구현 안내서

AWS의 분산 로드 테스트

AWS의 분산 로드 테스트: 구현 안내서

Table of Contents

대규모 소프트웨어 애플리케이션 테스트 자동화

게시 날짜: 2025년 12월

AWS의 분산 로드 테스트를 사용하면 애플리케이션을 릴리스하기 전에 소프트웨어 애플리케이션의 성능 테스트를 대규모로 자동화하여 병목 현상을 식별할 수 있습니다. 이 솔루션은 서버를 프로비저닝할 필요 없이 지속적으로 HTTP 요청을 생성하는 수천 명의 연결된 사용자를 시뮬레이션합니다.

이 솔루션은 AWS Fargate의 Amazon Elastic Container Service(Amazon ECS)를 활용하여 로드 테스트 시뮬레이션을 실행하는 컨테이너를 배포하고 다음 기능을 제공합니다.

- 독립적으로 실행되는 AWS Fargate 컨테이너에 Amazon ECS를 배포하여 애플리케이션의 로드 용량을 테스트합니다.
- 여러 AWS 리전에서 수만 명의 동시 사용자를 시뮬레이션하여 요청을 지속적으로 생성합니다.
- JMeter, K6, Locust 테스트 스크립트 또는 간단한 HTTP 엔드포인트 구성을 사용하여 애플리케이션 테스트를 사용자 지정합니다.
- 로드 테스트를 즉시, 향후 날짜 및 시간에 또는 반복 일정에 따라 실행하도록 예약합니다.
- 여러 시나리오 및 리전에서 여러 로드 테스트를 동시에 실행합니다.

이 구현 가이드는 AWS의 분산 로드 테스트 솔루션, 참조 아키텍처 및 구성 요소, 배포 계획 고려 사항, Amazon Web Services(AWS) 클라우드에 솔루션을 배포하기 위한 구성 단계에 대한 개요를 제공합니다. 여기에는 보안 및 가용성에 대한 AWS 모범 사례를 사용하여이 솔루션을 배포하는 데 필요한 AWS 서비스를 시작하고 구성하는 AWS CloudFormation 템플릿에 대한 링크가 포함되어 있습니다.

환경에서이 솔루션의 기능을 사용하기 위한 대상에는 AWS 클라우드에서 설계한 실제 경험이 있는 IT 인프라 아키텍트, 관리자 및 DevOps 전문가가 포함됩니다.

이 탐색 테이블을 사용하여 다음 질문에 대한 답을 빠르게 찾을 수 있습니다.

다음을 수행하려는 경우 . . .	읽기 . . .
이 솔루션을 실행하는 데 드는 비용을 파악합니다. 미국 동부(버지니아 북부) 리전에서이 솔루션을 실행하는 데 드는 예상 비용은 AWS 리소스에 대해 매월 30.90 USD입니다.	비용
이 솔루션의 보안 고려 사항을 이해합니다.	보안

다음을 수행하려는 경우 . . .	읽기 . . .
이 솔루션의 할당량을 계획하는 방법을 파악합니다.	할당량
이 솔루션을 지원하는 AWS 리전을 파악합니다.	지원되는 AWS 리전
AI 지원 로드 테스트 분석을 위한 선택적 MCP 서버에 대해 알아봅니다.	MCP 서버 통합
이 솔루션에 포함된 AWS CloudFormation 템플릿을 보거나 다운로드하여 이 솔루션의 인프라 리소스(“스택”)를 자동으로 배포합니다.	AWS CloudFormation 템플릿
소스 코드에 액세스하고 선택적으로 AWS Cloud Development Kit(AWS CDK)를 사용하여 솔루션을 배포합니다.	GitHub 리포지토리

기능

솔루션은 다음 기능을 제공합니다.

다중 테스트 프레임워크 지원

JMeter, K6 및 Locust 테스트 스크립트와 사용자 지정 스크립트 없이 간단한 HTTP 엔드포인트 테스트를 지원합니다. 자세한 내용은 아키텍처 세부 정보 섹션의 테스트 유형을 참조하세요.

높은 사용자 로드 시뮬레이션

수만 명의 동시 가상 사용자를 시뮬레이션하여 실제 로드 조건에서 애플리케이션을 스트레스 테스트합니다.

다중 리전 로드 배포

여러 AWS 리전에 로드 테스트를 분산하여 지리적으로 분산된 사용자 트래픽을 시뮬레이션하고 글로벌 성능을 평가합니다.

유연한 테스트 예약

자동 회귀 테스트를 위해 cron 표현식을 사용하여 테스트를 즉시, 특정 미래 날짜 및 시간에 또는 반복 일정에 따라 실행하도록 예약합니다.

실시간 모니터링

응답 시간, 가상 사용자 수, 요청 성공률을 포함한 실시간 지표를 사용하여 테스트 진행 상황을 모니터링하는 선택적 라이브 데이터 스트리밍을 제공합니다.

포괄적인 테스트 결과

성능 지표, 백분위수(p50, p90, p95, p99), 오류 분석 및 오프라인 분석을 위한 다운로드 가능한 아티팩트와 함께 자세한 테스트 결과를 표시합니다.

기준 비교

시간 경과에 따른 개선 또는 회귀를 추적하기 위해 성능 비교를 위한 기준 테스트 실행을 지정합니다.

엔드포인트 유연성

AWS 리전, 온프레미스 환경 또는 기타 클라우드 공급자에서 HTTP 또는 HTTPS 엔드포인트를 테스트합니다.

직관적 웹 콘솔

명령줄 상호 작용 없이 테스트를 생성, 관리 및 모니터링하기 위한 웹 기반 콘솔을 제공합니다.

AI 지원 분석(선택 사항)

로드 테스트 데이터의 지능형 분석을 위해 모델 컨텍스트 프로토콜(MCP) 서버를 통해 AI 개발 도구와 통합합니다.

다중 프로토콜 지원

사용자 지정 테스트 스크립트를 통해 HTTP, HTTPS, WebSocket, JDBC, JMS, FTP 및 gRPC를 비롯한 다양한 프로토콜을 지원합니다.

이점

솔루션은 다음과 같은 이점을 제공합니다.

포괄적인 성능 테스트

로드 테스트, 스트레스 테스트 및 내구성 테스트를 지원하여 다양한 조건에서 애플리케이션 성능을 철저하게 평가합니다.

조기 문제 감지

프로덕션 배포 전에 성능 병목 현상, 메모리 누수 및 확장성 문제를 식별하여 중단 위험을 줄입니다.

실제 사용 시뮬레이션

실제 사용자 동작과 트래픽 패턴을 정확하게 시뮬레이션하여 실제 조건에서 애플리케이션 성능을 검증합니다.

실행 가능한 성능 개선 도우미

애플리케이션 동작을 이해하고 최적화 작업을 안내하기 위한 자세한 지표, 백분위수 및 오류 분석을 제공합니다.

자동 테스트 워크플로

수동 개입 없이 지속적인 성능 모니터링 및 회귀 테스트를 위해 예약 및 반복 테스트를 활성화합니다.

비용 효율적인 인프라

서버리스 AWS Fargate 컨테이너를 pay-per-use과 함께 사용하므로 전용 테스트 인프라와 지속적인 구독 요금이 필요하지 않습니다.

빠른 테스트 배포

서버를 프로비저닝하거나 관리하지 않고도 몇 분 안에 테스트 인프라를 배포하고 확장합니다.

테스트 결과의 간편한 조사

선택적 모델 컨텍스트 프로토콜(MCP) 서버를 통해 AI 개발 도구와 통합하여 자연어 쿼리를 활성화하고 로드 테스트 데이터를 지능적으로 분석하여 인사이트를 높이고 문제를 해결할 수 있습니다.

사용 사례

사전 프로덕션 검증

새 버전을 시작하기 전에 프로덕션과 유사한 로드 조건에서 웹 및 모바일 애플리케이션을 테스트하여 성능을 검증하고 문제를 식별합니다.

용량 계획

애플리케이션이 현재 인프라에서 지원할 수 있는 최대 동시 사용자 수를 결정하고 규모 조정이 필요한 시기를 식별합니다.

피크 로드 확인

인프라가 성능 저하 없이 피크 로드, 계절 트래픽 급증 또는 예상치 못한 수요 급증을 처리할 수 있는지 확인합니다.

성능 최적화

느린 데이터베이스 쿼리, 비효율적인 코드 실행, 네트워크 지연 시간 또는 리소스 제약 조건과 같은 성능 병목 현상을 식별합니다.

회귀 테스트

반복 로드 테스트를 예약하여 새 코드 배포 또는 인프라 변경으로 인한 성능 회귀를 감지합니다.

글로벌 성과 평가

여러 지리적 리전의 애플리케이션 성능을 평가하여 전 세계 고객에게 일관된 사용자 경험을 제공합니다.

API 로드 테스트

REST APIs, GraphQL 엔드포인트 또는 마이크로서비스를 테스트하여 로드 시 응답 시간, 처리량 및 오류율을 검증합니다.

CI/CD 파이프라인 통합

자동화된 성능 테스트를 지속적 통합 및 배포 파이프라인에 통합하여 개발 주기 초기에 성능 문제를 파악합니다.

타사 서비스 테스트

다양한 로드 조건에서 애플리케이션이 의존하는 타사 APIs 또는 서비스의 성능과 신뢰성을 테스트합니다.

개념 및 정의

이 섹션에서는 이 솔루션과 관련된 핵심 개념 및 용어에 대해 설명합니다.

시나리오

테스트 이름, 설명, 작업 수, 동시성, AWS 리전, 램프업, 대기, 테스트 유형, 일정 날짜 및 반복 구성을 포함한 테스트 정의입니다.

작업 수

테스트 시나리오를 실행하기 위해 Fargate 클러스터에서 시작할 컨테이너 수입니다. Fargate 리소스에 대한 계정 한도에 도달하면 추가 작업이 생성되지 않습니다. 그러나 이미 실행 중인 작업은 계속됩니다.

concurrency

동시성(작업당 동시 가상 사용자 수). 기본 설정을 기반으로 권장되는 동시성은 200입니다. 동시성은 CPU와 메모리에 의해 제한됩니다. Apache JMeter 기반 테스트의 경우 동시성이 높을수록 ECS 작업에서 JVM이 사용하는 메모리가 증가합니다. 기본 ECS 작업 정의는 메모리가 4GB인 작업을 생성합니다. 작업 1개에 대해 더 낮은 동시성 값으로 시작하고 작업 클러스터에 대한 ECS CloudWatch 지표를 모니터링하는 것이 좋습니다. Amazon ECS 클러스터 사용률 지표를 참조하세요.

램프업

0에서 대상 동시성 수준으로 점진적으로 증가하는 기간입니다.

에 대한 보류

램프업이 완료된 후 대상 동시성 수준을 유지하는 기간입니다.

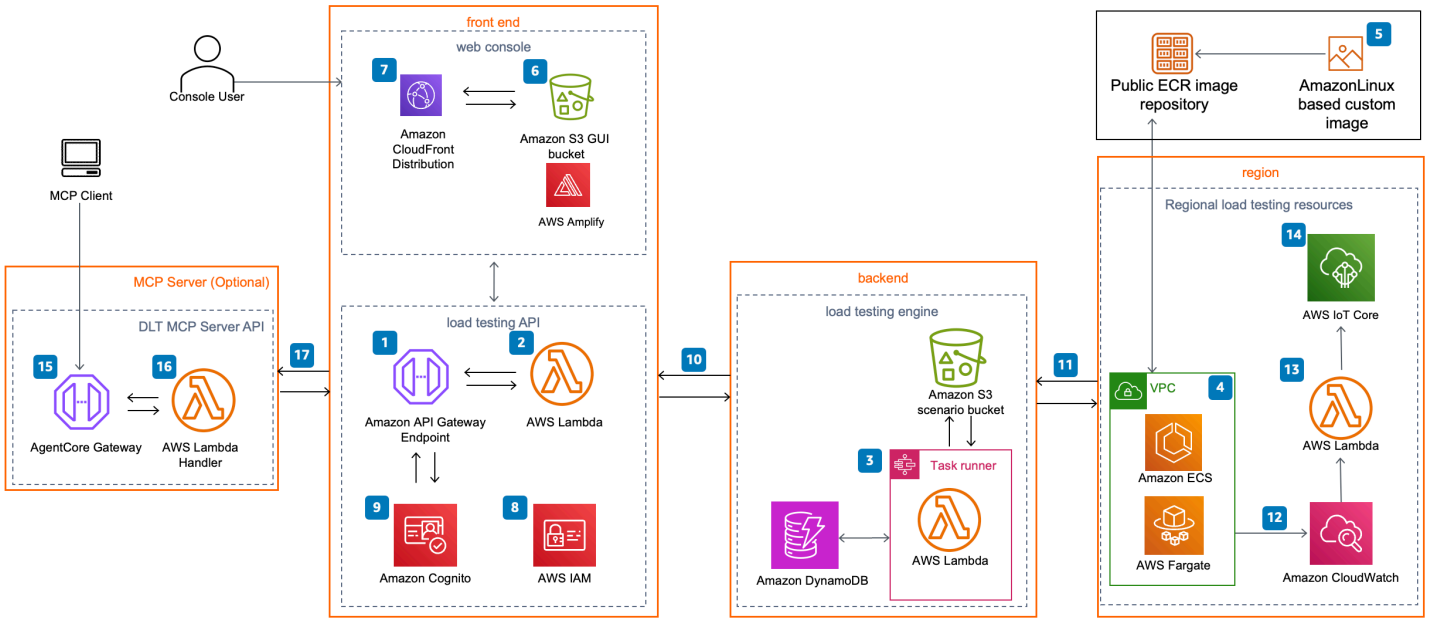
AWS 용어에 대한 일반 참조는 AWS 용어집을 참조하세요.

아키텍처 개요

아키텍처 다이어그램

기본 파라미터를 사용하여이 솔루션을 배포하면 AWS 계정에 다음 구성 요소가 배포됩니다.

AWS 아키텍처의 분산 로드 테스트



> **Note**
>
> AWS CloudFormation 리소스는 AWS Cloud Development Kit(AWS CDK) 구문에서 생성됩니다.

AWS CloudFormation 템플릿과 함께 배포된 솔루션 구성 요소의 상위 수준 프로세스 흐름은 다음과 같습니다.

1. 분산 로드 테스터 API는 Amazon API Gateway를 활용하여 솔루션의 마이크로서비스(AWS Lambda 함수)를 호출합니다.
2. 마이크로서비스는 테스트 데이터를 관리하고 테스트를 실행하는 비즈니스 로직을 제공합니다.

3. 이러한 마이크로서비스는 Amazon Simple Storage Service(Amazon S3), Amazon DynamoDB 및 AWS Step Functions와 상호 작용하여 테스트 시나리오 세부 정보 및 결과를 저장하고 테스트 실행을 오케스트레이션합니다.
4. Amazon Virtual Private Cloud(Amazon VPC) 네트워크 토폴로지는 AWS Fargate에서 실행되는 솔루션의 Amazon Elastic Container Service(Amazon ECS) 컨테이너가 포함된를 배포합니다.
5. 컨테이너는 Taurus 로드 테스트 프레임워크가 설치된 Amazon Linux 2023 기본 이미지를 사용합니다. Taurus는 JMeter, K6, Locust 및 기타 테스트 도구를 지원하는 오픈 소스 테스트 자동화 프레임워크입니다. 컨테이너 이미지는 Open Container Initiative(OCI)를 준수하며 Amazon Elastic Container Registry(Amazon ECR) 퍼블릭 리포지토리에서 AWS가 호스팅합니다. 자세한 내용은 컨테이너 이미지 사용자 지정을 참조하세요.
6. AWS Amplify로 구동되는 웹 콘솔은 정적 웹 호스팅을 위해 구성된 S3 버킷에 배포됩니다.
7. Amazon CloudFront는 솔루션의 웹 사이트 버킷 콘텐츠에 대한 안전한 퍼블릭 액세스를 제공합니다.
8. 초기 구성 중에 솔루션은 기본 관리자 역할(IAM 역할)을 생성하고 고객이 지정한 사용자 이메일 주소로 액세스 초대를 보냅니다.
9. Amazon Cognito 사용자 풀은 콘솔, 분산 로드 테스터 API 및 MCP 서버에 대한 사용자 액세스를 관리합니다.
10. 이 솔루션을 배포한 후 웹 콘솔 또는 APIs를 사용하여 일련의 작업을 정의하는 테스트 시나리오를 생성하고 실행할 수 있습니다.
11. 마이크로서비스는이 테스트 시나리오를 사용하여 지정된 리전의 Fargate에서 ECS 작업을 실행합니다.
12. 테스트가 완료되면 솔루션은 결과를 S3 및 DynamoDB에 저장하고 출력을 Amazon CloudWatch에 기록합니다.
13. 라이브 데이터 옵션을 활성화하면 솔루션은 테스트가 실행되는 각 리전의 테스트 중에 Fargate 작업의 CloudWatch 로그를 Lambda 함수로 전송합니다.
14. Lambda 함수는 기본 스택이 배포된 리전의 AWS IoT Core에서 해당 주제에 데이터를 게시합니다. 웹 콘솔은 주제를 구독하고 테스트가 실행되는 동안 실시간 데이터를 표시합니다.

> **Note**
>
> 다음 단계에서는 AI 지원 로드 테스트 분석을 위한 선택적 MCP Server 통합에 대해 설명합니다. 이 구성 요소는 솔루션 배포 중에 MCP 서버 옵션을 선택한 경우에만 배포됩니다.

15MCP 클라이언트(AI 개발 도구)는 AWS AgentCore Gateway 엔드포인트에 연결하여 모델 컨텍스트 프로토콜을 통해 분산 로드 테스트 솔루션의 데이터에 액세스합니다. AgentCore Gateway는 사용자의 Cognito 인증 토큰을 검증하여 MCP 서버에 대한 권한 있는 액세스를 보장합니다.

