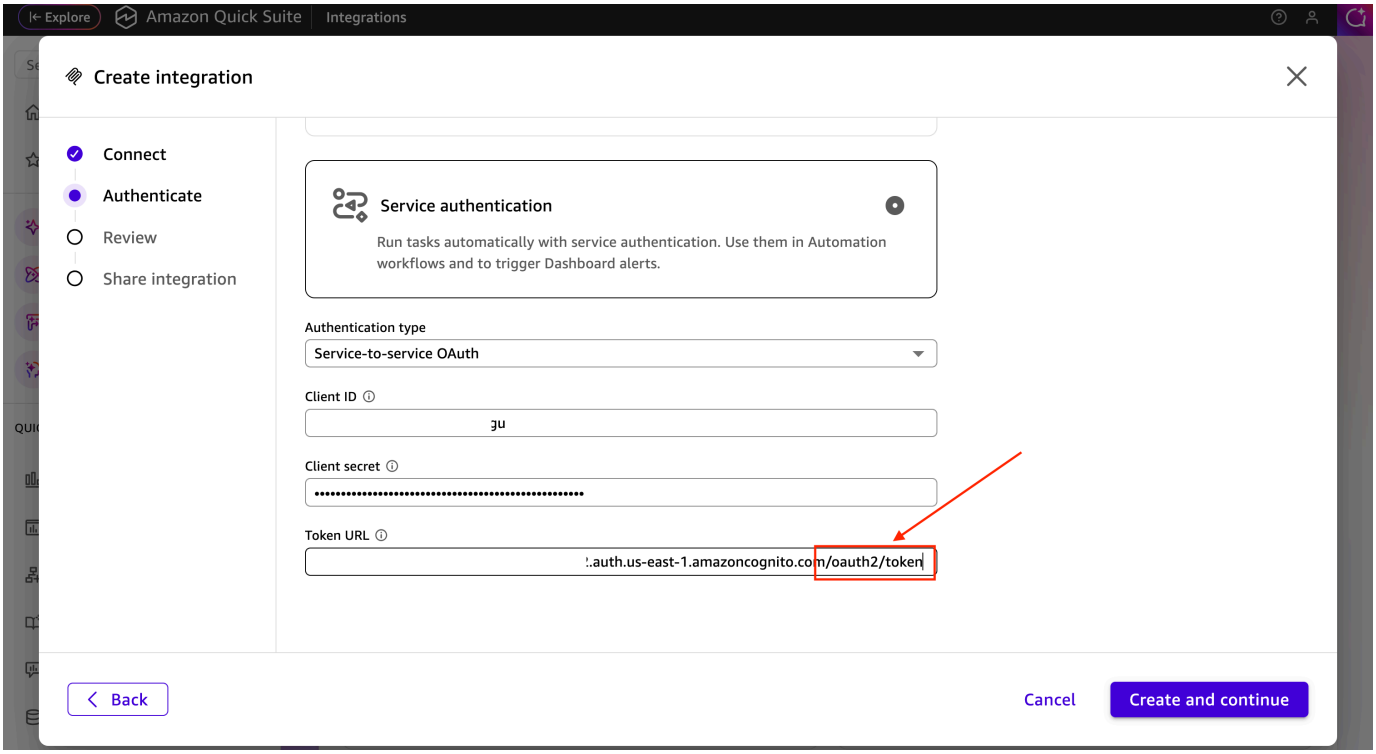


4. Configura i dettagli del server MCP:

- URL del server MCP: l'endpoint DLT MCP



- Tipo di autenticazione: autenticazione basata sui servizi
- Token Endpoint: l'URL dell'endpoint del token Cognito
- ID client: l'ID client del client dell'app m2m
- Client Secret: il client secret del client dell'app m2m



5. Salva la configurazione dell'azione del server MCP.
6. Aggiungi la nuova azione del server MCP al tuo agente.

Avvia e testa l'agente

1. Avvia l'agente in Amazon Q Suite.
2. Inizia una conversazione con l'agente utilizzando istruzioni in linguaggio naturale.
3. L'agente utilizzerà gli strumenti MCP per recuperare e analizzare i dati dei test di carico.

Prompt di esempio

Gli esempi seguenti mostrano come interagire con l'assistente AI per analizzare i dati dei test di carico tramite l'interfaccia MCP. Personalizza il test IDs, gli intervalli di date e i criteri per soddisfare le tue esigenze di test specifiche.

Per informazioni dettagliate sugli strumenti MCP disponibili e sui relativi parametri, fate riferimento alle specifiche degli strumenti MCP nella Developer Guide.

Semplice interrogazione dei risultati dei test

L'interazione del linguaggio naturale con il server MCP può essere tanto semplice quanto `Show me the load tests that have completed in the last 24 hours with their associated completion status` più descrittiva, ad esempio

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

Analisi interattiva delle prestazioni con divulgazione progressiva

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

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

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

Convalida della prontezza di produzione

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

1. Use list_scenarios to find recent test scenarios
2. For the most recent test scenario, use get_latest_test_run to get basic execution
 data
3. Use get_test_run with that test_run_id to get detailed response times, error rates,
 and throughput
4. Use get_scenario_details with the test_id to show me what load patterns and
 endpoints were tested
5. If I have a baseline, use get_baseline_test_run to compare current results with the
 baseline
```

```
6. Provide a clear go/no-go recommendation based on the performance data
7. If there are any concerns, use get_test_run_artifacts to help identify potential
 issues