16.인증에 성공하면 AgentCore Gateway는 MCP 도구 요청을 DLT MCP 서버 Lambda 함수로 전달합니다. Lambda 함수는 구조화된 데이터를 AgentCore Gateway로 반환하여 AI 지원 분석 및 인사이트를 위해 MCP 클라이언트로 다시 보냅니다.

17Lambda 함수는 요청을 처리하고 적절한 AWS 리소스(DynamoDB 테이블, S3 버킷 또는 CloudWatch 로그)를 쿼리하여 요청된 로드 테스트 데이터를 검색합니다.

AWS Well-Architected 설계 고려 사항

이 솔루션은 고객이 클라우드에서 안정적이고 안전하며 효율적이고 비용 효율적인 워크로드를 설계하고 운영할 수 있도록 지원하는 AWS Well-Architected Framework의 모범 사례를 사용합니다.

이 섹션에서는 Well-Architected Framework의 설계 원칙과 모범 사례가 이 솔루션에 어떤 이점을 제공하는지 설명합니다.

운영 우수성

이 섹션에서는 운영 우수성 요소의 원칙과 모범 사례를 사용하여 이 솔루션을 설계한 방법을 설명합니다.

- 모든 리소스는 AWS CDK 구문에서 생성된 AWS CloudFormation 템플릿을 사용하여 코드형 인프라로 정의됩니다.
- 솔루션은 다양한 단계에서 지표를 CloudWatch에 푸시하여 Lambda 함수, ECS 작업, S3 버킷 및 기타 솔루션 구성 요소에 대한 관찰성을 제공합니다.

보안

이 섹션에서는 보안 요소의 원칙과 모범 사례를 사용하여 이 솔루션을 설계한 방법을 설명합니다.

- Cognito는 웹 콘솔 사용자 및 API 요청을 인증하고 승인합니다.
- 모든 서비스 간 통신은 필요한 최소 권한만 포함하는 최소 권한 액세스 권한이 있는 AWS Identity and Access Management(IAM) 역할을 사용합니다.
- S3 버킷 및 DynamoDB 테이블을 포함한 모든 데이터 스토리지는 AWS 관리형 키를 사용하여 저장 데이터를 암호화합니다.

- 감사 및 규정 준수 목적으로 해당하는 경우 로깅, 추적 및 버전 관리가 활성화됩니다.
- 네트워크 액세스는 기본적으로 프라이빗이며, AWS 네트워크 내에서 트래픽을 유지할 수 있는 경우 VPC 엔드포인트가 활성화됩니다.

> **Note**
>
> 이 솔루션은 로그 볼륨 및 비용 고려 사항에 따라 보존 기간이 다양한 여러 CloudWatch 로그 그룹을 생성합니다.
>
> | 로그 유형 | 보존 기간 |
> | --- | --- |
> | ECS 컨테이너 인사이트 | 1일 |
> | Step Functions, ECS 사용자 지정 로그, API Gateway 액세스 로그 | 1년 |
> | Lambda 런타임 로그 | 2년 |
> | API Gateway 실행 로그 | 만료되지 않음 |
>
> 요구 사항에 따라 CloudWatch 콘솔에서 이러한 보존 기간을 수정할 수 있습니다.

신뢰성

이 섹션에서는 신뢰성 요소의 원칙과 모범 사례를 사용하여 이 솔루션을 설계한 방법을 설명합니다.

- 이 솔루션은 가능한 경우 AWS 서버리스 서비스(예: Lambda, API Gateway, Amazon S3, AWS Step Functions, Amazon DynamoDB 및 AWS Fargate)를 사용하여 서비스 장애로부터 고가용성과 복구를 보장합니다.
- 모든 컴퓨팅 처리는 Lambda 함수 또는 AWS Fargate의 Amazon ECS를 사용합니다.
- 데이터는 DynamoDB 및 Amazon S3에 저장되므로 기본적으로 여러 가용 영역에 유지됩니다.

성능 효율성

이 섹션에서는 성능 효율성 요소의 원칙과 모범 사례를 사용하여 이 솔루션을 설계한 방법을 설명합니다.

- 이 솔루션은 필요에 따라 수평적으로 확장할 수 있는 서버리스 아키텍처를 사용합니다.
- 솔루션은 AWS Lambda, Amazon API Gateway, Amazon S3, AWS Step Functions, Amazon DynamoDB, Amazon ECS, AWS Fargate, Amazon Cognito 등이 솔루션의 AWS 서비스를 지원하는 모든 리전에서 시작할 수 있습니다.
- 이 솔루션은 전체적으로 관리형 서비스를 사용하여 리소스 프로비저닝 및 관리의 운영 부담을 줄입니다.
- 솔루션은 AWS 서비스가 변경될 때 일관성을 달성하기 위해 매일 자동으로 테스트 및 배포되며, 솔루션 아키텍트와 주제 전문가가 실험 및 개선 영역에 대해 검토합니다.

비용 최적화

이 섹션에서는 비용 최적화 요소의 원칙과 모범 사례를 사용하여 이 솔루션을 설계한 방법을 설명합니다.

- 이 솔루션은 서버리스 아키텍처를 사용하므로 고객은 사용한 만큼만 요금을 청구받습니다.
- Amazon DynamoDB는 온디맨드로 용량을 확장하므로 사용하는 용량에 대해서만 비용을 지불하면 됩니다.
- AWS Fargate의 AWS ECS를 사용하면 사용한 컴퓨팅 리소스에 대해서만 선결제 비용 없이 비용을 지불할 수 있습니다.
- AWS AgentCore Gateway는 분산 로드 테스트 API에 대한 비용 효율적인 Lambda 기반 프록시 역할을 하므로 전용 인프라가 필요하지 않으며 서버리스 pay-per-request 요금을 통해 비용을 절감할 수 있습니다.

지속 가능성

이 섹션에서는 지속 가능성 요소의 원칙과 모범 사례를 사용하여 이 솔루션을 설계한 방법을 설명합니다.

- 이 솔루션은 관리형 서버리스 서비스를 사용하여 온프레미스 서비스를 지속적으로 운영하는 것에 비해 백엔드 서비스의 환경 영향을 최소화합니다.
- 서버리스 서비스를 사용하면 필요에 따라 스케일 업 또는 스케일 다운할 수 있습니다.

아키텍처 세부 정보

이 섹션에서는 이 솔루션을 구성하는 구성 요소 및 AWS 서비스와 이러한 구성 요소가 함께 작동하는 방식에 대한 아키텍처 세부 정보를 설명합니다.

AWS의 분산 로드 테스트 솔루션은 프런트 엔드, 백엔드 및 선택적 MCP 서버라는 세 가지 상위 수준 구성 요소로 구성됩니다.

프런트 엔드

프런트 엔드는 솔루션과 상호 작용하기 위한 인터페이스를 제공하며 다음을 포함합니다.

- 프로그래밍 방식 액세스를 위한 로드 테스트 API
- 성능 테스트 생성, 예약 및 실행을 위한 웹 콘솔
- 테스트 결과 및 오류의 AI 지원 분석을 위한 선택적 MCP 서버

로드 테스트 API

AWS의 분산 로드 테스트는 솔루션의 RESTful API를 호스팅하도록 Amazon API Gateway를 구성합니다. 사용자는 포함된 웹 콘솔, RESTful API 및 선택적 MCP 서버를 통해 로드 테스트 시스템과 안전하게 상호 작용할 수 있습니다. API는 Amazon DynamoDB에 저장된 테스트 데이터에 액세스하기 위한 "정문" 역할을 합니다. APIs를 사용하여 솔루션에 빌드하는 모든 확장 기능에 액세스할 수도 있습니다.

이 솔루션은 Amazon Cognito 사용자 풀의 사용자 인증 기능을 활용합니다. 사용자를 성공적으로 인증한 후 Amazon Cognito는 콘솔이 솔루션의 APIs(Amazon API Gateway 엔드포인트)에 요청을 제출하도록 허용하는 데 사용되는 JSON 웹 토큰을 발급합니다. HTTPS 요청은 토큰이 포함된 권한 부여 헤더와 함께 콘솔에서 APIs로 전송됩니다.

요청에 따라 API Gateway는 적절한 AWS Lambda 함수를 호출하여 DynamoDB 테이블에 저장된 데이터에 필요한 작업을 수행하고, 테스트 시나리오를 Amazon S3에 JSON 객체로 저장하고, Amazon CloudWatch 지표 이미지를 검색하고, 테스트 시나리오를 AWS Step Functions 상태 시스템에 제출합니다.

솔루션의 API에 대한 자세한 내용은이 가이드의 분산 로드 테스트 API 섹션을 참조하세요.

웹 콘솔

이 솔루션에는 테스트를 구성 및 실행하고, 실행 중인 테스트를 모니터링하고, 자세한 테스트 결과를 보는 데 사용할 수 있는 웹 콘솔이 포함되어 있습니다. 콘솔은 직관적인 웹 애플리케이션을 구축하기 위한 오픈 소스 설계 시스템인 Cloudscape로 구축된 ReactJS 애플리케이션입니다. 콘솔은 Amazon S3에서 호스팅되며 Amazon CloudFront를 통해 액세스됩니다. 애플리케이션은 AWS Amplify를 활용하여 Amazon Cognito와 통합하여 사용자를 인증합니다. 웹 콘솔에는 AWS IoT Core에서 해당 주제를 구독하는 실행 중인 테스트의 라이브 데이터를 보는 옵션도 포함되어 있습니다.

웹 콘솔 URL은 CloudFormation 출력에서 콘솔로 찾을 수 있는 CloudFront 배포 도메인 이름입니다. CloudFormation CloudFormation 템플릿을 시작하면 웹 콘솔 URL과 로그인하기 위한 일회용 암호가 포함된 이메일도 받게 됩니다.

MCP 서버(선택 사항)

선택적 모델 컨텍스트 프로토콜(MCP) 서버는 AI 개발 도구가 자연어 상호 작용을 통해 로드 테스트 데이터에 액세스하고 분석할 수 있는 추가 인터페이스를 제공합니다. 이 구성 요소는 솔루션 배포 중에 MCP 서버 옵션을 선택한 경우에만 배포됩니다.

MCP 서버를 사용하면 AI 에이전트가 Amazon Q, Claude 및 기타 MCP 호환 AI 어시스턴트와 같은 도구를 사용하여 테스트 결과를 쿼리하고, 성능 지표를 분석하고, 로드 테스트 데이터에 대한 인사이트를 얻을 수 있습니다. MCP Server 아키텍처 및 구성에 대한 자세한 내용은이 섹션의 MCP Server를 참조하세요.

Backend

백엔드는 테스트에 대한 로드를 생성하는 데 사용하는 컨테이너 이미지 파이프라인과 로드 테스트 엔진으로 구성됩니다. 프런트 엔드를 통해 백엔드와 상호 작용합니다. 또한 각 테스트에 대해 시작된 AWS Fargate의 Amazon ECS 태스크에는 고유한 테스트 식별자(ID)로 태그가 지정됩니다. 이러한 테스트 ID 태그를 사용하여이 솔루션의 비용을 모니터링할 수 있습니다. 자세한 내용은 AWS Billing and Cost Management 사용 설명서의 사용자 정의 비용 할당 태그를 참조하세요.

컨테이너 이미지 파이프라인

이 솔루션은 Amazon Linux 2023으로 빌드된 컨테이너 이미지를 Taurus 로드 테스트 프레임워크가 설치된 기본 이미지로 사용합니다. Taurus는 JMeter, K6, Locust 및 기타 테스트 도구를 지원하는 오픈 소스 테스트 자동화 프레임워크입니다. AWS는 Amazon Elastic Container Registry(Amazon ECR) 퍼블릭 리포지토리에서이 이미지를 호스팅합니다. 이 솔루션은이 이미지를 사용하여 AWS Fargate 클러스터의 Amazon ECS에서 작업을 실행합니다.

자세한 내용은이 가이드의 컨테이너 이미지 사용자 지정 섹션을 참조하세요.

인프라 테스트

이 솔루션은 기본 CloudFormation 템플릿 외에도 여러 리전에서 테스트를 실행하는 데 필요한 리소스를 시작하는 리전 템플릿을 제공합니다. 솔루션은이 템플릿을 Amazon S3에 저장하고 웹 콘솔에 해당 템플릿에 대한 링크를 제공합니다. 각 리전 스택에는 라이브 데이터를 처리하기 위한 VPC, AWS Fargate 클러스터 및 Lambda 함수가 포함되어 있습니다.

추가 리전에 테스트 인프라를 배포하는 방법에 대한 자세한 내용은이 가이드의 다중 리전 배포 섹션을 참조하세요.

로드 테스트 엔진

Distributed Load Testing 솔루션은 Amazon Elastic Container Service(Amazon ECS) 및 AWS Fargate를 사용하여 여러 리전에서 수천 명의 동시 사용자를 시뮬레이션하여 지속적인 속도로 HTTP 요청을 생성합니다.

포함된 웹 콘솔을 사용하여 테스트 파라미터를 정의합니다. 이 솔루션은 이러한 파라미터를 사용하여 JSON 테스트 시나리오를 생성하고 Amazon S3에 저장합니다. 테스트 스크립트 및 테스트 파라미터에 대한 자세한 내용은이 섹션의 테스트 유형을 참조하세요.

AWS Step Functions 상태 시스템은 AWS Fargate 클러스터에서 Amazon ECS 작업을 실행하고 모니터링합니다. AWS Step Functions 상태 시스템에는 ecr-checker AWS Lambda 함수, task-status-checker AWS Lambda 함수, task-runner AWS Lambda 함수, task-canceler AWS Lambda 함수 및 결과 구문 분석기 AWS Lambda 함수가 포함되어 있습니다. 워크플로에 대한 자세한 내용은이 가이드의 워크플로 테스트 섹션을 참조하세요. 테스트 결과에 대한 자세한 내용은이 가이드의 테스트 결과 섹션을 참조하세요. 테스트 취소 워크플로에 대한 자세한 내용은이 가이드의 테스트 취소 워크플로 섹션을 참조하세요.

라이브 데이터를 선택하면 솔루션은 해당 리전의 Fargate 작업에 해당하는 CloudWatch 로그를 통해 각 리전에서 real-time-data-publisher Lambda 함수를 시작합니다. 그런 다음 솔루션은 기본 스택을 시작한 리전 내의 AWS IoT Core에서 데이터를 처리하고 주제에 게시합니다. 자세한 내용은이 가이드의 라이브 데이터 섹션을 참조하세요.

MCP 서버

선택적 모델 컨텍스트 프로토콜(MCP) 서버 통합을 통해 AI 에이전트는 자연어 상호 작용을 통해 로드 테스트 데이터에 프로그래밍 방식으로 액세스하고 분석할 수 있습니다. 이 구성 요소는 솔루션 배포 중에 MCP 서버 옵션을 선택한 경우에만 배포됩니다.

MCP 서버는 AI 개발 도구와 DLT 배포 간의 연결 역할을 하여 성능 테스트 결과의 지능형 분석을 위한 표준화된 인터페이스를 제공합니다. 아키텍처는 여러 AWS 서비스를 통합하여 AI 에이전트 상호 작용을 위한 안전하고 확장 가능한 인터페이스를 생성합니다.

AWS AgentCore 게이트웨이

AWS AgentCore Gateway는 MCP 서버에 대해 표준화된 호스팅 및 프로토콜 관리를 제공하는 완전관리형 서비스입니다. 이 솔루션에서 AgentCore Gateway는 로드 테스트 데이터에 대한 액세스를 요청할 때 AI 에이전트가 연결하는 퍼블릭 엔드포인트 역할을 합니다.

서비스는 도구 검색, 인증 토큰 검증 및 요청 라우팅을 포함한 모든 MCP 프로토콜 통신을 처리합니다. AgentCore Gateway는 각 요청에 대한 Cognito 토큰 서명 및 클레임을 검증하는 동시에 퍼블릭 엔드포인트에 대한 일반적인 위협에 대한 보안 보호 기능이 내장된 다중 테넌트 서비스로 작동합니다.

DLT MCP 서버 Lambda

DLT MCP 서버 Lambda 함수는 AI 에이전트의 MCP 요청을 처리하고 이를 DLT 리소스에 대한 쿼리로 변환하는 사용자 지정 서버리스 구성 요소입니다.

이 Lambda 함수는 MCP 통합의 인텔리전스 계층 역할을 하여 DynamoDB 테이블에서 테스트 결과를 검색하고, S3 버킷에 저장된 성능 아티팩트에 액세스하고, 자세한 실행 정보를 위해 CloudWatch 로그를 쿼리합니다. Lambda 함수는 읽기 전용 액세스 패턴을 구현하고 원시 DLT 데이터를 에이전트가 쉽게 해석하고 분석할 수 있는 구조화되고 AI 친화적인 형식으로 변환합니다.

인증 통합

인증 시스템은 기존 Cognito 사용자 풀 인프라를 활용하여 웹 콘솔과 MCP 서버 인터페이스 모두에서 일관된 액세스 제어를 유지합니다.

이 통합은 OAuth 2.0 토큰 기반 인증을 사용합니다. 사용자는 Cognito 로그인 프로세스를 통해 한 번 인증하고 UI 상호 작용과 MCP 서버 액세스 모두에 적합한 토큰을 수신합니다. 시스템은 웹 인터페이스와 동일한 권한 경계 및 액세스 제어를 유지하여 사용자가 콘솔을 통해 액세스할 수 있는 것과 동일한 로드 테스트 데이터에 AI 에이전트를 통해서만 액세스할 수 있도록 합니다.

이 솔루션의 AWS 서비스

이 솔루션에는 다음 AWS 서비스가 포함되어 있습니다.