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

Analisi delle tendenze delle prestazioni

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

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

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

Risoluzione dei problemi dei test falliti

```
Help me troubleshoot my failed load tests:

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

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

Guida per sviluppatori

Questa sezione fornisce il codice sorgente della soluzione e personalizzazioni aggiuntive.

Codice sorgente

Visita il nostro GitHub repository per scaricare i modelli e gli script per questa soluzione e per condividere le tue personalizzazioni con altri.

Maintenance (Manutenzione)

Questa soluzione utilizza immagini Docker con versioni fisse che corrispondono a ciascuna versione della soluzione. Per impostazione predefinita, gli aggiornamenti automatici sono disabilitati, offrendoti il controllo completo su quando e con quale versione gli aggiornamenti vengono applicati alla tua distribuzione. Il team di soluzioni AWS utilizza Amazon ECR Enhanced Scanning per rilevare vulnerabilità ed esposizioni comuni (CVEs) nell'immagine di base e nei pacchetti installati. Quando possibile, il team pubblica immagini patchate con lo stesso tag di versione da risolvere CVEs senza compromettere la compatibilità con la versione della soluzione rilasciata.

Quando le immagini vengono applicate alla stessa versione secondaria, il tag stable viene aggiornato automaticamente e nel formato viene creato un tag di immagine aggiuntivo. `<solution-version>_<date-of-fix>` Se viene rilasciata una versione principale o secondaria, è necessario eseguire un aggiornamento completo dello stack per ottenere la versione più recente dell'immagine, poiché il tag stable viene incrementato in modo da corrispondere alla versione della soluzione.

Se attivi gli aggiornamenti automatici durante la distribuzione, le modifiche all'immagine, incluse le patch CVE e le correzioni di bug minori, vengono applicate automaticamente fino all'ultima versione secondaria corrispondente.

Versioni

Per impostazione predefinita, questa soluzione viene implementata con gli aggiornamenti automatici disattivati. Ciò significa che la versione dell'immagine del contenitore è bloccata sulla versione specifica corrispondente alla versione della soluzione distribuita, offrendoti il pieno controllo sugli aggiornamenti della versione.

Se scegli di abilitare gli aggiornamenti automatici selezionando Sì durante la CloudFormation distribuzione, la soluzione riceverà automaticamente patch di sicurezza e correzioni di bug minori e ininterrotte fino all'ultima versione secondaria corrispondente. Ad esempio, se distribuisci la versione 4.0.0 con gli aggiornamenti automatici abilitati, riceverai aggiornamenti fino alla 4.0.x, ma non alla 4.1.0 o versioni successive.

Per controllare manualmente la versione dell'immagine del contenitore, puoi modificare la definizione dell'attività per specificare una particolare versione dell'immagine utilizzando il formato della versione con tag. Ciò consente di associare una versione specifica dell'immagine indipendentemente dall'impostazione degli aggiornamenti automatici.

Personalizzazione dell'immagine del contenitore

Questa soluzione utilizza un repository di immagini Amazon Elastic Container Registry (Amazon ECR) pubblico gestito da AWS per archiviare l'immagine utilizzata per eseguire i test configurati. Se desideri personalizzare l'immagine del contenitore, puoi ricostruirla e inserirla in un repository di immagini ECR nel tuo account AWS.

Se desideri personalizzare questa soluzione, puoi utilizzare l'immagine del contenitore predefinita o modificare questo contenitore in base alle tue esigenze. Se personalizzi la soluzione, utilizza il seguente esempio di codice per dichiarare le variabili di ambiente prima di creare la soluzione personalizzata.

```
#!/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
```

Se scegli di personalizzare l'immagine del contenitore, puoi ospitarla in un repository di immagini privato o in un repository di immagini pubblico nel tuo account AWS. Le risorse delle immagini si trovano nella `deployment/ecr/distributed-load-testing-on-aws-load-tester` directory, situata nella base di codice.

È possibile creare e inviare l'immagine alla destinazione dell'host.

- Per gli archivi e le immagini private di Amazon ECR, consulta i repository privati e le immagini private di Amazon ECR nella Guida per l'utente di Amazon ECR.
- Per gli archivi e le immagini pubbliche di Amazon ECR, consulta gli archivi pubblici e le immagini pubbliche di Amazon ECR nella Amazon ECR Public User Guide.

Dopo aver creato la tua immagine, puoi dichiarare le seguenti variabili di ambiente prima di creare la tua soluzione personalizzata.

```
#!/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
```

L'esempio seguente mostra il file contenitore.

```
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 procps 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/ecslistener.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"]
```

Oltre a un file contenitore, la directory contiene il seguente script bash che scarica la configurazione di test da Amazon S3 prima di eseguire Taurus/Blazemeter il programma.

```
#!/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
    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
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
```

Oltre al Dockerfile e allo script bash, nella directory sono inclusi anche due script Python. Ogni attività esegue uno script Python dall'interno dello script bash. Le attività di lavoro eseguono lo `ecslistener.py` script, mentre l'attività principale eseguirà lo script. `ecscontroller.py` Lo `ecslistener.py` script crea un socket sulla porta 50000 e attende un messaggio. Lo

`ecscontroller.py` script si connette al socket e invia il messaggio di test di avvio alle attività del lavoratore, che consente loro di avviarsi contemporaneamente.

API di test di carico distribuita

Questa soluzione di test di carico consente di esporre i dati dei risultati del test in modo sicuro. L'API funge da «porta d'ingresso» per l'accesso ai dati di test archiviati in Amazon DynamoDB. Puoi anche utilizzare il APIs per accedere a qualsiasi funzionalità estesa incorporata nella soluzione.

Questa soluzione utilizza un pool di utenti Amazon Cognito integrato con Amazon API Gateway per l'identificazione e l'autorizzazione. Quando un pool di utenti viene utilizzato con l'API, i client possono chiamare i metodi attivati dal pool di utenti solo dopo aver fornito un token di identità valido.

Per ulteriori informazioni sull'esecuzione dei test direttamente tramite l'API, consulta Signing Requests nella documentazione di riferimento dell'API REST di Amazon API Gateway.

Le seguenti operazioni sono disponibili nell'API della soluzione.

> **Note**
>
> Per ulteriori informazioni `testScenario` e altri parametri, consulta gli scenari e gli esempi di payload nel GitHub repository.

Informazioni sullo stack

- OTTIENI /stack-info

Scenari

- GET /scenarios
- POST /scenari
- OPZIONI/scenari
- OTTIENI /scenarios/ {testID}
- PUBBLICA /scenarios/ {testID}
- ELIMINA /scenarios/ {testID}
- OPZIONI /scenarios/ {testID}

Esecuzioni di test

- GET /scenarios/ {testID} /testruns
- OTTIENI /scenarios/ {testID} /testruns/ {} testRunId
- ELIMINA /scenarios/ {testID} /testruns/ {testRunId}

Linea di base

- GET /scenarios/ {testID} /baseline
- INSERISCI /scenarios/ {testID} /baseline
- ELIMINA /scenarios/ {testID} /baseline

Attività

- OTTIENI /tasks
- OPZIONI /tasks

Regioni

- GET /regions
- OPZIONI/regioni

OTTIENI /stack-info

Description

L'`GET /stack-info`operazione recupera informazioni sullo stack distribuito, tra cui l'ora di creazione, la regione e la versione. Questo endpoint viene utilizzato dal front-end.

Risposta

200 - Successo

Nome	Description
`created_time`	Timestamp ISO 8601 al momento della creazione dello stack (ad esempio,) `2025-09-09T19:40:22Z`
`region`	Regione AWS in cui viene distribuito lo stack (ad esempio,) `us-east-1`
`version`	Versione della soluzione distribuita (ad esempio,) `v4.0.0`

Risposte di errore

- `403`- Proibito: autorizzazioni insufficienti per accedere alle informazioni sullo stack
- `404`- Non trovato: informazioni sullo stack non disponibili
- `500`- Errore interno del server

GET /scenarios

Description

L'`GET /scenarios`operazione consente di recuperare un elenco di scenari di test.

Risposta

Nome	Description
`data`	Un elenco di scenari che include l'ID, il nome, la descrizione, lo stato, il tempo di esecuzione, i tag, le esecuzioni totali e l'ultima esecuzione per ogni test

POST /scenari

Description

L'`POST /scenarios`operazione consente di creare o pianificare uno scenario di test.

Corpo della richiesta

Nome	Description
`testName`	Il nome del test
`testDescription`	La descrizione del test
`testTaskConfigs`	Un oggetto che specifica `concurrency` (il numero di esecuzioni parallele), `taskCount` (il numero di attività necessarie `region` per eseguire un test) e lo scenario
`testScenario`	La definizione del test che include concorrenza, tempo di test, host e metodo per il test
`testType`	Il tipo di test (ad esempio`simple`,`jmeter`)
`fileType`	Il tipo di file da caricare (ad esempio`none`,`script`,`zip`)
`tags`	Una serie di stringhe per la categorizzazione dei test. Campo opzionale con una lunghezza massima di 5 (ad esempio,) `["blue", "3.0", "critical"]`
`scheduleDate`	La data di esecuzione di un test. Fornito solo se si pianifica un test (ad esempio,`2021-02-28`)
`scheduleTime`	Il tempo necessario per eseguire un test. Fornito solo se si pianifica un test (ad esempio,`21:07`)

Nome	Description
`scheduleStep`	Fase del processo di pianificazione. Fornito solo se si pianifica un test ricorrente. (I passaggi disponibili includono e) `create` `start`
`cronvalue`	Il valore cron per personalizzare la pianifica zione ricorrente. Se usato, ometti ScheduleD ate e ScheduleTime.
`cronExpiryDate`	Data obbligatoria in modo che il cron scada e non venga eseguito all'infinito.
`recurrence`	La ricorrenza di un test programmato. Fornito solo se si pianifica un test ricorrente (ad esempio,, `daily` o) `weekly` `biweekly` `monthly`

Risposta

Nome	Description
`testId`	L'ID univoco del test
`testName`	Il nome del test
`status`	Lo stato del test

OPZIONI/scenari

Description

L'`OPTIONS /scenarios`operazione fornisce una risposta alla richiesta con le intestazioni di risposta CORS corrette.

Risposta

Nome	Description
`testId`	L'ID univoco del test
`testName`	Il nome del test
`status`	Lo stato del test

GET /scenarios/ {testID}

Description

L'`GET /scenarios/{testId}`operazione consente di recuperare i dettagli di uno scenario di test specifico.

Parametri della richiesta

`testId`

- L'ID univoco del test

  Tipo: stringa

  Obbligatorio: sì

`latest`

- Parametro di query per restituire solo l'ultima esecuzione del test. L'impostazione predefinita è `true`

  Tipo: Booleano

  Obbligatorio: no

`history`