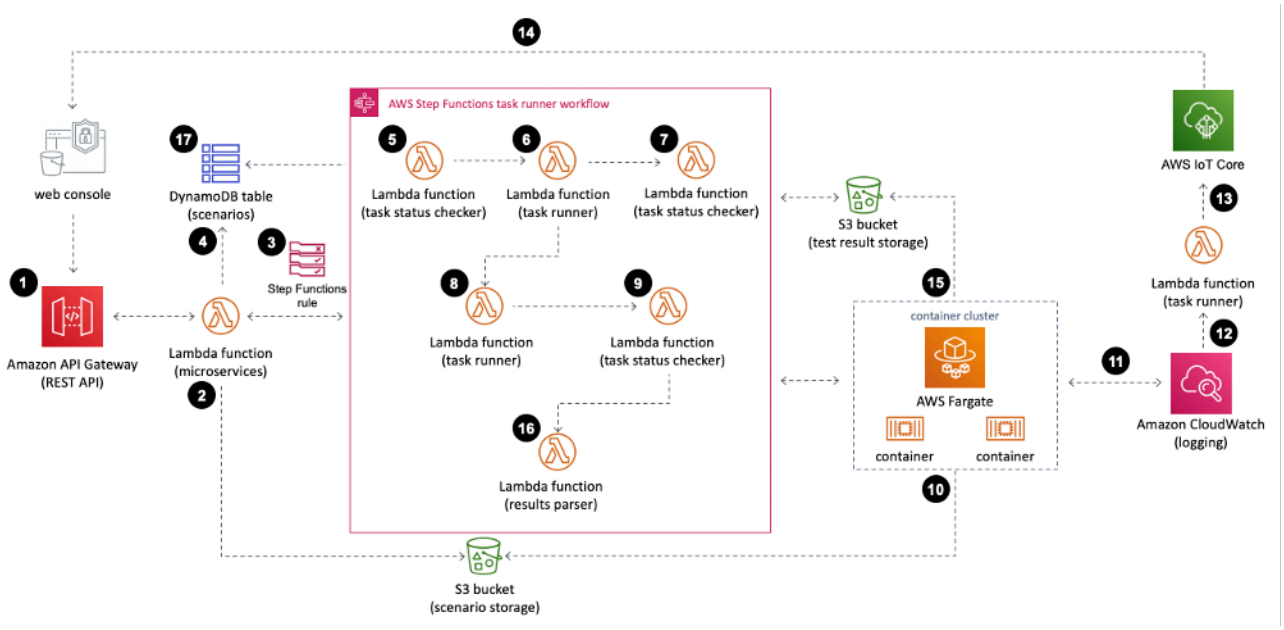
AWS 서비스	설명
Amazon API Gateway	Core. 솔루션에서 REST API 엔드포인트를 호스팅합니다.
CloudFormation	Core. 솔루션 인프라에 대한 배포를 관리합니다.
Amazon CloudFront	Core. Amazon S3에서 호스팅되는 웹 콘텐츠를 제공합니다.
Amazon CloudWatch()	Core. 솔루션 로그 및 지표를 저장합니다.
Amazon Cognito	Core. API에 대한 사용자 관리 및 인증을 처리합니다.
Amazon DynamoDB	Core. 배포 정보를 저장하고 시나리오 세부 정보 및 결과를 테스트합니다.
Amazon Elastic Container Service	Core. AWS Fargate 컨테이너에서 독립적인 Amazon ECS 작업을 배포하고 관리합니다.
AWS Fargate	Core. 호스트 솔루션의 Amazon ECS 컨테이너
AWS Identity and Access Management	Core. 사용자 역할 및 권한 관리를 처리합니다.
Lambda	Core. APIs 구현, 테스트 결과 구문 분석 및 작업자/리더 작업 시작을 위한 로직을 제공합니다.
AWS Step Functions	Core. 지정된 리전의 AWS Fargate 태스크에서 Amazon ECS 컨테이너 프로비저닝을 오케스트레이션합니다.
Amplify	지원. AWS Amplify로 구동되는 웹 콘솔을 제공합니다.
Amazon CloudWatch Events	지원. 지정된 날짜 또는 반복 날짜에 자동으로 시작되도록 테스트를 예약합니다.
Amazon Elastic 컨테이너 레지스트리	지원. 컨테이너 이미지를 퍼블릭 ECR 리포지토리에 호스팅합니다.
AWS IoT Core	지원. AWS IoT Core에서 해당 주제를 구독하여 실행 중인 테스트에 대한 라이브 데이터를 볼 수 있습니다.

AWS 서비스	설명
AWS Systems Manager	지원. 리소스 작업 및 비용 데이터에 대한 애플리케이션 수준의 리소스 모니터링 및 시각화를 제공합니다.
Amazon S3	지원. 정적 웹 콘텐츠, 로그, 지표 및 테스트 데이터를 호스팅합니다.
Amazon Virtual Private Cloud	지원. AWS Fargate에서 실행되는 솔루션의 Amazon ECS 컨테이너를 포함합니다.
Amazon Bedrock AgentCore	지원, 선택 사항. AI 에이전트와 API 통합을 위한 솔루션의 선택적 원격 모델 컨텍스트 프로토콜(MCP) 서버를 호스팅합니다.

AWS의 분산 로드 테스트 작동 방식

다음 세부 분석에는 테스트 시나리오 실행과 관련된 단계가 나와 있습니다.

테스트 워크플로



1. 웹 콘솔을 사용하여 구성 세부 정보가 포함된 테스트 시나리오를 솔루션의 API에 제출합니다.
2. 테스트 시나리오 구성은 Amazon Simple Storage Service(Amazon S3)에 JSON 파일()로 업로드됩니다`s3://<bucket-name>/test-scenarios/<$TEST_ID>/<$TEST_ID>.json`.

3. AWS Step Functions 상태 시스템은 테스트 ID, 작업 수, 테스트 유형 및 파일 유형을 AWS Step Functions 상태 시스템 입력으로 사용하여 실행됩니다. 테스트가 예약된 경우 먼저 지정된 날짜에 AWS Step Functions를 트리거하는 CloudWatch Events 규칙을 생성합니다. 일정 워크플로에 대한 자세한 내용은이 가이드의 일정 워크플로 테스트 섹션을 참조하세요.
4. 구성 세부 정보는 Amazon DynamoDB 테이블 시나리오에 저장됩니다.
5. AWS Step Functions 작업 실행기 워크플로에서 task-status-checker AWS Lambda 함수는 Amazon Elastic Container Service(Amazon ECS) 작업이 동일한 테스트 ID에 대해 이미 실행 중인지 확인합니다. 동일한 테스트 ID의 작업이 실행 중인 것으로 확인되면 오류가 발생합니다. AWS Fargate 클러스터에서 실행 중인 Amazon ECS 작업이 없는 경우 함수는 테스트 ID, 작업 수 및 테스트 유형을 반환합니다.
6. 작업 실행기 AWS Lambda 함수는 이전 단계에서 작업 세부 정보를 가져오고 AWS Fargate 클러스터에서 Amazon ECS 작업자 작업을 실행합니다. Amazon ECS API는 RunTask 작업을 사용하여 작업자 작업을 실행합니다. 이러한 작업자 작업이 시작된 다음 테스트를 시작하기 위해 리더 작업의 시작 메시지를 기다립니다. RunTask 작업은 정의당 10개의 작업으로 제한됩니다. 작업 수가 10개를 초과하는 경우 모든 작업자 작업이 시작될 때까지 작업 정의가 여러 번 실행됩니다. 또한 함수는 결과 구문 분석 AWS Lambda 함수에서 현재 테스트를 구분하는 접두사를 생성합니다.
7. task-status-checker AWS Lambda 함수는 모든 Amazon ECS 작업자 작업이 동일한 테스트 ID로 실행 중인지 확인합니다. 태스크가 여전히 프로비저닝 중인 경우 1분 동안 기다렸다가 다시 확인합니다. 모든 Amazon ECS 작업이 실행되면 테스트 ID, 작업 수, 테스트 유형, 모든 작업 IDs 및 접두사를 반환하고 이를 작업 실행기 함수에 전달합니다.
8. Task-runner AWS Lambda 함수가 다시 실행되며, 이번에는 리더 노드 역할을 하는 단일 Amazon ECS 작업을 시작합니다. 이 ECS 태스크는 테스트를 동시에 시작하기 위해 각 작업자 태스크에 테스트 시작 메시지를 보냅니다.
9. task-status-checker AWS Lambda 함수는 Amazon ECS 작업이 동일한 테스트 ID로 실행 중인지 다시 확인합니다. 태스크가 계속 실행 중인 경우 1분 동안 기다렸다가 다시 확인합니다. 실행 중인 Amazon ECS 작업이 없으면 테스트 ID, 작업 수, 테스트 유형 및 접두사를 반환합니다.
10. 작업 실행기 AWS Lambda 함수가 AWS Fargate 클러스터에서 Amazon ECS 작업을 실행하면 각 작업은 Amazon S3에서 테스트 구성을 다운로드하고 테스트를 시작합니다.
11. 테스트가 실행되면 각 작업에 대한 평균 응답 시간, 동시 사용자 수, 성공한 요청 수 및 실패한 요청 수가 Amazon CloudWatch에 로깅되고 CloudWatch 대시보드에서 볼 수 있습니다.
12. 테스트에 라이브 데이터를 포함시킨 경우 솔루션은 구독 필터를 사용하여 CloudWatch에서 실시간 테스트 결과를 필터링합니다. 그런 다음 솔루션은 Lambda 함수에 데이터를 전달합니다.
13. 그런 다음 Lambda 함수는 수신된 데이터를 구조화하여 AWS IoT Core 주제에 게시합니다.

14.웹 콘솔은 테스트를 위해 AWS IoT Core 주제를 구독하고 주제에 게시된 데이터를 수신하여 테스트가 실행되는 동안 실시간 데이터를 그래프로 표시합니다.

15.테스트가 완료되면 컨테이너 이미지는 세부 보고서를 XML 파일로 Amazon S3로 내보냅니다. 각 파일에는 파일 이름에 대한 UUID가 부여됩니다. 예: s3://dlte-bucket/test-scenarios/<$TEST_ID>/results/<$UUID>.json.

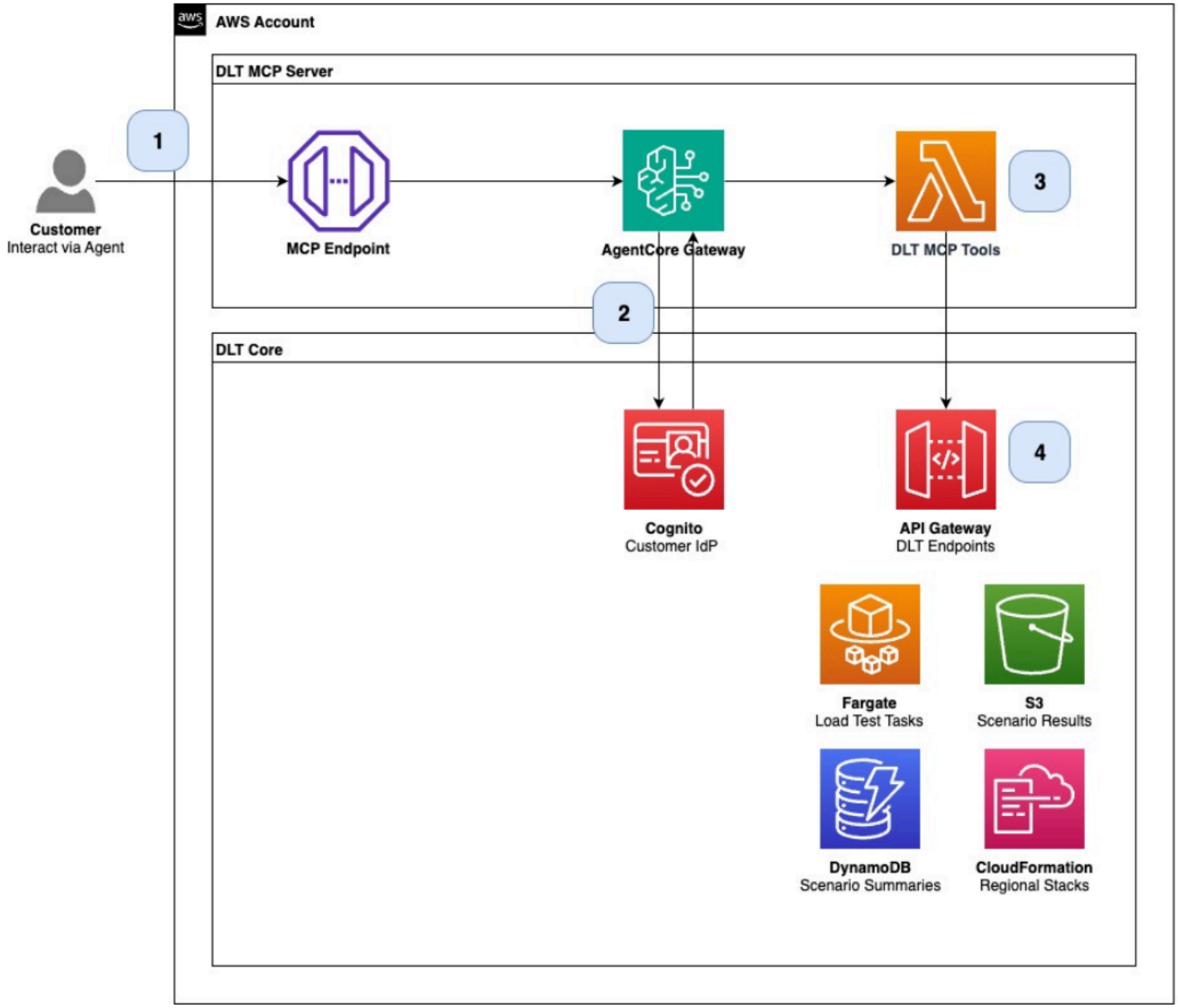
16.XML 파일이 Amazon S3에 업로드되면 결과 구문 분석기 AWS Lambda 함수는 접두사로 시작하는 XML 파일의 결과를 읽고 모든 결과를 구문 분석하여 하나의 요약된 결과로 집계합니다.

17.결과 구문 분석기 AWS Lambda 함수는 집계 결과를 Amazon DynamoDB 테이블에 기록합니다.

MCP Server 워크플로(선택 사항)

선택적 MCP 서버 통합을 배포하는 경우 AI 에이전트는 다음 워크플로를 통해 로드 테스트 데이터에 액세스하고 분석할 수 있습니다.

MCP 서버 아키텍처



1. 고객 상호 작용 - 고객은 AWS AgentCore Gateway에서 호스팅하는 MCP 엔드포인트를 통해 DLT의 MCP와 상호 작용합니다. AI 에이전트는이 엔드포인트에 연결하여 로드 테스트 데이터에 대한 액세스를 요청합니다.
2. 권한 부여 - AgentCore Gateway는 Solution Cognito 사용자 풀 애플리케이션 클라이언트에 대한 권한 부여를 처리합니다. 게이트웨이는 사용자의 Cognito 토큰을 검증하여 DLT MCP 서버에 액세스할 수 있는 권한이 있는지 확인합니다. 권한 있는 사용자에게는 에이전트 도구 액세스 권한이 읽기 전용 작업으로 제한되어 있는 액세스 권한이 부여됩니다.
3. 도구 사양 - AgentCore Gateway는 DLT MCP 서버 Lambda 함수에 연결합니다. 도구 사양은 AI 에이전트가 로드 테스트 데이터와 상호 작용하는 데 사용할 수 있는 도구를 정의합니다.

4. 읽기 전용 API 액세스 - Lambda 함수는 기존 DLT API Gateway 엔드포인트를 통한 읽기 전용 API 액세스로 범위가 지정됩니다. 함수는 네 가지 기본 작업을 제공합니다.
   - 시나리오 나열 - DynamoDB 시나리오 테이블에서 테스트 시나리오 목록을 검색합니다.
   - 시나리오 테스트 결과 가져오기 - DynamoDB 및 S3에서 특정 시나리오에 대한 세부 테스트 결과에 액세스
   - Fargate 로드 테스트 실행기 가져오기 - ECS 클러스터에서 Fargate 작업을 실행하는 방법에 대한 쿼리 정보
   - 사용 가능한 리전 스택 가져오기 - CloudFormation에서 배포된 리전 인프라에 대한 정보 검색

MCP Server 통합은 기존 DLT 인프라(API Gateway, Cognito, DynamoDB, S3)를 활용하여 AI 기반 분석 및 인사이트를 위한 테스트 데이터에 대한 안전하고 읽기 전용 액세스를 제공합니다.

설계 고려 사항

이 섹션에서는 지원되는 애플리케이션, 테스트 유형, 예약 옵션 및 배포 고려 사항을 포함하여 AWS의 분산 로드 테스트 솔루션에 대한 중요한 설계 결정 및 구성 옵션에 대해 설명합니다.

지원되는 애플리케이션

이 솔루션은 AWS 계정에서 애플리케이션으로 네트워크 연결이 있는 한 클라우드 기반 애플리케이션 및 온프레미스 애플리케이션 테스트를 지원합니다. 솔루션은 HTTP 또는 HTTPS 프로토콜을 사용하는 APIs 지원합니다.

테스트 유형

AWS의 분산 로드 테스트는 간단한 HTTP 엔드포인트 테스트, JMeter, K6 및 Locust와 같은 여러 테스트 유형을 지원합니다.

간단한 HTTP 엔드포인트 테스트

웹 콘솔은 사용자 지정 스크립트를 작성하지 않고도 HTTP 또는 HTTPS 엔드포인트를 테스트할 수 있는 HTTP 엔드포인트 구성 인터페이스를 제공합니다. 엔드포인트 URL을 정의하고 드롭다운 메뉴에서 HTTP 메서드(GET, POST, PUT, DELETE 등)를 선택한 다음 선택적으로 사용자 지정 요청 헤더와 본문 페이로드를 추가합니다. 이 구성을 사용하면 사용자 지정 권한 부여 토큰, 콘텐츠 유형 또는 애플리케이션에 필요한 기타 HTTP 헤더 및 요청 본문을 사용하여 APIs를 테스트할 수 있습니다.

JMeter 테스트

웹 콘솔을 사용하여 테스트 시나리오를 생성할 때 JMeter 테스트 스크립트를 업로드할 수 있습니다. 솔루션은 스크립트를 시나리오 S3 버킷에 업로드합니다. Amazon ECS 작업이 실행되면 S3에서 JMeter 스크립트를 다운로드하고 테스트를 실행합니다.

> **Important**
>
> JMeter 스크립트는 동시성(가상 사용자), 트랜잭션 속도(TPS), 증가 시간 및 기타 로드 파라미터를 정의할 수 있지만 솔루션은 테스트 생성 중에 트래픽 셰이프 화면에서 지정한 값으로 이러한 구성을 재정의합니다. 트래픽 셰이프 구성은 테스트 실행의 작업 수, 동시성(작업당 가상 사용자 수), 증가 기간 및 대기 기간을 제어합니다.

JMeter 입력 파일이 있는 경우 JMeter 스크립트와 함께 입력 파일을 압축할 수 있습니다. 테스트 시나리오를 생성할 때 zip 파일을 선택할 수 있습니다.

플러그인을 포함하려는 경우 번들 zip 파일의 /plugins 하위 디렉터리에 포함된 모든 .jar 파일이 JMeter 확장 디렉터리에 복사되고 로드 테스트에 사용할 수 있습니다.

> **Note**
>
> JMeter 입력 파일을 JMeter 스크립트 파일에 포함하는 경우 JMeter 스크립트 파일에 입력 파일의 상대 경로를 포함해야 합니다. 또한 입력 파일은 상대 경로에 있어야 합니다. 예를 들어 JMeter 입력 파일과 스크립트 파일이 /home/user 디렉터리에 있고 JMeter 스크립트 파일에서 입력 파일을 참조하는 경우 입력 파일의 경로는 이어야 합니다.INPUT_FILES. 대신 /home/user/INPUT_FILES를 사용하면 입력 파일을 찾을 수 없으므로 테스트가 실패합니다.

JMeter 플러그인을 포함하는 경우 zip 파일의 루트 내에서 .jar 파일을 /plugins라는 하위 디렉터리에 번들링해야 합니다. zip 파일의 루트를 기준으로 jar 파일의 경로는 ./plugins/BUNDLED_PLUGIN.jar여야 합니다.

JMeter 스크립트를 사용하는 방법에 대한 자세한 내용은 JMeter 사용 설명서를 참조하세요.

K6 테스트

이 솔루션은 K6 프레임워크 기반 테스트를 지원합니다. K6는 AGPL-3.0 라이선스에 따라 릴리스됩니다. 솔루션은 새 K6 테스트를 생성할 때 라이선스 승인 메시지를 표시합니다. K6 테스트 파일을 필요한 입력 파일과 함께 아카이브 파일에 업로드할 수 있습니다.

> **Important**
>
> K6 스크립트는 동시성(가상 사용자), 단계, 임계값 및 기타 로드 파라미터를 정의할 수 있지만 솔루션은 테스트 생성 중에 트래픽 셰이프 화면에서 지정한 값으로 이러한 구성을 재정의합니다. 트래픽 셰이프 구성은 테스트 실행의 작업 수, 동시성(작업당 가상 사용자 수), 증가 기간 및 대기 기간을 제어합니다.

Locust 테스트

이 솔루션은 Locust 프레임워크 기반 테스트를 지원합니다. 아카이브 파일에 필요한 입력 파일과 함께 Locust 테스트 파일을 업로드할 수 있습니다.

> **Important**
>
> Locust 스크립트는 동시성(사용자 수), 생성 속도 및 기타 로드 파라미터를 정의할 수 있지만 솔루션은 테스트 생성 중에 트래픽 셰이프 화면에서 지정한 값으로 이러한 구성을 재정의합니다. 트래픽 셰이프 구성은 테스트 실행의 작업 수, 동시성(작업당 가상 사용자 수), 증가 기간 및 대기 기간을 제어합니다.

테스트 예약

이 솔루션은 로드 테스트를 실행하기 위한 세 가지 실행 타이밍 옵션을 제공합니다.

- 지금 실행 - 생성 직후 로드 테스트 실행
- 한 번 실행 - 향후 특정 날짜 및 시간에 테스트를 실행합니다.
- 일정에 따라 실행 - cron 표현식을 사용하여 반복 테스트를 생성하여 일정을 정의합니다.

한 번 실행을 선택하면 실행 시간을 24시간 형식으로 지정하고 로드 테스트 실행을 시작해야 하는 실행 날짜를 지정합니다.

일정에 따라 실행을 선택하면 cron 표현식을 수동으로 입력하거나 일반적인 cron 패턴(예: 매시간, 매일 특정 시간, 평일 또는 매월) 중에서 선택할 수 있습니다. cron 표현식은 분, 시간, 월, 월, 요일 및 연도에 대한 필드와 함께 세분화된 일정 형식을 사용합니다. 또한 예약된 테스트 실행을 중지해야 하는 시기를 정의하는 만료 날짜도 지정해야 합니다. 예약의 작동 방식에 대한 자세한 내용은이 가이드의 예약 테스트 워크플로 섹션을 참조하세요.

> **Note**
>
> - 테스트 기간: 예약 시 총 테스트 기간을 고려합니다. 예를 들어, 램프 업 시간이 10분이고 대기 시간이 40분인 테스트는 완료하는 데 약 80분이 걸립니다.
> - 최소 간격: 예약된 테스트 사이의 간격이 예상 테스트 기간보다 긴지 확인합니다. 예를 들어 테스트에 약 80분이 걸리는 경우 3시간마다 실행되도록 예약합니다.
> - 시간당 제한: 예상 테스트 기간이 1시간 미만인 경우에도 시스템에서는 1시간 차이로만 테스트를 예약할 수 없습니다.

동시 테스트

이 솔루션은 Amazon ECS 클러스터에서 실행 중인 모든 작업의 결합된 출력을 실시간으로 표시하는 각 테스트에 대한 Amazon CloudWatch 대시보드를 생성합니다. CloudWatch 대시보드에는 평균 응답 시간, 동시 사용자 수, 성공한 요청 수, 실패한 요청 수가 표시됩니다. 솔루션은 각 지표를 초 단위로 집계하고 1분마다 대시보드를 업데이트합니다.

사용자 관리

초기 구성 중에 Amazon Cognito가 솔루션의 웹 콘솔에 대한 액세스 권한을 부여하는 데 사용하는 사용자 이름과 이메일 주소를 제공합니다. 콘솔은 사용자 관리를 제공하지 않습니다. 사용자를 추가하려면 Amazon Cognito 콘솔을 사용해야 합니다. 자세한 내용은 Amazon Cognito 개발자 안내서의 사용자 풀에서 사용자 관리를 참조하세요.

기존 사용자를 Amazon Cognito 사용자 풀로 마이그레이션하려면 사용자를 Amazon Cognito 사용자 풀로 마이그레이션하기 위한 AWS 블로그 접근 방식을 참조하세요.

리전 배포

이 솔루션은 특정 AWS 리전에서만 사용할 수 있는 Amazon Cognito를 사용합니다. 따라서 Amazon Cognito를 사용할 수 있는 리전에이 솔루션을 배포해야 합니다. 리전별 최신 서비스 가용성은 AWS 리전 서비스 목록을 참조하세요.

배포 계획

이 섹션에서는 솔루션을 배포하기 전에 검토해야 하는 비용, 보안, 지원되는 리전, 할당량 및 기타 고려 사항에 대해 설명합니다.

비용

이 솔루션을 실행하는 동안 사용되는 AWS 서비스의 비용은 사용자의 책임입니다. 총 비용은 실행된 로드 테스트 수, 해당 테스트 기간 및 생성된 데이터의 양에 따라 달라집니다. 이 개정부터 미국 동부(버지니아 북부) 리전의 기본 설정으로이 솔루션을 실행하는 데 드는 예상 비용은 매월 약 30.90 USD입니다.

다음 표에서는 1개월 동안 미국 동부(버지니아 북부) 리전의 기본 파라미터를 사용하여이 솔루션을 배포하기 위한 샘플 비용 분석을 제공합니다.

AWS 서비스	측정 기준	비용[USD]
AWS Fargate	30시간 동안 실행되는 온디맨드 작업 10개(vCPUs 및 4GB 메모리 사용)	29.62 USD
Amazon DynamoDB	온디맨드 쓰기 용량 단위 1,000개 온디맨드 읽기 용량 단위 1,000개	0.0015 USD
AWS Lambda	요청 1,000개 총 기간 10분	1.25 USD
AWS Step Functions	1,000개의 상태 전환	0.025 USD
합계:		매월 30.90 USD

솔루션 리소스에는 Key=SolutionId 및 Value=SO0062로 태그가 지정됩니다. 설명서 activating-tags에 따라 태그 키 SolutionId를 활성화할 수 있습니다. https://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/activating-tags.html 태그가 활성화되면 설명서에 따라 비용 범주를 생성하여 비용 범주

규칙을 생성할 수 있습니다. 비용 범주 콘솔을 모니터링하고 비용 범주 이름을 선택하여 솔루션에 발생한 비용을 볼 수 있습니다.

비용 관리에 도움이 되도록 AWS Cost Explorer를 통해 예산을 생성하는 것이 좋습니다. 요금은 변경될 수 있습니다. 자세한 내용은 이 솔루션에 사용되는 각 AWS 서비스의 요금 웹 페이지를 참조하세요.

> **Note**
>
> 기본 작업 구성은 작업당 vCPUs개와 메모리 4GB를 사용합니다. 로드 테스트에 이러한 리소스가 필요하지 않은 경우 리소스를 줄여 비용을 절감할 수 있습니다. 반대로 리소스를 늘려 작업당 더 높은 동시성을 지원할 수 있습니다. 자세한 내용은이 가이드의 컨테이너 리소스 증가 섹션을 참조하세요.

> **Note**
>
> 이 솔루션은 테스트를 실행할 때 라이브 데이터를 포함하는 옵션을 제공합니다. 이 기능을 사용하려면 추가 비용이 발생하는 추가 AWS Lambda 함수와 AWS IoT Core 주제가 필요합니다.

MCP Server 추가 비용(선택 사항)

다음 표에는 1개월 동안 미국 동부(버지니아 북부) 리전의 요금과 MCP Server 통합에 대한 비용 내역이 나와 있습니다.

서비스 구성 요소	측정 기준	비용[USD]
AgentCore 게이트웨이 - 도구 인덱싱	도구 10개 × 도구 100개당 0.02 USD	0.002 USD
AgentCore Gateway - 검색 API	10,000건의 상호 작용 × 1,000명당 0.025 USD	0.25 USD
AgentCore 게이트웨이 - API 호출	50,000회 호출 × 1,000명당 0.005 USD	0.25 USD
AWS Lambda 함수	사용량에 따른 변수(일반 워크로드)	5.00 USD~20.00 USD

서비스 구성 요소	측정 기준	비용[USD]
총 예상 추가 비용:		매월 5.50~20.50 USD

요금은 변경될 수 있습니다. AgentCore Gateway 요금에 대한 자세한 내용은 Amazon Bedrock 요금(AgentCore Gateway 섹션)을 참조하세요. Lambda 요금은 AWS Lambda 요금을 참조하세요.

보안

AWS 인프라에 시스템을 빌드하면 보안 책임은 사용자와 AWS가 분담합니다. 이 공동 책임 모델은 운영 부담을 덜어줍니다. AWS가 호스트 운영 체제, 가상화 계층, 서비스 운영 시설의 물리적 보안을 포함한 구성 요소를 운영, 관리, 제어하기 때문입니다. AWS 보안에 대한 자세한 내용은 AWS 클라우드 보안을 참조하세요.

IAM 역할

AWS Identity and Access Management(IAM) 역할을 통해 고객은 AWS 클라우드의 서비스 및 사용자에게 세분화된 액세스 정책 및 권한을 할당할 수 있습니다. 이 솔루션은 솔루션의 AWS Lambda 함수에 리전 리소스를 생성할 수 있는 액세스 권한을 부여하는 IAM 역할을 생성합니다.

Amazon CloudFront

이 솔루션은 Amazon CloudFront에서 배포하는 Amazon S3 버킷에 호스팅된 웹 UI를 배포합니다. Amazon CloudFront 지연 시간을 줄이고 보안을 강화하기 위해이 솔루션에는 솔루션 웹 사이트의 버킷 콘텐츠에 대한 퍼블릭 액세스를 제공하는 CloudFront 사용자인 오리진 액세스 ID가 있는 CloudFront 배포가 포함됩니다. 기본적으로 CloudFront 배포는 TLS 1.2를 사용하여 최고 수준의 보안 프로토콜을 적용합니다. 자세한 내용은 Amazon CloudFront 개발자 안내서의 Amazon S3 오리진에 대한 액세스 제한을 참조하세요. Amazon CloudFront

CloudFront는 추가 보안 완화를 활성화하여 각 최종 사용자 응답에 HTTP 보안 헤더를 추가합니다. 자세한 내용은 CloudFront 응답에서 HTTP 헤더 추가 또는 제거를 참조하세요.

이 솔루션은 TLS v1.0의 최소 지원 보안 프로토콜이 있는 기본 CloudFront 인증서를 사용합니다. TLS v1.2 또는 TLS v1.3 사용을 적용하려면 기본 CloudFront 인증서 대신 사용자 지정 SSL 인증서를 사용해야 합니다. 자세한 내용은 SSL/TLS 인증서를 사용하도록 CloudFront 배포를 구성하려면 어떻게 해야 합니까?를 참조하세요.

Amazon API Gateway

이 솔루션은 엣지 최적화 Amazon API Gateway 엔드포인트를 배포하여 사용자 지정 도메인이 아닌 기본 APIs Gateway 엔드포인트를 사용하여 로드 테스트 기능에 대한 RESTful API를 제공합니다. 기본 엔드포인트를 사용하는 엣지 최적화 APIs의 경우 API Gateway는 TLS-1-0 보안 정책을 사용합니다. 자세한 내용은 Amazon API Gateway APIs.

이 솔루션은 TLS v1.0의 최소 지원 보안 프로토콜이 있는 기본 API Gateway 인증서를 사용합니다. TLS v1.2 또는 TLS v1.3 사용을 적용하려면 기본 API Gateway 인증서 대신 사용자 지정 SSL 인증서가 있는 사용자 지정 도메인을 사용해야 합니다. 자세한 내용은 REST APIs.

AWS Fargate 보안 그룹

기본적으로이 솔루션은 AWS Fargate 보안 그룹의 아웃바운드 규칙을 퍼블릭으로 엽니다. AWS Fargate가 어디서나 트래픽을 전송하지 못하도록 차단하려면 아웃바운드 규칙을 특정 Classless Inter-Domain Routing(CIDR)으로 변경합니다.

또한이 보안 그룹에는 포트 50,000의 로컬 트래픽을 동일한 보안 그룹에 속하는 모든 소스로 허용하는 인바운드 규칙이 포함되어 있습니다. 이는 컨테이너가 서로 통신할 수 있도록 하는 데 사용됩니다.

Amazon VPC

VPC: Amazon VPC 서비스를 기반으로 하는 Virtual Private Cloud(VPC)는 AWS 클라우드에서 논리적으로 격리된 프라이빗 네트워크를 제공합니다.

배포 중에 AWS CloudFormation 파라미터에서 자체 VPC를 지정할 수 있습니다. VPC는 로드를 생성하는 ECS 작업에서만 사용되며 웹 콘솔과 API는이 VPC 내에 배포되지 않습니다. 기존 VPC를 지정하지 않으면 솔루션이 필요한 네트워킹 구성으로 새 VPC를 생성합니다. 기존 VPC를 사용하기로 선택한 경우 로드 테스트 작업을 성공적으로 실행하려면 다음 요구 사항을 충족해야 합니다.

VPC 요구 사항

AWS에서 분산 로드 테스트와 함께 사용할 VPC의 최소 요구 사항은 다음과 같습니다.

- VPC에는 최소 2개의 AZs 포함되어야 합니다.
- VPC에는 각각 별도의 AZ에 있는 서브넷이 두 개 이상 포함되어야 합니다.
- VPC 서브넷은 퍼블릭 또는 프라이빗일 수 있지만 동일한 구성(둘 다 퍼블릭 또는 둘 다 프라이빗)을 사용해야 합니다.
- VPC는 ECR, CloudWatch Logs, S3 및 IoT Core의 엔드포인트에 대한 액세스를 제공해야 합니다.
- VPC는 로드 테스트의 대상이 되는 서비스(들)에 대한 액세스를 제공해야 합니다.

> **Note**
>
> 이러한 기준을 충족하는 VPC가 없는 경우 VPC 마법사를 사용하여 VPC를 빠르게 생성할 수 있습니다. 자세한 내용은 VPC 생성을 참조하세요.

퍼블릭 서브넷은 다음을 포함하여 이러한 요구 사항을 충족할 수 있습니다.

- VPC에 연결된 인터넷 게이트웨이
- 인터넷 게이트웨이에 대한 경로(0.0.0.0/0)

프라이빗 서브넷은 아래 설명된 대로 NAT 게이트웨이 또는 VPC 엔드포인트를 사용하여 이러한 요구 사항을 충족할 수 있습니다.

옵션 1: NAT 게이트웨이

- 프라이빗 서브넷이 있는 각 AZ에 NAT 게이트웨이 배포
- NAT 게이트웨이를 통해 인터넷 바운드 트래픽(0.0.0.0/0)을 라우팅하도록 라우팅 테이블 구성

옵션 2: VPC 엔드포인트

VPC에서 다음 VPC 엔드포인트를 생성합니다.