- Parametro di interrogazione per includere la cronologia dei test eseguiti nella risposta. Il valore predefinito è “`true`”. Imposta su `false` per escludere la cronologia

  Tipo: Booleano

Obbligatorio: no

Risposta

Nome	Description
`testId`	L'ID univoco del test
`testName`	Il nome del test
`testDescription`	La descrizione del test
`testType`	Il tipo di test che viene eseguito (ad esempio`simple`,`jmeter`)
`fileType`	Il tipo di file che viene caricato (ad esempio`none`,`script`,`zip`)
`tags`	Una serie di stringhe per la categorizzazione dei test
`status`	Lo stato del test
`startTime`	L'ora e la data di inizio dell'ultimo test
`endTime`	L'ora e la data in cui è terminato l'ultimo test
`testScenario`	La definizione del test che include concorrenza, ora del test, host e metodo per il test
`taskCount`	Il numero di attività necessarie per eseguire il test
`taskIds`	Un elenco di attività IDs per l'esecuzione dei test
`results`	I risultati finali del test

Nome	Description
`history`	Un elenco dei risultati finali dei test precedenti (escluso quando`history=false`)
`totalRuns`	Il numero totale di test eseguiti per questo scenario
`lastRun`	Il timestamp dell'ultima esecuzione del test
`errorReason`	Un messaggio di errore generato quando si verifica un errore
`nextRun`	La prossima esecuzione pianificata (ad esempio,`2017-04-22 17:18:00`)
`scheduleRecurrence`	La ricorrenza del test (ad esempio,,, `daily`,`weekly`,`biweekly`) `monthly`

POST /scenarios/ {testID}

Description

L'`POST /scenarios/{testId}`operazione consente di annullare uno scenario di test specifico.

Parametro di richiesta

`testId`

- L'ID univoco del test

  Tipo: stringa

  Obbligatorio: sì

Risposta

Nome	Description
`status`	Lo stato del test

DELETE /scenarios/ {testID}

Description

L'`DELETE /scenarios/{testId}`operazione consente di eliminare tutti i dati relativi a uno scenario di test specifico.

Parametro di richiesta

`testId`

- L'ID univoco del test

  Tipo: stringa

  Obbligatorio: sì

Risposta

Nome	Description
`status`	Lo stato del test

OPZIONI /scenarios/ {testID}

Description

L'`OPTIONS /scenarios/{testId}`operazione fornisce una risposta alla richiesta con le intestazioni di risposta CORS corrette.

Risposta

Nome	Description
`testId`	L'ID univoco del test
`testName`	Il nome del test
`testDescription`	La descrizione del test
`testType`	Il tipo di test che viene eseguito (ad esempio`simple`,`jmeter`)
`fileType`	Il tipo di file che viene caricato (ad esempio`none`,`script`,`zip`)
`status`	Lo stato del test
`startTime`	L'ora e la data di inizio dell'ultimo test
`endTime`	L'ora e la data in cui è terminato l'ultimo test
`testScenario`	La definizione del test che include concorrenza, ora del test, host e metodo per il test
`taskCount`	Il numero di attività necessarie per eseguire il test
`taskIds`	Un elenco di attività IDs per l'esecuzione dei test
`results`	I risultati finali del test
`history`	Un elenco dei risultati finali dei test precedenti
`errorReason`	Un messaggio di errore generato quando si verifica un errore

GET /scenarios/ {testID} /testruns

Description

L'`GET /scenarios/{testId}/testruns`operazione recupera l'esecuzione del test per uno scenario di test specifico, IDs facoltativamente filtrato per intervallo di tempo. When`latest=true`, restituisce solo la singola esecuzione del test più recente.

Parametri della richiesta

`testId`

- L'ID dello scenario di test

  Tipo: stringa

  Obbligatorio: sì

`latest`

- Restituisce solo l'ID di esecuzione del test più recente

  Tipo: Booleano

  Impostazione predefinita: `false`

  Obbligatorio: no

`start_timestamp`

- Timestamp ISO 8601 da cui filtrare le esecuzioni di test (incluso). Ad esempio, `2024-01-01T00:00:00Z`

  Tipo: Stringa (formato data-ora)

  Obbligatorio: no

`end_timestamp`

- Il timestamp ISO 8601 per filtrare i test viene eseguito fino a (incluso). Ad esempio, `2024-12-31T23:59:59Z`

  Tipo: Stringa (formato data-ora)

  Obbligatorio: no

`limit`

- Numero massimo di esecuzioni di test da restituire (ignorato quando) `latest=true`

  Tipo: numero intero (minimo: 1, massimo: 100)

  Impostazione predefinita: `20`

  Obbligatorio: no

`next_token`

- Token di impaginazione dalla risposta precedente alla pagina successiva

  ▬Tipo: stringa

  Obbligatorio: no

Risposta

200 - Successo

Nome	Description
`testRuns`	Serie di oggetti di esecuzione del test, ciascuno contenente `testRunId` (stringa) e `startTime` (data-ora ISO 8601)
`pagination`	Oggetto contenente `limit` (numero intero) e `next_token` (stringa o null). Il token è nullo se non ci sono altri risultati

Risposte di errore

- `400`- Formato o parametri del timestamp non validi
- `404`- Scenario di test non trovato
- `500`- Errore interno del server

Esempio di utilizzo

- Solo l'ultimo test eseguito: `GET /scenarios/test123/testruns?latest=true`
- Ultimo entro l'intervallo di tempo: `GET /scenarios/test123/testruns?latest=true&start_timestamp=2024-01-01T00:00:00Z`
- Richiesta di pagina successiva: `GET /scenarios/test123/testruns?limit=20&next_token=eyJ0ZXN0SWQiOiJzZVFVeTEyTEtMIiwic3RhcnRUaW1lIjoiMjAyNC0wMS`

GET /scenarios/ {testID} /testruns/ {} testRunId

Description

L'`GET /scenarios/{testId}/testruns/{testRunId}`operazione recupera risultati e metriche completi per una specifica esecuzione di test. Facoltativamente, ometti i risultati della cronologia con `history=false` per una risposta più rapida.

Parametri della richiesta

`testId`

- L'ID dello scenario di test

  Tipo: stringa

  Obbligatorio: sì

`testRunId`

- L'ID specifico dell'esecuzione del test

  Tipo: stringa

  Obbligatorio: sì

`history`

- Include l'array di cronologia in risposta. Imposta su `false` per omettere la cronologia per una risposta più rapida

  Tipo: Booleano

  Impostazione predefinita: `true`

Obbligatorio: no

Risposta

200 - Successo

Nome	Description
`testId`	L'ID univoco del test (ad esempio,`seQUy12LKL`)
`testRunId`	L'ID specifico dell'esecuzione del test (ad esempio,`2DEwHItEne`)
`testDescription`	Descrizione del test di carico
`testType`	Il tipo di test (ad esempio`simple`,`jmeter`)
`status`	Lo stato dell'esecuzione del test: `completerunning`,`failed`, o `cancelled`
`startTime`	L'ora e la data di inizio del test (ad esempio,`2025-09-09 21:01:00`)
`endTime`	L'ora e la data in cui è terminato il test (ad esempio,`2025-09-09 21:18:29`)
`succPercent`	Percentuale di successo (ad esempio,`100.00`)
`testTaskConfigs`	Matrice di oggetti di configurazione delle attività contenenti `regiontaskCount` , e `concurrency`
`completeTasks`	Le regioni di mappatura degli oggetti in base al numero di attività completate
`results`	Oggetto contenente metriche dettagliate tra cui `avg_lt` (latenza media), percentili (`p0_0`,,`p50_0`,`p90_0`,`p95_0`,`p100_0`)

Nome	Description
	p99_0p99_9, avg_rt (tempo di risposta medio), avg_ct (tempo medio di connessio ne), stdev_rt (tempo di risposta in deviazion e standard),concurrency , (numero di successi)throughput , succ (conteggio errori),, bytes testDuration metricS3L ocation , fail rc (array di codici di risposta) e array labels
testScenario	Oggetto contenente la configurazione di test con execution e le proprietà reporting scenarios
history	Matrice di risultati storici dei test (escluso quandohistory=false)

Risposte di errore

- 400- TestID non valido o testRunId
- 404- Esecuzione del test non trovata
- 500- Errore interno del server

ELIMINA /scenarios/ {testID} /testruns/ {} testRunId

Description

L'DELETE /scenarios/{testId}/testruns/{testRunId}operazione elimina tutti i dati e gli artefatti relativi a una specifica esecuzione di test. I dati di esecuzione del test vengono rimossi da DynamoDB, mentre i dati di test effettivi in S3 rimangono invariati.

Parametri della richiesta

testId

- L'ID dello scenario di test

Tipo: stringa

Obbligatorio: sì

`testRunId`

- L'ID di esecuzione del test specifico da eliminare

  Tipo: stringa

  Obbligatorio: sì

Risposta

204 - Successo

Esecuzione del test eliminata con successo (nessun contenuto restituito)

Risposte di errore

- `400`- TestID non valido o testRunId
- `403`- Proibito: autorizzazioni insufficienti per eliminare l'esecuzione del test
- `404`- Esecuzione del test non trovata
- `409`- Conflitto: l'esecuzione del test è attualmente in esecuzione e non può essere eliminata
- `500`- Errore interno del server

GET /scenarios/ {testID} /baseline

Description

L'`GET /scenarios/{testId}/baseline`operazione recupera il risultato del test di base designato per uno scenario. Restituisce l'ID di esecuzione del test di base o i risultati di base completi a seconda del parametro. `data`

Parametri della richiesta

`testId`

- L'ID dello scenario di test

  Tipo: stringa

Obbligatorio: sì

`data`

- Restituisce i dati completi di esecuzione del test di base se`true`, altrimenti solo testRunId

  Tipo: Booleano

  Impostazione predefinita: `false`

  Obbligatorio: no

Risposta

200 - Successo

Quando `data=false` (impostazione predefinita):

Nome	Description
`testId`	L'ID dello scenario di test (ad esempio,`seQUy12LKL`)
`baselineTestRunId`	L'ID di esecuzione del test di base (ad esempio,`2DEwHItEne`)

Quando`data=true`:

Nome	Description
`testId`	L'ID dello scenario di test (ad esempio,`seQUy12LKL`)
`baselineTestRunId`	L'ID di esecuzione del test di base (ad esempio,`2DEwHItEne`)
`baselineData`	Oggetto completo dei risultati dell'esecuzione del test (stessa `GET /scenarios/{testId}/testruns/{testRunId}` struttura di)

Risposte di errore

- `400`- Parametro TestID non valido
- `404`- Scenario di test non trovato o nessuna linea di base impostata
- `500`- Errore interno del server

PUT /scenarios/ {testID} /baseline

Description

L'`PUT /scenarios/{testId}/baseline`operazione designa un test specifico come base per il confronto delle prestazioni. È possibile impostare una sola linea di base per scenario.

Parametri della richiesta

`testId`

- L'ID dello scenario di test

  Tipo: stringa

  Obbligatorio: sì

Corpo della richiesta

Nome	Description
`testRunId`	L'ID di esecuzione del test da impostare come riferimento (ad esempio,`2DEwHItEne`)

Risposta

200 - Successo

Nome	Description
`message`	Messaggio di conferma (ad esempio,`Baseline set successfully`)

Nome	Description
`testId`	L'ID dello scenario di test (ad esempio,`seQUy12LKL`)
`baselineTestRunId`	L'ID di esecuzione del test di base che è stato impostato (ad esempio,`2DEwHItEne`)

Risposte di errore

- `400`- TestID non valido o testRunId
- `404`- Scenario di test o esecuzione del test non trovati
- `409`- Conflitto: l'esecuzione del test non può essere impostata come base (ad esempio, test fallito)
- `500`- Errore interno del server

DELETE /scenarios/ {testID} /baseline

Description

L'`DELETE /scenarios/{testId}/baseline`operazione cancella il valore di base per uno scenario impostandolo su una stringa vuota.

Parametri della richiesta

`testId`

- L'ID dello scenario di test

  Tipo: stringa

  Obbligatorio: sì

Risposta

204 - Successo

Baseline cancellata con successo (nessun contenuto restituito)

Risposte di errore

- `400`- TestID non valido
- `500`- Errore interno del server

GET /tasks

Description

L'`GET /tasks`operazione consente di recuperare un elenco di attività Amazon Elastic Container Service (Amazon ECS) in esecuzione.

Risposta

Nome	Description
`tasks`	Un elenco di attività IDs per l'esecuzione dei test

OPZIONI/task

Description

L'operazione `OPTIONS /tasks` tasks fornisce una risposta alla richiesta con le intestazioni di risposta CORS corrette.

Risposta

Nome	Description