- Amazon ECR API 엔드포인트: `com.amazonaws.<region>.ecr.api`
- Amazon ECR DKR 엔드포인트: `com.amazonaws.<region>.ecr.dkr`
- Amazon CloudWatch Logs 엔드포인트: `com.amazonaws.<region>.logs`
- Amazon S3 Gateway 엔드포인트: `com.amazonaws.<region>.s3`
- AWS IoT Core 엔드포인트(라이브 데이터 차트를 사용하는 경우 필수) `com.amazonaws.<region>.iot.data`

다른 VPC 구성도 작동할 수 있습니다.

> **Important**
>
> 각 VPC 엔드포인트 인터페이스에 연결된 보안 그룹은 ECS 태스크 보안 그룹에서 포트 443의 인바운드 TCP 트래픽을 허용해야 합니다.

보안 그룹 구성

배포 중에 솔루션은 VPC 내에 보안 그룹을 생성하여 ECS 클러스터의 작업으로 다음 트래픽을 허용합니다.

- 모든 아웃바운드 트래픽
- 동일한 보안 그룹의 다른 작업에서 포트 50000의 인바운드 트래픽으로 작업자와 리더 작업 간의 조정을 용이하게 합니다.

네트워크 스트레스 테스트

네트워크 스트레스 테스트 정책에 따라이 솔루션을 사용할 책임은 사용자에게 있습니다. 이 정책은 Amazon EC2 인스턴스에서 다른 Amazon EC2 인스턴스Amazon EC2, AWS 속성/서비스 또는 외부 엔드포인트와 같은 다른 위치로 직접 대용량 네트워크 테스트를 실행하려는 경우와 같은 상황을 다룹니다. 이러한 테스트를 스트레스 테스트, 로드 테스트 또는 게임데이 테스트라고도 합니다. 대부분의 고객 테스트는이 정책에 해당되지 않습니다. 그러나 1분 이상, 1Gbps(초당 10억 비트) 또는 1Gpps(초당 10억 패킷)를 초과하는 트래픽을 총 1분 이상 지속할 것으로 생각되면이 정책을 참조하세요.

퍼블릭 사용자 인터페이스에 대한 액세스 제한

IAM 및 Amazon Cognito에서 제공하는 인증 및 권한 부여 메커니즘을 넘어 퍼블릭 사용자 인터페이스에 대한 액세스를 제한하려면 AWS WAF(웹 애플리케이션 방화벽) 보안 자동화 솔루션을 사용합니다.

이 솔루션은 일반적인 웹 기반 공격을 필터링하는 AWS WAF 규칙 세트를 자동으로 배포합니다. 사용자는 AWS WAF 웹 액세스 제어 목록(웹 ACL)에 포함된 규칙을 정의하는 사전 구성된 보호 기능 중에서 선택할 수 있습니다.

MCP 서버 보안(선택 사항)

선택적 MCP 서버 통합을 배포하는 경우 솔루션은 AWS AgentCore Gateway를 사용하여 AI 에이전트의 로드 테스트 데이터에 대한 보안 액세스를 제공합니다. AgentCore Gateway는 각 요청에 대한 Amazon Cognito 인증 토큰을 검증하여 권한이 있는 사용자만 MCP 서버에 액세스할 수 있도록 합니다. MCP Server Lambda 함수는 읽기 전용 액세스 패턴을 구현하여 AI 에이전트가 테스트 구성 또는 결과를 수정하지 못하도록 합니다. 모든 MCP 서버 상호 작용은 웹 콘솔과 동일한 권한 경계 및 액세스 제어를 사용합니다.

지원되는 AWS 리전

이 솔루션은 현재 일부 AWS 리전에서 사용할 수 없는 Amazon Cognito 서비스를 사용합니다. 리전별 AWS 서비스의 최신 가용성은 AWS 리전 서비스 목록을 참조하세요.

AWS의 분산 로드 테스트는 다음 AWS 리전에서 사용할 수 있습니다.

리전 이름	
미국 동부(오하이오)	아시아 태평양(도쿄)
미국 동부(버지니아 북부)	캐나다(중부)
미국 서부(캘리포니아 북부)	유럽(프랑크푸르트)
미국 서부(오리곤)	유럽(아일랜드)
아시아 태평양(뭄바이)	유럽(런던)
아시아 태평양(서울)	유럽(파리)
아시아 태평양(싱가포르)	유럽(스톡홀름)
아시아 태평양(시드니)	남아메리카(상파울루)

MCP 서버 지원 AWS 리전(선택 사항)

선택적 MCP Server 통합을 배포하려는 경우 AWS AgentCore Gateway를 사용할 수 있는 AWS 리전에 솔루션을 배포해야 합니다. MCP Server 기능은 다음 AWS 리전에서만 사용할 수 있습니다.

리전 이름	리전 코드
미국 동부(버지니아 북부)	us-east-1
미국 서부(오리건)	us-west-2
아시아 태평양(싱가포르)	ap-southeast-1
아시아 태평양(시드니)	ap-southeast-2

리전 이름	리전 코드
아시아 태평양(도쿄)	ap-northeast-1
유럽(프랑크푸르트)	eu-central-1
유럽(아일랜드)	eu-west-1
유럽(런던)	eu-west-2
유럽(파리)	eu-west-3

리전별 AWS AgentCore Gateway의 최신 가용성은 AWS AgentCore Gateway 개발자 안내서의 AWS AgentCore Gateway 엔드포인트 및 할당량을 참조하세요. AgentCore

할당량

서비스 할당량(제한이라고도 함)은 AWS 계정의 최대 서비스 리소스 또는 작업 수입니다.

이 솔루션의 AWS 서비스에 대한 할당량

이 솔루션에 구현된 각 서비스의 할당량이 충분한지 확인하세요. 자세한 내용은 AWS 서비스 할당량을 참조하세요.

다음 링크를 선택하여 해당 서비스에 대한 페이지로 이동합니다. 페이지를 전환하지 않고 설명서의 모든 AWS 서비스에 대한 서비스 할당량을 보려면 PDF 대신 서비스 엔드포인트 및 할당량 페이지에서 정보를 확인하세요.

AWS CloudFormation 할당량

AWS 계정에는 이 솔루션에서 스택을 시작할 때 알아두어야 하는 AWS CloudFormation 할당량이 있습니다. 이러한 할당량을 이해하면 이 솔루션을 성공적으로 배포하지 못하는 제한 오류를 방지할 수 있습니다. 자세한 내용은 AWS CloudFormation 사용 설명서의에서 AWS CloudFormation 할당량을 참조하세요 AWS CloudFormation.

로드 테스트 할당량

AWS Fargate 시작 유형을 사용하여 Amazon ECS에서 실행할 수 있는 최대 태스크 수는 태스크의 vCPU 크기를 기반으로 합니다. AWS의 분산 로드 테스트의 기본 태스크 크기는 vCPU 2개입니다. 현

재 기본 할당량을 보려면 Amazon ECS 서비스 할당량을 참조하세요. 현재 계정 할당량은 나열된 할당량과 다를 수 있습니다. 계정별 할당량을 확인하려면 AWS Management Console에서 Fargate 온디맨드 vCPU 리소스 수에 대한 서비스 할당량을 확인합니다. 증가를 요청하는 방법에 대한 지침은 AWS 일반 참조 안내서의 AWS 서비스 할당량을 참조하세요.

Amazon Linux 2023 컨테이너 이미지(Taurus 설치됨)는 작업당 동시 연결을 제한하지 않지만 무제한 사용자 수를 지원할 수 있음을 의미하지는 않습니다. 컨테이너가 테스트를 위해 생성할 수 있는 동시 사용자 수를 확인하려면이 가이드의 사용자 수 결정 섹션을 참조하세요.

> **Note**
>
> 기본 설정을 기반으로 한 동시 사용자의 권장 제한은 200명입니다.

동시 테스트

이 솔루션은 Amazon ECS 클러스터에서 실행 중인 모든 작업의 결합된 출력을 실시간으로 표시하는 각 테스트에 대한 Amazon CloudWatch 대시보드를 생성합니다. CloudWatch 대시보드에는 평균 응답 시간, 동시 사용자 수, 성공한 요청 수, 실패한 요청 수가 표시됩니다. 솔루션은 각 지표를 초 단위로 집계하고 1분마다 대시보드를 업데이트합니다.

Amazon EC2 테스트 정책

네트워크 트래픽이 1Gbps 미만으로 유지되는 한이 솔루션을 사용하여 로드 테스트를 실행하기 위해 AWS의 승인이 필요하지 않습니다. 테스트에서 1Gbps 이상이 생성되면 AWS에 문의하십시오. 자세한 내용은 Amazon EC2 테스트 정책을 참조하세요.

Amazon CloudFront 로드 테스트 정책

CloudFront 엔드포인트를 로드 테스트하려는 경우 Amazon CloudFront 개발자 안내서의 로드 테스트 지침을 참조하세요. 또한 트래픽을 여러 작업 및 리전에 분산하는 것이 좋습니다. 로드 테스트에 최소 30분의 램프업 시간을 제공합니다. 초당 500,000개 이상의 요청을 보내거나 300Gbps 이상의 데이터를 요구하는 로드 테스트의 경우 먼저 트래픽 전송에 대한 사전 승인을 받는 것이 좋습니다. CloudFront는 CloudFront 서비스 가용성에 영향을 미치는 승인되지 않은 로드 테스트 트래픽을 제한할 수 있습니다.

배포 후 솔루션 모니터링

솔루션을 배포한 후에는 Amazon CloudWatch 경보 및 지표를 사용하여 솔루션의 리소스를 지속적으로 모니터링하는 것이 좋습니다.

CloudWatch 경보 설정

CloudWatch 경보를 설정하여 주요 지표를 모니터링하고 임계값이 초과되면 알림을 받을 수 있습니다. 다음 리소스에 대한 경보를 설정하는 것이 좋습니다.

Amazon CloudFront 배포 지표

CloudFront 배포 성능 및 오류를 모니터링합니다. 자세한 내용은 Amazon CloudFront 개발자 안내서의 CloudFront 배포 지표를 참조하세요. Amazon CloudFront

Amazon API Gateway 지표

API 요청 속도, 지연 시간 및 오류를 모니터링합니다. 자세한 내용은 Amazon API Gateway 개발자 안내서의 Amazon API Gateway 차원 및 지표를 참조하세요. Amazon API Gateway

AWS Lambda 함수 지표

솔루션의 마이크로서비스에 대한 Lambda 함수 호출, 기간, 오류 및 스로틀을 모니터링합니다.

Amazon ECS 및 AWS Fargate 지표

로드 테스트 중에 작업 CPU 및 메모리 사용률을 모니터링하여 적절한 리소스를 확보합니다.

Amazon DynamoDB 지표

읽기 및 쓰기 용량 소비, 제한된 요청 및 지연 시간을 모니터링합니다.

전문가 참여

AWS에서 분산 로드 테스트를 위한 AWS Countdown Premium 단기 계약

AWS 엔지니어는 성능 테스트 기본 사항, 스크립트 개발 및 결과 분석에 대한 전문가 지침을 제공합니다. 지금 가입하세요.

개요

AWS Countdown Premium(CDP) 단기 참여는 대규모 성능 테스트를 수행하는 조직에 전문가 지침을 제공합니다. AWS 엔지니어는 공동 'do-it-yourself' 모델을 통해 전략적 감독 및 기술 전문 지식을 제공하는 동시에 팀이 실행 책임을 유지합니다. 전문가 AWS 엔지니어는 가입 후 1주일 이내에 장기 계약 없이 이용할 수 있습니다.

서비스 모델

CDP 엔지니어는 팀과 협력하여 성능 테스트 구현 전반에 걸쳐 지침과 감독을 제공합니다. 이 핸드오프 접근 방식을 사용하면 내부 기능을 구축하는 동시에 전문가의 지시를 받을 수 있습니다. 이 서비스는 AWS에서 분산 로드 테스트를 효과적으로 구현하기 위해 전문 AWS 전문 지식이 필요한 기존 테스트 기능을 갖춘 조직에 적합합니다.

CDP 엔지니어가 제공하는 내용

CDP 엔지니어는 성능 테스트 기본 사항과 AWS 아키텍처의 분산 로드 테스트를 안내합니다. JMeter, K6 및 Locust 스크립트 구조 및 테스트 스크립트 개발에 대한 지침을 제공하고, CloudFormation 템플릿 배포를 지원하고, 성능 최적화 권장 사항을 사용하여 테스트 결과를 평가합니다. 지원에는 리소스 사용률 분석, 모범 사례 조정, 초기 설정부터 결과 분석까지 end-to-end 지침이 포함되어 있으므로 팀에 지식을 전달할 수 있습니다.

고객의 책임

팀은 애플리케이션 수준 구성, 테스트 스크립트 개발 및 테스트 시나리오 확인을 처리합니다. 성능 테스트 이벤트 이전, 도중 및 이후의 모든 테스트 활동을 포함하여 실제 테스트 실행 및 작업에 대한 책임은 사용자에게 있습니다.

주요 이점

CDP 단기 참여는 팀의 소유권 및 역량 개발을 유지하면서 전문가 감독, 워크로드별 상황별 지침, 성능 최적화 권장 사항, 더 빠른 문제 해결, 모범 사례 조정, 포괄적인 지원을 통해 위험을 줄입니다.

지원되는 아키텍처

AWS의 분산 로드 테스트는 AWS의 분산 로드 테스트를 활용하여 대규모 웹 애플리케이션, APIs, 마이크로서비스 및 서버리스 아키텍처에 대한 테스트를 지원합니다. 테스트 기능은 이러한 일반적인 사용 사례를 훨씬 넘어 데이터베이스, TCP/UDP 프로토콜, LDAP 디렉터리, SMTP 메일 서버, 로드 시 성능 검증이 필요한 기타 여러 시스템 및 프로토콜을 포함합니다.

시작하기

AWS의 분산 로드 테스트를 위한 CDP 단기 참여에 관심이 있는 조직은 여기 AWS 웹 사이트를 통해 직접 가입하고 중점 영역에 대해 "사용 사례 구현"을 선택할 수 있습니다.

범위를 벗어남

CDP는 사용자 지정 테스트 스크립트 개발(지침만 해당)을 제공하거나, 테스트 실행 작업을 관리하거나, 사용자 지정 실습 또는 워크숍을 생성하지 않습니다. 현장 지원도 범위를 벗어납니다.

솔루션 배포

AWS Launch Wizard는이 솔루션에 권장되는 배포 방법입니다. 이 커널은 다음을 제공합니다.

- 각 단계의 세부 도움말 패널이 포함된 안내 구성 경험
- 모든 배포의 상태를 모니터링하는 중앙 집중식 페이지
- 배포 또는 업그레이드에 사용할 수 있는 솔루션의 최신 버전이 있는 경우의 표시

또는 AWS CloudFormation 템플릿을 사용하여 솔루션을 직접 배포할 수 있습니다.

배포 프로세스 개요

솔루션을 배포하기 전에이 가이드의 앞부분에서 설명한 비용, 아키텍처, 보안 및 기타 고려 사항을 검토하세요.

배포 시간: 기본 스택의 경우 약 15분, 각 추가 리전의 경우 5분

> **Note**
>
> 이 솔루션에는 AWS에 대한 데이터 수집 지표가 포함되어 있습니다. 당사는 이 데이터를 사용하여 고객이 이 솔루션과 관련 서비스 및 제품을 어떻게 사용하는지 더 잘 이해합니다. AWS는 이 설문 조사를 통해 수집된 데이터를 소유합니다. 데이터 수집에는 AWS 개인 정보 보호 고지가 적용됩니다.

> **Note**
>
> 이 솔루션을 실행하는 동안 사용되는 AWS 서비스의 비용은 사용자의 책임입니다. 자세한 내용은이 가이드의 비용 섹션을 방문하여이 솔루션에 사용되는 각 AWS 서비스의 요금 웹 페이지를 참조하세요.

AWS Launch Wizard를 사용하여 배포

이 솔루션은 AWS Launch Wizard를 사용하는 안내 배포 프로세스를 제공합니다. 다음 단계에 따라 AWS에서 분산 로드 테스트를 계정에 배포합니다.

1. AWS Management Console에 로그인하고 아래 버튼을 선택하여 배포 프로세스를 시작합니다.

2. 솔루션에 사용할 수 있는 배포 패턴이 두 개 이상인 경우 사용 사례에 가장 적합한 배포 패턴을 선택합니다.
3. 배포할 버전을 선택합니다. 최신 버전이 권장됩니다.
4. 배포 시작 마법사 버튼을 클릭합니다.

그런 다음 일련의 단계에 따라 솔루션을 배포하는 데 필요한 정보를 수집합니다. 필요한 리소스를 프로비저닝하는 데 약 15분이 걸립니다.

배포 목록에서 배포를 선택하여 배포 상태를 확인합니다.

AWS CloudFormation을 사용하여 배포

이 솔루션은 AWS CloudFormation 템플릿 및 스택을 사용하여 솔루션의 배포를 자동화합니다. CloudFormation 템플릿은 이 솔루션에 포함된 AWS 리소스와 해당 속성을 지정합니다. CloudFormation 스택은 템플릿에 설명된 리소스를 프로비저닝합니다.

AWS CloudFormation 템플릿

배포하기 전에이 솔루션에 대한 CloudFormation 템플릿을 다운로드할 수 있습니다. 이 솔루션은 AWS CloudFormation을 사용하여 AWS에서 분산 로드 테스트의 배포를 자동화합니다. 여기에는 배포 전에 다운로드할 수 있는 다음 AWS CloudFormation 템플릿이 포함되어 있습니다.

distributed-load-testing-on-aws.template -이 템플릿을 사용하여 솔루션 및 모든 관련 구성 요소를 시작합니다. 기본 구성은 이 솔루션 섹션의 AWS 서비스에 있는 코어 및 지원 서비스를 배포하지만 특정 요구 사항에 맞게 템플릿을 사용자 지정할 수 있습니다.

> **Note**
>
> AWS CloudFormation 리소스는 AWS Cloud Development Kit(AWS CDK) 구문에서 생성됩니다. 이전에이 솔루션을 배포한 경우 업데이트 지침은 솔루션 업데이트를 참조하세요.