`taskIds`	Un elenco di attività IDs per l'esecuzione dei test

GET /regions

Description

L'`GET /regions`operazione consente di recuperare le informazioni sulle risorse regionali necessarie per eseguire un test in quella regione.

Risposta

Nome	Description
`testId`	L'ID della regione
`ecsCloudWatchLogGroup`	Il nome del gruppo di CloudWatch log di Amazon per le attività di Amazon Fargate nella regione
`region`	La regione in cui esistono le risorse della tabella
`subnetA`	L'ID di una delle sottoreti della regione
`subnetB`	L'ID di una delle sottoreti nella regione
`taskCluster`	Il nome del cluster AWS Fargate nella regione
`taskDefinition`	L'ARN della definizione dell'attività nella regione
`taskImage`	Il nome dell'immagine dell'attività nella regione
`taskSecurityGroup`	L'ID del gruppo di sicurezza nella regione

OPZIONI/regioni

Description

L'`OPTIONS /regions`operazione fornisce una risposta alla richiesta con le intestazioni di risposta CORS corrette.

Risposta

Nome	Description
`testId`	L'ID della regione

Nome	Description
`ecsCloudWatchLogGroup`	Il nome del gruppo di CloudWatch log di Amazon per le attività di Amazon Fargate nella regione
`region`	La regione in cui esistono le risorse della tabella
`subnetA`	L'ID di una delle sottoreti della regione
`subnetB`	L'ID di una delle sottoreti nella regione
`taskCluster`	Il nome del cluster AWS Fargate nella regione
`taskDefinition`	L'ARN della definizione dell'attività nella regione
`taskImage`	Il nome dell'immagine dell'attività nella regione
`taskSecurityGroup`	L'ID del gruppo di sicurezza nella regione

Aumenta le risorse del contenitore

Per aumentare il numero di utenti virtuali simultanei (concorrenza) che i test di carico possono simulare, è necessario aumentare le risorse di CPU e memoria allocate per ogni attività di Amazon ECS. Ciò comporta la creazione di una nuova revisione della definizione di attività con limiti di risorse più elevati, quindi l'aggiornamento della configurazione DynamoDB della soluzione per utilizzare la nuova definizione di attività per future esecuzioni di test.

Crea una nuova revisione della definizione delle attività

Segui questi passaggi per creare una nuova definizione di attività con maggiori risorse di CPU e memoria:

1. Accedi alla console Amazon Elastic Container Service.
2. Nel menu di navigazione a sinistra, seleziona Task Definitions.

3. Seleziona la casella di controllo accanto alla definizione dell'attività corrispondente a questa soluzione. Ad esempio, `[replaceable]` <stackName>`- EcsTaskDefinition -*<system-generated-random-Hash>*.
4. Scegliere Create new revision (Crea nuova revisione).
5. Nella pagina Crea nuova revisione, esegui le seguenti azioni:
   a. In Dimensioni dell'attività, modifica la memoria dell'attività e la CPU dell'attività con i valori desiderati. I valori più alti consentono un numero maggiore di utenti virtuali simultanei per attività.
   b. In Definizioni dei contenitori, esamina i limiti di memoria hard/soft. Se questo limite è inferiore alla memoria desiderata, scegli il contenitore.
   c. Nella finestra di dialogo Modifica contenitore, vai a Limiti di memoria e aggiorna il limite rigido in modo che corrisponda o sia inferiore all'allocazione della memoria dell'attività.
   d. Scegliere Aggiorna.
6. Nella pagina Crea nuova revisione, scegli Crea.
7. Dopo aver creato correttamente la definizione dell'attività, registrare l'ARN completo della definizione dell'attività, incluso il numero di versione. Ad esempio: `[replaceable]` <stackName>`- EcsTaskDefinition -*<system-generated-random-Hash>*: [sostituibile]. `<system-generated-versionNumber>`

Aggiornare la tabella DynamoDB

Dopo aver creato la nuova revisione della definizione delle attività, è necessario aggiornare la tabella DynamoDB della soluzione in modo che le future esecuzioni di test utilizzino la nuova definizione di attività. Ripeti questi passaggi per ogni regione AWS in cui desideri utilizzare la definizione di attività aggiornata:

1. Accedere alla console DynamoDB.
2. Dal riquadro di navigazione a sinistra, seleziona Esplora gli elementi in Tabelle.
3. Seleziona la tabella `scenarios-table` DynamoDB associata a questa soluzione. Ad esempio, `[replaceable]` <stackName>`- DLTTest RunnerStorage DLTScenarios Tabella-. *<system-generated-random-Hash>*
4. Seleziona l'elemento che corrisponde alla regione in cui hai creato la nuova revisione della definizione dell'attività. Ad esempio, `region-[replaceable]` <region-name>`.

5. Nell'editor degli elementi, individua l'attributo TaskDefinition e aggiorna il suo valore con l'ARN completo della definizione dell'attività registrato nella sezione precedente (incluso il numero di versione).
6. Scegli Save changes (Salva modifiche).

> **Note**
>
> La definizione dell'attività aggiornata verrà utilizzata solo per nuove esecuzioni di test. Tutti i test attualmente in esecuzione o pianificati continueranno a utilizzare la definizione di attività precedente.

Specifiche degli strumenti MCP

La soluzione Distributed Load Testing espone una serie di strumenti MCP che consentono agli agenti AI di interagire con gli scenari e i risultati dei test. Questi strumenti forniscono funzionalità astratte di alto livello che si allineano al modo in cui gli agenti di intelligenza artificiale elaborano le informazioni, consentendo loro di concentrarsi sull'analisi e sugli approfondimenti anziché su contratti API dettagliati.

> **Note**
>
> Tutti gli strumenti MCP forniscono accesso in sola lettura ai dati della soluzione. Nessuna modifica agli scenari o alle configurazioni di test è supportata tramite l'interfaccia MCP.

list_scenarios

Description

Lo `list_scenarios` strumento recupera un elenco di tutti gli scenari di test disponibili con metadati di base.

Endpoint

```
GET /scenarios
```

Parameters

Nessuno

Risposta

Nome	Description
`testId`	Identificatore univoco per lo scenario di test
`testName`	Nome dello scenario di test
`status`	Stato attuale dello scenario di test
`startTime`	Quando il test è stato creato o eseguito l'ultima volta
`testDescription`	Descrizione dello scenario di test

get_scenario_details

Description

Lo `get_scenario_details` strumento recupera la configurazione del test e l'esecuzione del test più recente per un singolo scenario di test.

Endpoint

```
GET /scenarios/<test_id>?history=false&results=false
```

Parametro di richiesta

`test_id`

- L'identificatore univoco per lo scenario di test

  Tipo: stringa

  Obbligatorio: sì

Risposta

Nome	Description
`testTaskConfigs`	Configurazione delle attività per ogni regione
`testScenario`	Definizione e parametri del test
`status`	Stato attuale del test
`startTime`	Timestamp di inizio del test
`endTime`	Timestamp di fine del test (se completato)

list_test_runs

Description

Lo `list_test_runs` strumento recupera un elenco di esecuzioni di test per uno scenario di test specifico, ordinate dalla più recente alla meno recente. Restituisce un massimo di 30 risultati.

Endpoint

```
GET /scenarios/<testid>/testruns/?limit=<limit>
```

or

```
GET /scenarios/<testid>/testruns/?
limit=30&start_date=<start_date>&end_date=<end_date>
```

Parametri della richiesta

`test_id`

- L'identificatore univoco per lo scenario di test

  Tipo: stringa

  Obbligatorio: sì

`limit`

- Numero massimo di esecuzioni di test da restituire

Tipo: numero intero

Impostazione predefinita: 20

Massimo: 30

Obbligatorio: no

`start_date`

- Il timestamp ISO 8601 per il filtraggio parte da una data specifica

  Tipo: Stringa (formato data-ora)

  Obbligatorio: no

`end_date`

- Il timestamp ISO 8601 da filtrare viene eseguito fino a una data specifica

  Tipo: Stringa (formato data-ora)

  Obbligatorio: no

Risposta

Nome	Description
`testRuns`	Serie di riepiloghi delle esecuzioni di test con metriche e percentili delle prestazioni per ogni esecuzione

get_test_run

Description

Lo `get_test_run` strumento recupera i risultati dettagliati per una singola esecuzione di test con suddivisioni regionali ed endpoint.

Endpoint

```
GET /scenarios/<testid>/testruns/<testrunid>
```

Parametri della richiesta

`test_id`

- L'identificatore univoco per lo scenario di test

  Tipo: stringa

  Obbligatorio: sì

`test_run_id`

- L'identificatore univoco per l'esecuzione specifica del test

  Tipo: stringa

  Obbligatorio: sì

Risposta

Nome	Description
`results`	Dati completi sull'esecuzione del test, tra cui la ripartizione dei risultati regionali, le metriche specifiche degli endpoint, i percentili di prestazioni (p50, p90, p95, p99), il numero di successi e fallimenti, i tempi di risposta e la latenza e la configurazione del test utilizzata per l'esecuzione

get_latest_test_run

Description

Lo `get_latest_test_run` strumento recupera il test eseguito più recentemente per uno scenario di test specifico.

Endpoint

```
GET /scenarios/<testid>/testruns/?limit=1
```

> **Note**
>
> I risultati vengono ordinati in base all'ora utilizzando un indice secondario globale (GSI), che garantisce la restituzione del test eseguito più recente.

Parametro di richiesta

`test_id`

- L'identificatore univoco per lo scenario di test

  Tipo: stringa

  Obbligatorio: sì

Risposta

Nome	Description
`results`	Dati di esecuzione del test più recenti con lo stesso formato di `get_test_run`

get_baseline_test_run

Description

`get_baseline_test_run`Lo strumento recupera l'esecuzione del test di base per uno scenario di test specifico. La baseline viene utilizzata per il confronto delle prestazioni.

Endpoint