스택 시작

다음 단계에 따라 AWS의 분산 로드 테스트를 계정에 배포합니다. 이 자동화된 AWS CloudFormation 템플릿은 AWS에 분산 로드 테스트를 배포합니다.

1. AWS Management Console에 로그인하고 버튼을 선택하여 CloudFormation 템플릿을 시작합니다.

   또는 템플릿을 자체 구현의 시작점으로 다운로드할 수 있습니다.
2. 이 템플릿은 기본적으로 미국 동부(버지니아 북부) 리전에서 실행됩니다. 다른 AWS 리전에서이 솔루션을 시작하려면 콘솔 탐색 모음에서 리전 선택기를 사용합니다.

   > **Note**
   >
   > 이 솔루션은 현재 특정 AWS 리전에서만 사용할 수 있는 Amazon Cognito를 사용합니다. 따라서 Amazon Cognito를 사용할 수 있는 AWS 리전에서이 솔루션을 시작해야 합니다. 리전별 최신 서비스 가용성은 AWS 리전 서비스 목록을 참조하세요.
3. 스택 생성 페이지에서 Amazon S3 URL 텍스트 상자에 올바른 템플릿 URL이 표시되는지 확인하고 다음을 선택합니다.
4. 스택 세부 정보 지정 페이지에서 솔루션 스택 이름을 할당합니다.
5. 파라미터에서 템플릿의 파라미터를 검토하고 필요에 따라 수정합니다. 이 솔루션은 다음과 같은 기본값을 사용합니다.

파라미터	기본값	설명
관리자 이름	<입력 필수>	초기 솔루션 관리자의 사용자 이름입니다.
관리자 이메일	*<## ##>*	관리자 사용자의 이메일 주소입니다. 시작 후 콘솔 로그인 지침이 포함된 이메일이이 주소로 전송됩니다.
기존 VPC ID	<선택 사항 입력>	사용하려는 VPC가 있고 이미 생성된 경우 스택이 배포된 리

파라미터	기본값	설명
		전과 동일한 리전에 있는 기존 VPC의 ID를 입력합니다. 예: vpc-1a2b3c4d5e6f.
첫 번째 기존 서브넷	<선택 사항 입력>	기존 VPC 내 첫 번째 서브넷의 ID입니다. 이 서브넷은 테스트를 실행하기 위해 컨테이너 이미지를 가져오려면 인터넷에 대한 경로가 필요합니다. 예: subnet-7h8i9j0k.
두 번째 기존 서브넷	<선택 사항 입력>	기존 VPC 내 두 번째 서브넷의 ID입니다. 이 서브넷은 테스트를 실행하기 위해 컨테이너 이미지를 가져오려면 인터넷에 대한 경로가 필요합니다. 예: subnet-1x2y3z.
VPC를 생성할 솔루션에 유효한 CIDR 블록 제공	192.168.0.0/16	기존 VPC를 사용하는 경우이 파라미터를 비워 둘 수 있습니다.
솔루션에서 VPC를 생성할 수 있도록 서브넷 A에 유효한 CIDR 블록 제공	192.168.0.0/20	AWS Fargate VPC의 서브넷 A에 대한 CIDR 블록
솔루션에서 VPC를 생성할 수 있도록 서브넷 B에 유효한 CIDR 블록 제공	192.168.16.0/20	AWS Fargate VPC의 서브넷 B에 대한 CIDR 블록
Fargate 작업의 아웃바운드 트래픽을 허용하기 위한 CIDR 블록 제공	0.0.0.0/0	Amazon ECS 컨테이너 아웃바운드 액세스를 제한하는 CIDR 블록입니다.

파라미터	기본값	설명
컨테이너 이미지 자동 업데이트	No	다음 마이너 릴리스까지 최신 이미지를 자동으로 사용하고 이미지를 보호합니다. 선택하면 보안 업데이트 없이 원래 릴리스된 이미지를 No 가져옵니다.
선택적 MCP 서버 배포	No	AgentCore Gateway를 사용하여 AI 애플리케이션을 AWS의 Distributed Load Testing에 연결하여 선택적 원격 MCP 서버를 배포합니다.

6. 다음을 선택합니다.
7. 스택 옵션 구성 페이지에서 다음을 선택합니다.
8. 검토 페이지에서 설정을 검토하고 확인합니다. 템플릿이 AWS Identity and Access Management(IAM) 리소스를 생성할 것임을 승인하는 확인란을 선택합니다.
9. 스택 생성을 선택하여 스택을 배포합니다.

AWS CloudFormation 콘솔의 상태 열에서 스택의 상태를 볼 수 있습니다. 약 15분 후에 CREATE_COMPLETE 상태를 받게 됩니다.

> **Note**
>
> 이 솔루션에는 기본 AWS Lambda 함수 외에도 초기 구성 중에 또는 리소스가 업데이트되거나 삭제될 때만 실행되는 사용자 지정 리소스 Lambda 함수가 포함되어 있습니다.

이 솔루션을 실행하면 사용자 지정 리소스 Lambda 함수가 비활성화됩니다. 그러나 연결된 리소스를 관리하는 데 필요하므로이 함수를 삭제하지 마십시오.

다중 리전 배포

배포 시간: 리전당 약 5분

여러 리전에서 테스트를 실행할 수 있습니다.

Distributed Load Testing 솔루션을 배포하면 시나리오 S3 버킷에 리전 CloudFormation 템플릿이 생성됩니다. 이 템플릿의 URL은 기본 스택의 CloudFormation 출력에 "RegionalCFTemplate" 키 아래에 나열됩니다.

다중 리전 테스트를 실행하려면 테스트를 실행하려는 각 리전에 리전 CloudFormation 템플릿을 배포해야 합니다.

> **Note**
>
> 각 AWS 계정은 리전당 하나의 리전 스택만 사용할 수 있습니다. 또한 리전 스택은 기본 스택과 동일한 리전에서 사용할 수 없습니다.

다음과 같이 리전 템플릿을 설치할 수 있습니다.

1. 솔루션의 웹 콘솔에서 왼쪽 메뉴의 대시보드로 이동합니다.
2. 클립보드 아이콘을 사용하여 Amazon S3에서 CloudFormation 템플릿 링크를 복사합니다.
3. AWS CloudFormation 콘솔에 로그인하고 올바른 리전을 선택합니다.
4. 스택 생성 페이지에서 Amazon S3 URL 텍스트 상자에 올바른 템플릿 URL이 표시되는지 확인하고 다음을 선택합니다.
5. 스택 세부 정보 지정 페이지에서 솔루션 스택 이름을 할당합니다.
6. 파라미터에서 템플릿의 파라미터를 검토하고 필요에 따라 수정합니다. 이 솔루션은 다음과 같은 기본값을 사용합니다.

파라미터	기본값	설명
기존 VPC ID	<선택 사항 입력>	사용하려는 VPC가 있고 이미 생성된 경우 스택이 배포된 리전과 동일한 리전에 있는 기존 VPC의 ID를 입력합니다. 예: vpc-1a2b3c4d5e6f.
첫 번째 기존 서브넷	<선택 사항 입력>	기존 VPC 내 첫 번째 서브넷의 ID입니다. 이 서브넷은 테스트를 실행하기 위해 컨테이

파라미터	기본값	설명
		너 이미지를 가져오려면 인터넷에 대한 경로가 필요합니다. 예: subnet-7h8i9j0k.
두 번째 기존 서브넷	<선택 사항 입력>	기존 VPC 내 두 번째 서브넷의 ID입니다. 이 서브넷은 테스트를 실행하기 위해 컨테이너 이미지를 가져오기 위한 인터넷 경로가 필요합니다. 예: subnet-1x2y3z.
VPC를 생성할 솔루션에 유효한 CIDR 블록 제공	192.168.0.0/16	기존 VPC에 값을 제공하지 않으면 솔루션 생성 Amazon VPC의 CIDR 블록에 AWS Fargate의 IP 주소가 포함됩니다.
Fargate 작업의 아웃바운드 트래픽을 허용하기 위한 CIDR 블록 제공	0.0.0.0/0	Amazon ECS 컨테이너 아웃바운드 액세스를 제한하는 CIDR 블록입니다.

7. 다음을 선택합니다.
8. 스택 옵션 구성 페이지에서 다음을 선택합니다.
9. 검토 페이지에서 설정을 검토하고 확인합니다. 템플릿이 AWS Identity and Access Management(IAM) 리소스를 생성할 것임을 확인하는 확인란을 선택해야 합니다.
10. 스택 생성을 선택하여 스택을 배포합니다.

    AWS CloudFormation 콘솔의 상태 열에서 스택의 상태를 볼 수 있습니다. 약 5분 후에 CREATE_COMPLETE 상태를 받게 됩니다.

리전이 성공적으로 배포되면 웹 콘솔에 표시됩니다. 테스트를 생성하면 사용 가능한 모든 리전이 대시보드 및 시나리오 생성에 나열됩니다. 시나리오 생성의 트래픽 셰이프 단계에서 테스트에 리전을 추가할 수 있습니다.

솔루션은 시나리오 테이블에서 배포된 각 리전에 대해 DynamoDB 항목을 생성합니다. 여기에는 해당 리전의 테스트 리소스에 대한 필수 정보가 포함되어 있습니다. 웹 콘솔에서 리전별로 테스트 결과를 정

렬할 수 있습니다. 다중 리전 테스트에서 모든 리전의 집계 결과를 보려면 Amazon CloudWatch 지표를 사용합니다. 테스트가 완료된 후 테스트 결과에서 그래프의 소스 코드를 찾을 수 있습니다.

> **Note**
>
> 웹 콘솔 없이 리전 스택을 시작할 수 있습니다. Amazon S3 시나리오 버킷의 리전 템플릿 링크를 가져와서 필요한 리전에서 리전 스택을 시작할 때 소스로 제공합니다. 또는 템플릿을 다운로드하여 원하는 리전의 소스로 업로드할 수 있습니다.

솔루션 업데이트

솔루션을 업데이트하면 배포에 최신 기능, 보안 패치 및 버그 수정이 적용됩니다. 최신 버전으로 업데이트하려면 원래 배포 방법에 따라 적절한 섹션인 AWS Launch Wizard 또는 AWS CloudFormation을 참조하세요.

> **Important**
>
> 업데이트하기 전에 현재 실행 중인 로드 테스트가 없는지 확인합니다. 업데이트 프로세스로 인해 솔루션의 가용성이 일시적으로 중단될 수 있습니다.

AWS Launch Wizard를 사용하여 업데이트

콘솔은 배포 버전 드롭다운에 사용 가능한 최신 버전의 솔루션을 자동으로 표시합니다. 이전에 솔루션을 배포한 경우 다음 절차에 따라 배포를 최신 버전으로 업데이트합니다.

1. Launch Wizard Deployments로 이동합니다.
2. 업데이트할 배포를 선택합니다.
3. 작업을 선택한 다음 배포 버전 업데이트를 선택합니다.
4. 사용 가능한 배포 버전에서 최신 버전을 선택합니다.
5. 구성을 검토합니다.
6. 각 단계에서 필요한 사항을 변경합니다.
7. 업데이트를 확인합니다.

AWS CloudFormation을 사용하여 업데이트

이전에 솔루션을 배포한 경우 다음 절차에 따라 CloudFormation 스택을 최신 버전으로 업데이트합니다.

1. CloudFormation 콘솔에 로그인하고 기존 CloudFormation 스택을 선택한 다음 스택 업데이트를 선택합니다.
2. 직접 업데이트를 선택합니다.
3. 기존 템플릿 교체를 선택합니다.
4. 템플릿 지정에서 다음을 수행합니다.

a. Amazon S3 URL을 선택합니다.
b. 최신 템플릿의 링크를 복사합니다.
c. Amazon S3 URL 상자에 링크를 붙여넣습니다.
d. Amazon S3 URL 텍스트 상자에 올바른 템플릿 URL이 표시되는지 확인합니다.
e. 다음을 선택합니다.
f. 다음을 다시 선택합니다.

5. 파라미터에서 템플릿의 파라미터를 검토하고 필요에 따라 수정합니다. 파라미터에 대한 자세한 내용은 스택 시작을 참조하세요.
6. 다음을 선택합니다.
7. 스택 옵션 구성 페이지에서 다음을 선택합니다.
8. 검토 페이지에서 설정을 검토하고 확인합니다.
9. 템플릿이 IAM 리소스를 생성할 수 있음을 확인하는 확인란을 선택하세요.
10. 변경 세트 보기를 선택하고 변경 사항을 확인합니다.
11. 스택 생성을 선택하여 스택을 배포합니다.

AWS CloudFormation 콘솔의 상태 열에서 스택의 상태를 볼 수 있습니다. 약 15분 후에 `UPDATE_COMPLETE` 상태를 받게 됩니다.

> **Note**
>
> 스택 업그레이드 후 브라우저에서 로그인하는 동안 Amazon Cognito 인증 문제가 발생하는 경우 브라우저(Windows/Linux의 경우 Ctrl+Shift+R 또는 Mac의 경우 Cmd+Shift+R)를 새로 고쳐 캐시된 데이터를 지우고 다시 시도하세요.

v3.3.0 이전 버전의 업데이트 문제 해결

> **Note**
>
> 이 섹션은 v3.3.0 이전 버전의 업데이트에만 적용됩니다. v3.3.0 이상에서 업데이트하는 경우 AWS Launch Wizard 또는 AWS CloudFormation을 통해 표준 업데이트 절차를 따릅니다.

1. distributed-load-testing-on-aws.template을 다운로드합니다.

2. 템플릿을 열고 로 이동하여를 Conditions: 찾습니다DLTCommonResourcesAppRegistryCondition.
3. 다음과 유사한 결과가 출력되어야 합니다.

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

4. 두 번째 true 값을 로 변경합니다false.

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

5. 사용자 지정 템플릿을 사용하여 AWS CloudFormation을 사용하여 업데이트의 단계에 따라 스택을 업데이트합니다.
6. 이 업데이트는 스택에서 앱 레지스트리 관련 리소스를 제거하여 업데이트가 성공적으로 완료되도록 합니다.
7. 최신 템플릿 URL을 사용하여 다른 스택 업데이트를 수행합니다.

리전 스택 업데이트

솔루션을 여러 리전에 배포한 경우 각 리전 스택을 별도로 업데이트해야 합니다. 테스트 인프라를 배포한 리전의 각 리전 CloudFormation 스택에 대한 표준 업데이트 절차를 따릅니다.

AWS Systems Manager 애플리케이션 관리자

솔루션을 업데이트한 후 AWS Systems Manager Application Manager는 솔루션과 해당 리소스에 대한 애플리케이션 수준 보기를 제공합니다. Application Manager를 사용하여 다음을 수행할 수 있습니다.

- 리소스, 스택 및 AWS 계정에 배포된 리소스 비용, 중앙 위치의 로그를 모니터링합니다.
- 배포 상태, CloudWatch 경보, 리소스 구성 및 운영 문제와 같은 애플리케이션의 컨텍스트에서 솔루션 리소스에 대한 작업 데이터를 봅니다.

문제 해결

알려진 문제 해결은 알려진 오류를 완화하기 위한 지침을 제공합니다. 이러한 지침으로 문제가 해결되지 않는 경우 AWS Support에 문의하세요.이 솔루션에 대한 AWS Support 사례를 개설하기 위한 지침을 제공합니다.

알려진 문제 해결

문제: 기존 VPC를 사용 중이며 테스트가 실패하고 상태가 실패인 경우 다음 오류 메시지가 표시됩니다.

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

- 해결 방법:

서브넷이 지정된 VPC에 존재하고 인터넷 게이트웨이 또는 NAT 게이트웨이가 있는 인터넷 경로가 있는지 확인합니다. AWS Fargate는 테스트를 성공적으로 실행하기 위해 퍼블릭 리포지토리에서 컨테이너 이미지를 가져올 수 있는 액세스 권한이 필요합니다.

문제: 테스트가 실행되는 데 너무 오래 걸리거나 무기한으로 중단됨

- 해결 방법:

테스트를 취소하고 AWS Fargate를 확인하여 모든 작업이 중지되었는지 확인합니다. 중지하지 않은 경우 모든 Fargate 작업을 수동으로 중지합니다. 계정의 온디맨드 Fargate 작업 제한을 확인하여 원하는 수의 작업을 시작할 수 있는지 확인합니다. 또한 Lambda 작업 실행기 함수에 대한 CloudWatch 로그를 확인하여 Fargate 작업을 시작할 때 실패에 대한 더 많은 인사이트를 얻을 수 있습니다. 실행 중인 Fargate 컨테이너에서 발생하는 상황에 대한 자세한 내용은 CloudWatch ECS 로그를 확인하세요.

문제: 테스트가 시작되었지만 완료되지 않거나 ECS 작업의 상태를 알 수 없음

- 해결 방법:

솔루션이 배포된 계정에서 기존 VPC를 제공하는 옵션을 선택한 경우 ECS 작업에서 사용 중인 VPC에 테스트 입력에 제공된 작업 수를 시작하기에 충분한 여유 IP 주소가 있는지 확인합니다. ECS 작업 정의는 인터넷 게이트웨이 또는 인터넷 경로가 필요한 ECR 이미지를 사용하므로 ECS 서비스가 aws-solutions/distributed-load-testing-on-aws-load-tester에서 솔루션 ECR 이미지를 다운로드하여 작

업을 프로비저닝할 수 있습니다. VPC의 모든 서브넷이 프라이빗이므로 인터넷 경로를 제공할 수 없는 경우 ECR 풀스루 캐시를 사용하여 계정에서 ECR 이미지를 호스팅할 수 있습니다. 새 ECR 이미지 URI로 작업 정의를 업데이트하고 새 개정을 생성합니다. 작업 정의가 업데이트되면 새 개정을 사용하려면 DynamoDB 테이블의 솔루션 구성을 업데이트해야 합니다. DynamoDB 테이블 이름은 키 ScenariosTable 아래의 CloudFormation 스택 출력 탭에서 찾을 수 있습니다. 키 testId 및 값 region-[SOLUTION-DEPLOYED-REGION]을 사용하여 항목의 속성 taskDefinition을 업데이트합니다.