```
GET /scenarios/<test_id>/baseline
```

Parametro di richiesta

`test_id`

- L'identificatore univoco per lo scenario di test

  Tipo: stringa

  Obbligatorio: sì

Risposta

Nome	Description
`baselineData`	Dati di base relativi all'esecuzione del test a scopo di confronto, incluse tutte le metriche e la configurazione relative all'esecuzione di base designata

get_test_run_artifacts

Description

Lo `get_test_run_artifacts` strumento recupera le informazioni sui bucket Amazon S3 per accedere agli artefatti dei test, inclusi log, file di errore e risultati.

Endpoint

`GET /scenarios/<testid>/testruns/<testrunid>`

Parametri della richiesta

`test_id`

- L'identificatore univoco per lo scenario di test

  Tipo: stringa

  Obbligatorio: sì

`test_run_id`

- L'identificatore univoco per l'esecuzione specifica del test

  Tipo: stringa

  Obbligatorio: sì

Risposta

Nome	Description
`bucketName`	Nome del bucket S3 in cui sono archiviati gli artefatti
`testRunPath`	Prefisso del percorso per l'attuale archiviazione degli artefatti (versione 4.0+)
`testScenarioPath`	Prefisso del percorso per l'archiviazione degli artefatti legacy (versione precedente alla 4.0)

> **Note**
>
> Tutti gli strumenti MCP sfruttano gli endpoint API esistenti. Non sono necessarie modifiche al sottostante per supportare la APIs funzionalità MCP.

Riferimento

Questa sezione include informazioni sulla raccolta dei dati, riferimenti a risorse correlate e un elenco di costruttori che hanno contribuito a questa soluzione.

Raccolta dei dati

Questa soluzione invia metriche operative ad AWS (i «Dati») sull'utilizzo di questa soluzione. Utilizziamo questi dati per comprendere meglio come i clienti utilizzano questa soluzione e i servizi e i prodotti correlati. La raccolta di questi dati da parte di AWS è soggetta all'Informativa sulla privacy di AWS.

Collaboratori

- Tom Nightingale
- Fernando Dingler
- Lee Beomseok
- Georg Lenz
- Erin McGill
- Dimitri López
- Kamyar Ziabari
- Bassem Wanis
- Garvit Singh
- Nikhil Reddy
- Simon Kroll
- Ahern Knox
- Ian Downard
- Owen Brady
- Jim Thario
- Thyag Ramachandran
- Yang Qin
- Jim Wang

Glossario

Questo glossario definisce gli acronimi e le abbreviazioni utilizzati nella Distributed Load Testing on AWS Implementation Guide.

Protocolli e formati tecnici

AGPL

Licenza pubblica generale Affero. Una licenza software open source utilizzata da K6.

"Hello, World!"

Interfaccia di programmazione delle applicazioni. Un insieme di protocolli e strumenti per creare applicazioni software e consentire la comunicazione tra diversi sistemi.

CLI

Interfaccia a riga di comando. Interfaccia testuale per interagire con software e sistemi operativi.

CORE

Condivisione di risorse tra le origini. Una funzionalità di sicurezza che consente o limita le applicazioni Web in esecuzione su un'origine di accedere a risorse da un'origine diversa.

CSV

Valori separati da virgole. Un formato di file utilizzato per memorizzare dati tabulari in testo semplice, comunemente usato per l'esportazione dei dati.

gRPC

Chiamata di procedura remota gRPC. Un framework open source ad alte prestazioni per chiamate di procedura remota.

HTTP

Protocollo di trasferimento ipertestuale. Il protocollo di base utilizzato per la trasmissione di dati sul World Wide Web.

HTTPS

HTTP sicuro. Un'estensione di HTTP che utilizza la crittografia per comunicazioni sicure su una rete.

JSON

JavaScript Notazione degli oggetti. Un formato di scambio dati leggero, facile da leggere e scrivere per gli esseri umani e facile da analizzare e generare per le macchine.

JWT

Token Web JSON. Un mezzo compatto e sicuro per gli URL per rappresentare le affermazioni da trasferire tra due parti per l'autenticazione e l'autorizzazione.

OAuth

Autorizzazione aperta. Uno standard aperto per la delega di accesso comunemente utilizzato per l'autenticazione e l'autorizzazione basate su token.

REST

Trasferimento statale rappresentativo. Uno stile architettonico per la progettazione di applicazioni in rete che utilizzano comunicazioni stateless e metodi HTTP standard.

SSE

Eventi inviati dal server. Una tecnologia server push che consente a un client di ricevere aggiornamenti automatici da un server tramite una connessione HTTP.

Interfaccia utente

Interfaccia utente. Gli elementi visivi e i controlli attraverso i quali gli utenti interagiscono con le applicazioni software.

URL

Uniform Resource Locator. L'indirizzo utilizzato per accedere alle risorse su Internet.

XML

Linguaggio di markup estensibile. Un linguaggio di markup che definisce le regole per la codifica dei documenti in un formato leggibile sia dall'uomo che dalla macchina.

Termini relativi ai test e ai database

FTP

Protocollo di trasferimento dei file. Un protocollo di rete standard utilizzato per il trasferimento di file tra un client e un server.

GSI

Indice secondario globale. Una funzionalità di DynamoDB che consente di interrogare i dati utilizzando una chiave alternativa.

JDBC

Connettività al database Java. Un'API Java per la connessione e l'esecuzione di query con i database.

JMS

Servizio messaggi Java. Un'API Java per l'invio di messaggi tra due o più client.

TPS

Transazioni al secondo. Misura del numero di transazioni che un sistema può elaborare in un secondo.

AWS e termini di sistema

ARN

Amazon Resource Name. Un identificatore univoco per le risorse AWS utilizzato per specificare le risorse tra i servizi AWS.

ISO

Organizzazione internazionale per la standardizzazione. Un'organizzazione indipendente e non governativa che sviluppa standard internazionali. A cui si fa riferimento in questa guida per il formato timestamp ISO 8601.

SLA

Accordo sul livello di servizio. Un impegno tra un fornitore di servizi e un cliente che definisce il livello di servizio previsto.

UUID

Identificatore universalmente unico. Un numero a 128 bit utilizzato per identificare in modo univoco le informazioni nei sistemi informatici.

VPCU

Unità di elaborazione centrale virtuale. Un processore virtuale assegnato a una macchina virtuale o a un contenitore, che rappresenta una parte della potenza di elaborazione della CPU fisica.

Termini del test di carico

simultaneità

Il numero di utenti virtuali simultanei per attività. Questo parametro controlla il numero di utenti simulati generato da ciascuna attività Fargate durante un test di carico.

stack regionale

Uno CloudFormation stack distribuito in una regione AWS per fornire un'infrastruttura di test per test di carico multiregione.

conteggio delle attività

Il numero di container Fargate (attività) avviati per eseguire uno scenario di test. Il carico totale generato è uguale al numero di attività moltiplicato per la concomitanza.

scenario di test

Un test di carico configurato che include il tipo di test, gli endpoint di destinazione, il numero di attività, la concorrenza, la durata e altri parametri.

Revisioni

Visita Changelog.md nel nostro GitHub repository per tenere traccia dei miglioramenti e delle correzioni specifici della versione.

Note

I clienti sono responsabili della propria valutazione indipendente delle informazioni contenute nel presente documento. Questo documento: (a) è solo a scopo informativo, (b) rappresenta le attuali offerte e pratiche di prodotti AWS, che sono soggette a modifiche senza preavviso, e (c) non crea alcun impegno o garanzia da parte di AWS e delle sue affiliate, fornitori o licenzianti. I prodotti o i servizi AWS sono forniti «così come sono» senza garanzie, dichiarazioni o condizioni di alcun tipo, esplicite o implicite. Le responsabilità e gli obblighi di AWS nei confronti dei propri clienti sono controllati da accordi AWS e questo documento non fa parte né modifica alcun accordo tra AWS e i suoi clienti.

Distributed Load Testing on AWS è concesso in licenza secondo i termini della versione 2.0 della licenza Apache, disponibile presso The Apache Software Foundation.

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.