문제: 테스트는 프라이빗이거나 인터넷 게이트웨이를 통해 사용할 수 없는 엔드포인트를 사용해야 합니다.

- 해결 방법:

인터넷 게이트웨이를 통해 액세스할 수 없는 프라이빗 API 엔드포인트를 테스트할 때는 다음 접근 방식을 고려하세요.

1. 네트워크 구성: ECS 작업에서 사용하는 서브넷 라우팅 테이블이 테스트 중인 프라이빗 엔드포인트의 IP 주소 범위에 대한 경로로 업데이트되었는지 확인합니다. 이렇게 하면 테스트 트래픽이 VPC 내의 프라이빗 엔드포인트에 도달할 수 있습니다.
2. DNS 확인: 사용자 지정 도메인의 경우 프라이빗 엔드포인트의 도메인 이름을 확인하도록 VPC의 DNS 설정을 구성합니다. 자세한 지침은 VPC DNS 설명서를 참조하세요.
3. VPC 엔드포인트: AWS 서비스를 테스트하는 경우 VPC 엔드포인트(AWS PrivateLink)를 사용하여 프라이빗 연결을 설정하는 것이 좋습니다. 예를 들어 프라이빗 API Gateway를 테스트하려면 API Gateway에 대한 VPC 엔드포인트를 생성할 수 있습니다. 프라이빗 API Gateway 설명서를 참조하세요.
4. VPC 피어링: 프라이빗 엔드포인트가 다른 VPC에 있는 경우 솔루션이 배포되는 VPC와 프라이빗 엔드포인트가 포함된 VPC 간에 VPC 피어링을 설정합니다. 두 VPCs 구성합니다. VPC 피어링 설명서를 참조하세요.
5. Transit Gateway: 여러 VPCs 포함된 보다 복잡한 네트워킹 시나리오의 경우 AWS Transit Gateway를 사용하여 솔루션의 VPC와 프라이빗 엔드포인트가 포함된 VPC 간에 트래픽을 라우팅하는 것이 좋습니다. Transit Gateway 설명서를 참조하세요.
6. 보안 그룹: ECS 태스크와 연결된 보안 그룹이 프라이빗 엔드포인트로의 아웃바운드 트래픽을 허용하고 프라이빗 엔드포인트의 보안 그룹이 ECS 태스크의 인바운드 트래픽을 허용하는지 확인합니다.

내부 Application Load Balancer 또는 EC2 인스턴스를 테스트하려면 VPC CIDR 범위가 겹치지 않고 필요한 경로가 라우팅 테이블에 구성되어 있는지 확인합니다.

문제: 테스트가 완료되었지만 UI에서 결과를 사용할 수 없음

- 해결 방법:

테스트가 완료되었지만 UI에서 결과를 사용할 수 없는 경우 테스트를 실행한 ECS 작업의 S3 버킷에서 결과 파일을 계속 사용할 수 있어야 합니다. 이는 솔루션에서 알려진 제한 사항입니다. 현재 아키텍처에서 솔루션은 결과 구문 분석 Lambda 함수를 사용하여 여러 ECS 작업의 결과를 요약한 다음 DynamoDB 테이블에 항목으로 저장됩니다. DynamoDB 테이블의 최대 항목 크기는 400KB입니다. 이 제한은 테스트 스크립트의 복잡성, 동시성 및 사용 중인 작업 수에 따라 결정됩니다. 오류는 테스트가 실패하고 있음을 의미하지 않으며, 결과를 요약하고 CRUD 작업을 위해 DynamoDB 테이블에 저장하는 프로세스가 실패했음을 나타냅니다. 결과는 테스트 시나리오의 S3 버킷에서 계속 사용할 수 있습니다.

AWS Support에 문의

AWS Business Support+, AWS Enterprise Support 또는 통합 운영이 있는 경우 AWS Support Center를 사용하여이 솔루션에 대한 전문가 지원을 받을 수 있습니다. 이후 단원에서는 그 방법에 대해서 설명합니다.

사례 생성

1. 지원 센터에 로그인합니다.
2. 사례 생성을 선택합니다.

지원 방법

1. 기술 선택
2. 서비스에서 솔루션을 선택합니다.
3. 범주에서 AWS의 분산 로드 테스트를 선택합니다.
4. 심각도에서 사용 사례에 가장 적합한 옵션을 선택합니다.
5. 서비스, 카테고리 및 심각도를 입력하면 인터페이스가 일반적인 문제 해결 질문에 대한 링크를 제공합니다. 이러한 링크로 질문을 해결할 수 없는 경우 다음 단계: 추가 정보를 선택합니다.

추가 정보

1. 제목에 질문 또는 문제를 요약하는 텍스트를 입력합니다.
2. 설명에서 AWS vX.Y.Z의 Distributed Load Testing 예제와 같이이 제품의 이름과 사용 중인 버전을 포함하여 문제를 자세히 설명합니다.
3. 파일 연결을 선택합니다.
4. AWS Support에서 요청을 처리하는 데 필요한 정보를 첨부합니다.

사례를 더 빠르게 해결할 수 있도록 지원

1. 필요한 정보를 입력합니다.
2. 다음 단계: 지금 해결하거나 AWS에 문의하기를 선택합니다.

지금 해결 또는 문의

1. 지금 해결 솔루션을 검토합니다.
2. 이러한 솔루션의 문제를 해결할 수 없는 경우 문의를 선택하고 요청된 정보를 입력한 다음 제출을 선택합니다.

솔루션 제거

AWS 관리 콘솔에서 또는 AWS 명령줄 인터페이스를 사용하여 AWS에서 분산 로드 테스트를 제거할 수 있습니다. 이 솔루션에서 생성한 콘솔, 시나리오 및 Amazon Simple Storage Service(Amazon S3) 버킷 로깅을 수동으로 삭제해야 합니다. AWS 솔루션 구현은 보존할 데이터가 있는 경우 자동으로 삭제되지 않습니다.

> **Note**
>
> 리전 스택을 배포한 경우 기본 스택을 삭제하기 전에 해당 리전에서 스택을 삭제해야 합니다.

AWS 관리 콘솔 사용

AWS CloudFormation

1. AWS CloudFormation 콘솔에 로그인합니다.
2. 스택 페이지에서 이 솔루션의 설치 스택을 선택합니다.
3. 삭제를 선택합니다.

AWS Launch Wizard

1. AWS Launch Wizard 콘솔에 로그인합니다.
2. Launch Wizard 배포 페이지에서이 솔루션의 배포를 선택합니다.
3. 작업을 선택한 다음 삭제를 선택합니다.
4. 삭제를 확인합니다.

AWS Command Line Interface 사용

사용자 환경에서 AWS Command Line Interface(AWS CLI)를 사용할 수 있는지 확인합니다. 설치 지침은 AWS CLI 사용 설명서의 AWS Command Line Interface란 무엇입니까?를 참조하세요. AWS CLI를 사용할 수 있는지 확인한 후 다음 명령을 실행합니다.

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

Amazon S3 버킷 삭제

이 솔루션은 실수로 데이터가 손실되지 않도록 AWS CloudFormation 스택을 삭제하기로 결정한 경우 솔루션에서 생성한 Amazon S3 버킷(옵트인 리전에 배포용)을 유지하도록 구성됩니다. 솔루션을 제거한 후 데이터를 보존할 필요가 없는 경우이 S3 버킷을 수동으로 삭제할 수 있습니다. 다음 단계에 따라 Amazon S3 버킷을 삭제합니다.

1. Amazon S3 콘솔에 로그인합니다.
2. 왼쪽 탐색 창에서 버킷을 선택합니다.
3. 이름으로 버킷 찾기 필드에이 솔루션 스택의 이름을 입력합니다.
4. 솔루션의 S3 버킷 중 하나를 선택하고 비우기를 선택합니다.
5. 확인 필드에 영구적으로 삭제를 입력하고 비우기를 선택합니다.
6. 방금 비운 S3 버킷을 선택하고 삭제를 선택합니다.
7. 확인 필드에 S3 버킷 이름을 입력하고 버킷 삭제를 선택합니다.

모든 S3 버킷을 삭제할 때까지 4~7단계를 반복합니다.

AWS CLI를 사용하여 S3 버킷을 삭제하려면 다음 명령을 실행합니다.

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

솔루션 사용

이 섹션에서는 첫 번째 테스트 시나리오 생성부터 자세한 결과 분석에 이르기까지 AWS에서 분산 로드 테스트 솔루션을 사용하는 방법에 대한 포괄적인 가이드를 제공합니다. 워크플로에는 테스트 시나리오 생성, 테스트 실행, 테스트 결과 탐색이 포함됩니다.

테스트 시나리오 생성

테스트 시나리오 생성에는 일반 설정 구성, 시나리오 정의, 트래픽 패턴 구성, 구성 검토의 네 가지 주요 단계가 포함됩니다.

1단계: 일반 설정

테스트 이름, 설명 및 일반 구성 옵션을 포함하여 로드 테스트의 기본 파라미터를 구성합니다.

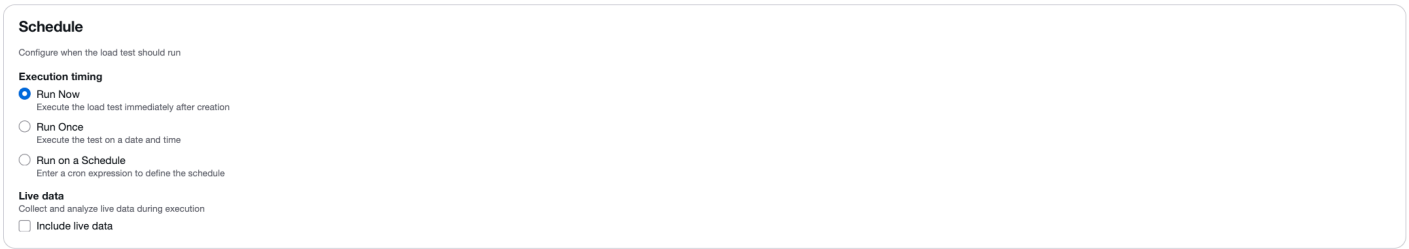
테스트 식별

- 테스트 이름(필수) - 테스트 시나리오를 설명하는 이름입니다.
- 테스트 설명(필수) - 테스트 목적 및 구성에 대한 추가 세부 정보
- 태그(선택 사항) - 최대 5개의 태그를 추가하여 테스트 시나리오를 분류하고 구성합니다.

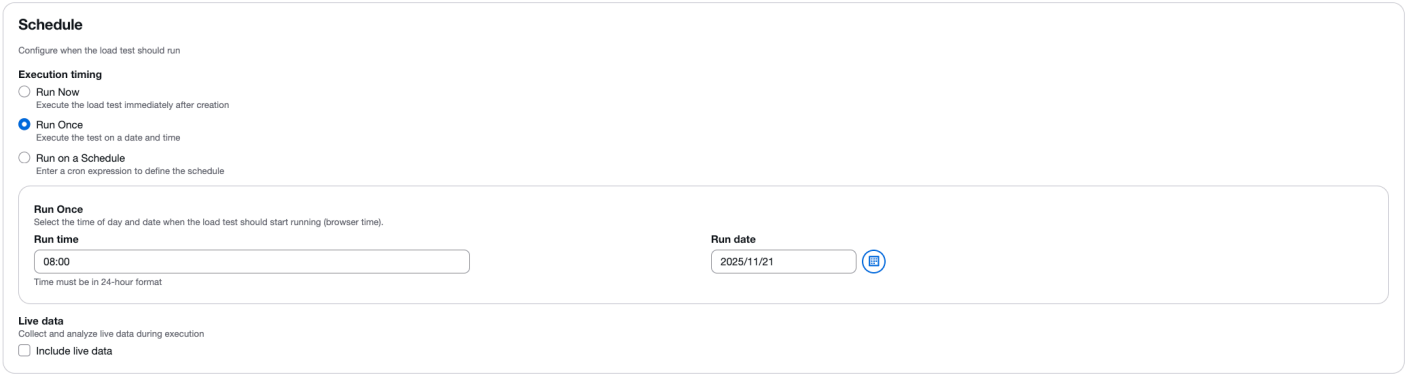
예약 옵션

테스트를 실행해야 하는 시기를 구성합니다.

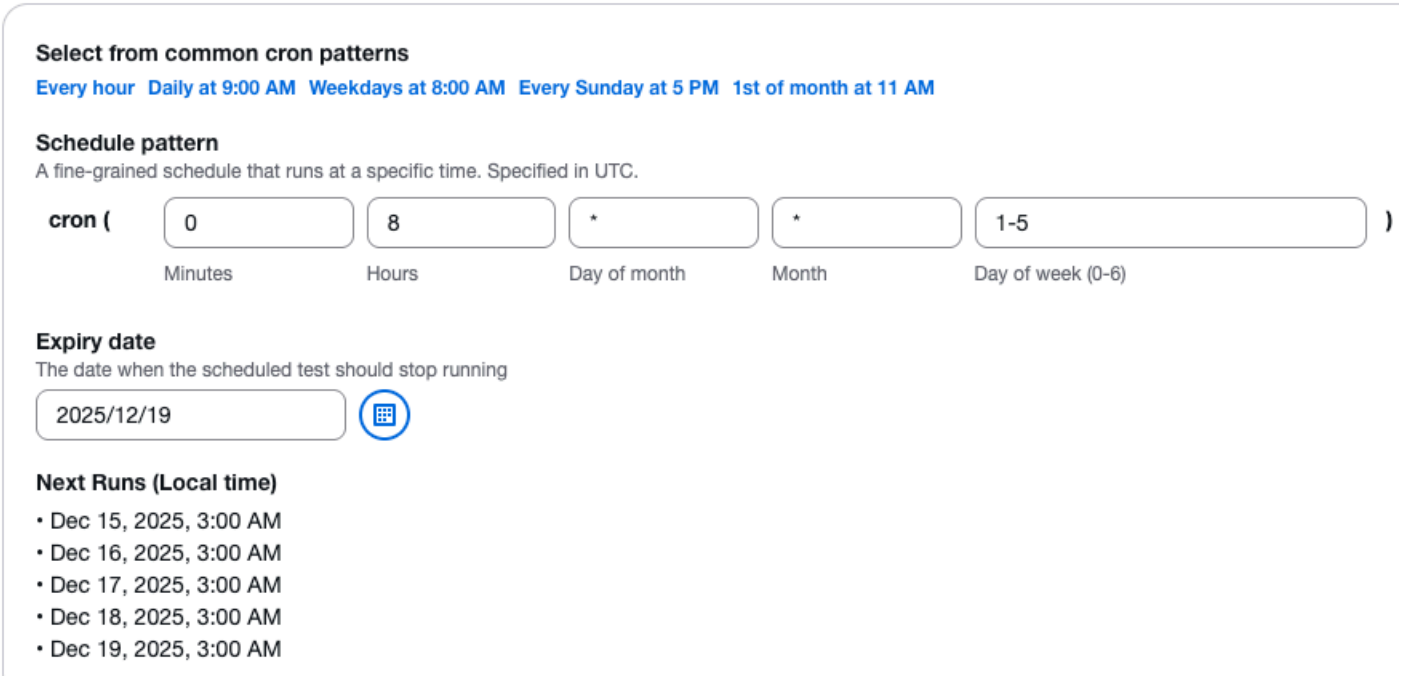
- 지금 실행 - 생성 직후 테스트를 실행합니다.



- 한 번 실행 - 특정 날짜 및 시간에 실행되도록 테스트를 예약합니다.



- 일정에 따라 실행 - cron 기반 예약을 사용하여 정기적으로 테스트를 자동으로 실행합니다. 일반적인 패턴(매시간, 매일, 매주) 중에서 선택하거나 사용자 지정 cron 표현식을 정의할 수 있습니다.



워크플로 예약

테스트를 예약하면 다음 워크플로가 발생합니다.

- 일정 파라미터는 Amazon API Gateway로 전송됩니다.
- API는 지정된 날짜에 실행되도록 예약된 CloudWatch Events 규칙을 생성하는 Lambda 함수에 파라미터를 전달합니다.
- 일회성 테스트(한 번 실행)의 경우 CloudWatch Events 규칙은 지정된 날짜에 실행되고 `api-services` Lambda 함수는 테스트를 실행합니다.

- 반복 테스트(일정에 따라 실행)의 경우 CloudWatch Events 규칙이 지정된 날짜에 활성화되고 `api-services` Lambda 함수는 지정된 빈도에 따라 즉시 반복적으로 실행되는 새 규칙을 생성합니다.

라이브 데이터

테스트가 실행되는 동안 실시간 지표를 보려면 라이브 데이터 포함 확인란을 선택합니다. 활성화되면 다음을 모니터링할 수 있습니다.

- 평균 응답 시간입니다.
- 가상 사용자 수입니다.
- 성공적인 요청 수입니다.
- 실패한 요청 수입니다.

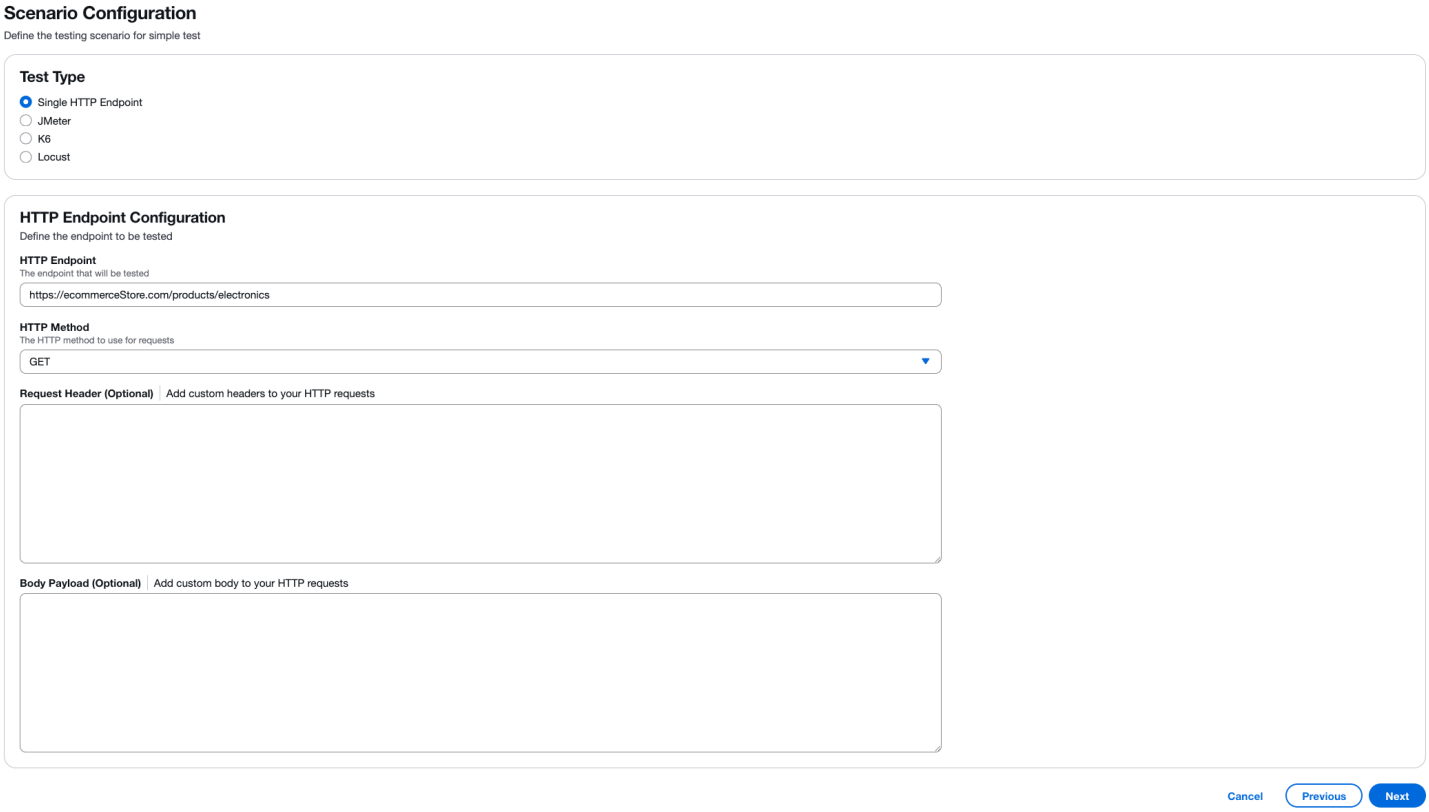
라이브 데이터 기능은 1초 간격으로 집계된 데이터를 사용하여 실시간 차트를 제공합니다. 자세한 내용은 라이브 데이터로 모니터링을 참조하세요.

2단계: 시나리오 구성

특정 테스트 시나리오를 정의하고 원하는 테스트 프레임워크를 선택합니다.

테스트 유형 선택

수행할 로드 테스트 유형을 선택합니다.



- 단일 HTTP 엔드포인트 - 간단한 구성으로 단일 API 엔드포인트 또는 웹 페이지를 테스트합니다.
- JMeter - JMeter 테스트 스크립트(.jmx 파일 또는 .zip 아카이브)를 업로드합니다.
- K6 - K6 테스트 스크립트(.js 파일 또는 .zip 아카이브)를 업로드합니다.
- Locust - Locust 테스트 스크립트(.py 파일 또는 .zip 아카이브)를 업로드합니다.

HTTP 엔드포인트 구성 이미지::images/test-types.png[실행할 테스트 유형 선택] "단일 HTTP 엔드포인트"를 선택하면 다음 설정을 구성합니다.

HTTP 엔드포인트(필수)

테스트하려는 엔드포인트의 전체 URL을 입력합니다. 예를 들어 `https://api.example.com/users`입니다. AWS 인프라에서 엔드포인트에 액세스할 수 있는지 확인합니다.

HTTP 메서드(필수)

요청에 대한 HTTP 메서드를 선택합니다. 기본값은 `GET`입니다. 다른 옵션에는 `POST`, `PUT`, `DELETE`, `PATCH`, 및 `HEAD`가 포함됩니다`OPTIONS`.

요청 헤더(선택 사항)

요청에 사용자 지정 HTTP 헤더를 추가합니다. 일반적인 예는 다음과 같습니다.

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

헤더 추가를 선택하여 여러 헤더를 포함합니다.

본문 페이로드(선택 사항)

POST 또는 PUT 요청에 대한 요청 본문 콘텐츠를 추가합니다. JSON, XML 또는 일반 텍스트 형식을 지원합니다. 예를 들어 `{"userId": 123, "action": "test"}`입니다.

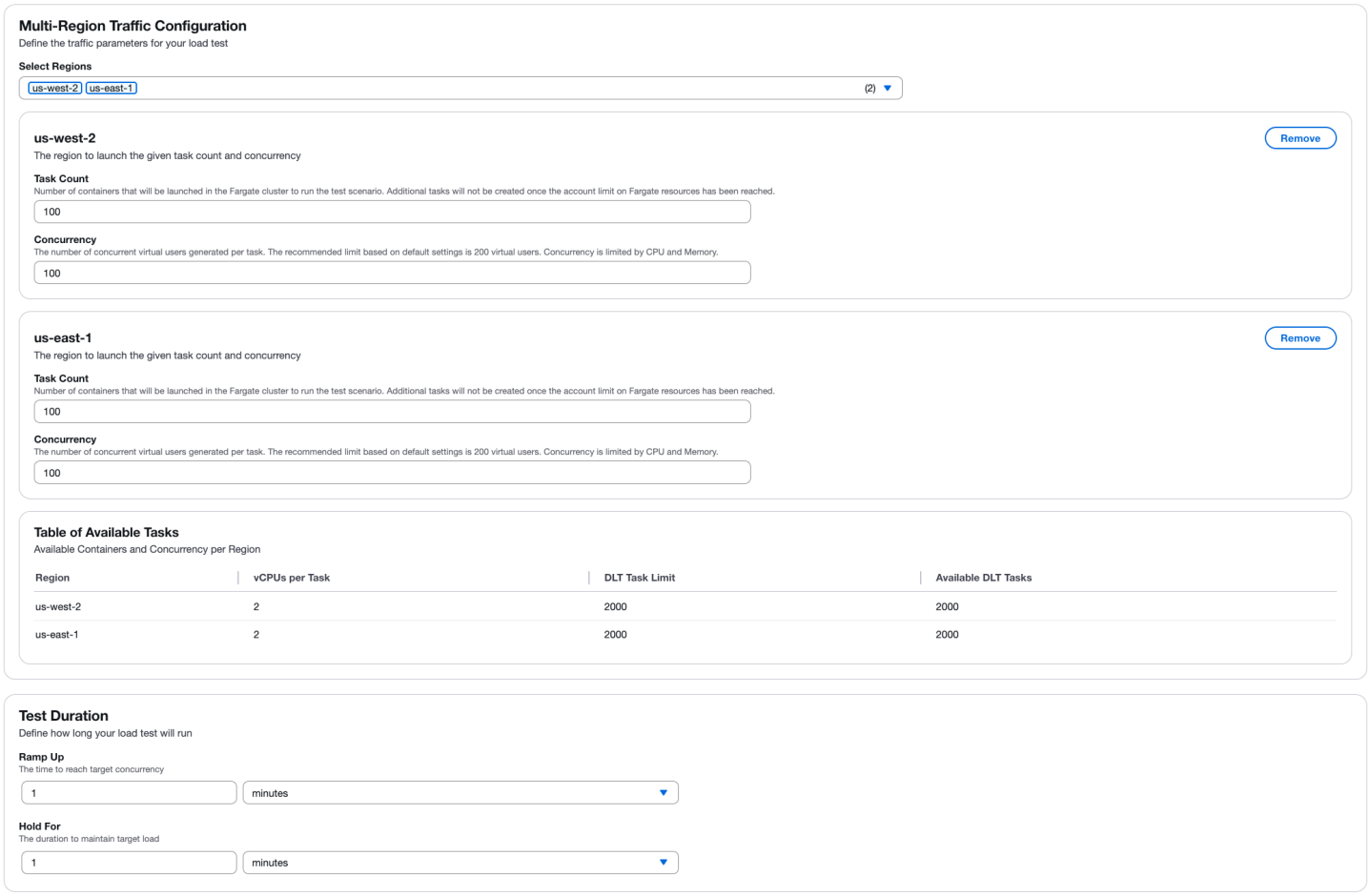
프레임워크 스크립트 테스트

JMeter, K6 또는 Locust를 사용하는 경우 테스트 스크립트 파일 또는 테스트 스크립트와 지원 파일이 포함된.zip 아카이브를 업로드합니다. JMeter의 경우 .zip 아카이브 내의 `/plugins` 폴더에 사용자 지정 플러그인을 포함할 수 있습니다.

> **Important**
>
> 테스트 스크립트(JMeter, K6 또는 Locust)가 동시성(가상 사용자), 트랜잭션 속도(TPS), 증가 시간 및 기타 로드 파라미터를 정의할 수 있지만 솔루션은 테스트 생성 중에 트래픽 셰이프 화면에서 지정한 값으로 이러한 구성을 재정의합니다. 트래픽 셰이프 구성은 테스트 실행의 작업 수, 동시성(작업당 가상 사용자 수), 증가 기간 및 대기 기간을 제어합니다.

3단계: 트래픽 셰이프

다중 리전 지원을 포함하여 테스트 중에 트래픽이 분산되는 방식을 구성합니다.



다중 리전 트래픽 구성

하나 이상의 AWS 리전을 선택하여 로드 테스트를 지리적으로 배포합니다. 선택한 각 리전에 대해 다음을 구성합니다.

작업 수

테스트 시나리오를 위해 Fargate 클러스터에서 시작될 컨테이너(작업) 수입니다. 계정이 "Fargate 리소스에 도달" 한도에 도달하면 추가 작업이 생성되지 않습니다.

동시성

작업당 생성된 동시 가상 사용자 수입니다. 권장 제한은 작업당 vCPUs 2개의 기본 설정을 기반으로 합니다. 동시성은 CPU 및 메모리 리소스에 의해 제한됩니다.

사용자 수 결정

컨테이너가 테스트에 지원할 수 있는 사용자 수는 Amazon CloudWatch에서 사용자 수를 점진적으로 늘리고 성능을 모니터링하여 확인할 수 있습니다. CPU 및 메모리 성능이 제한에 도달하는 것을 관찰하면 컨테이너가 기본 구성(vCPU 2개 및 메모리 4GB)에서 해당 테스트를 지원할 수 있는 최대 사용자 수에 도달한 것입니다.

보정 프로세스

다음 예제를 사용하여 테스트에 대한 동시 사용자 제한을 확인할 수 있습니다.

1. 사용자가 200명 이하인 테스트를 생성합니다.
2. 테스트가 실행되는 동안 CloudWatch 콘솔을 사용하여 CPU와 메모리를 모니터링합니다.
   a. 왼쪽 탐색 창의 Container Insights에서 성능 모니터링을 선택합니다.
   b. 성능 모니터링 페이지의 왼쪽 드롭다운 메뉴에서 ECS 클러스터를 선택합니다.
   c. 오른쪽 드롭다운 메뉴에서 Amazon Elastic Container Service(Amazon ECS) 클러스터를 선택합니다.
3. 모니터링하는 동안 CPU와 메모리를 관찰합니다. CPU가 75%를 초과하지 않거나 메모리가 85%를 초과하지 않는 경우(일회성 피크 무시) 사용자 수가 많은 다른 테스트를 실행할 수 있습니다.

테스트가 리소스 제한을 초과하지 않은 경우 1~3단계를 반복합니다. 선택적으로 컨테이너 리소스를 늘려 동시 사용자 수를 늘릴 수 있습니다. 그러나 이로 인해 비용이 더 많이 듭니다. 자세한 내용은 개발자 안내서를 참조하세요.

> **Note**
>
> 정확한 결과를 얻으려면 동시 사용자 제한을 결정할 때 한 번에 하나의 테스트만 실행합니다. 모든 테스트는 동일한 클러스터를 사용하며 CloudWatch 컨테이너 인사이트는 클러스터를 기반으로 성능 데이터를 집계합니다. 이로 인해 두 테스트가 CloudWatch Container Insights에 동시에 보고되어 단일 테스트에 대한 리소스 사용률 지표가 부정확해집니다.

엔진당 사용자 보정에 대한 자세한 내용은 BlazeMeter 설명서의 타우루스 테스트 보정을 참조하세요.

> **Note**
>
> 솔루션은 각 리전에 사용 가능한 용량 정보를 표시하므로 사용 가능한 한도 내에서 테스트 구성을 계획하는 데 도움이 됩니다.

사용 가능한 작업 테이블

사용 가능한 작업 테이블에는 선택한 각 리전의 리소스 가용성이 표시됩니다.

- 리전 - AWS 리전 이름입니다.
- 작업당 vCPUs - 각 작업에 할당된 가상 CPUs 수(기본값: 2).
- DLT 작업 제한 - 계정의 Fargate 제한(기본값: 2000)에 따라 생성할 수 있는 최대 작업 수입니다.
- 사용 가능한 DLT 작업 - 리전에서 사용할 수 있는 현재 작업 수입니다(기본값: 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

작업당 사용 가능한 작업 또는 vCPUs 수를 늘리려면 개발자 안내서를 참조하세요.

테스트 기간

로드 테스트가 실행되는 기간을 정의합니다.

램프 업

대상 동시성에 도달하는 시간입니다. 이 기간 동안 로드가 0에서 구성된 동시성 수준으로 점진적으로 증가합니다.

에 대해 보류

대상 로드를 유지하는 기간입니다. 테스트는이 기간 동안 전체 동시성으로 계속됩니다.

4단계: 검토 및 생성

테스트 시나리오를 생성하기 전에 모든 구성을 검토합니다. 다음을 확인합니다.

- 일반 설정(이름, 설명, 일정).
- 시나리오 구성(테스트 유형, 엔드포인트 또는 스크립트).
- 트래픽 셰이프(작업, 사용자, 기간, 리전).

검토 후 생성을 선택하여 테스트 시나리오를 저장합니다.

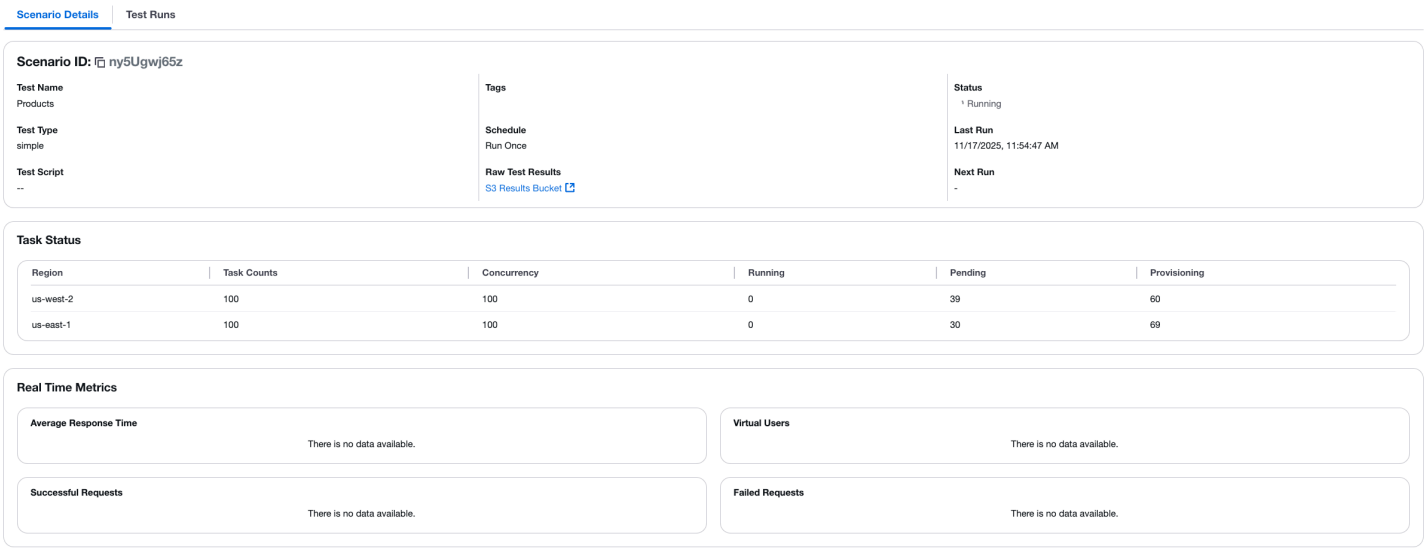
테스트 시나리오 관리

테스트 시나리오를 생성한 후 다음을 수행할 수 있습니다.

- 편집 - 테스트 구성을 수정합니다. 일반적인 사용 사례는 다음과 같습니다.
  - 원하는 트랜잭션 속도를 달성하기 위해 트래픽 셰이프를 구체화합니다.
- 복사 - 기존 테스트 시나리오를 복제하여 변형을 생성합니다. 일반적인 사용 사례는 다음과 같습니다.
  - 엔드포인트 업데이트 또는 헤더/본문 파라미터 추가.
  - 테스트 스크립트 추가 또는 수정.
- 삭제 - 더 이상 필요하지 않은 테스트 시나리오를 제거합니다.

테스트 시나리오 실행

테스트 시나리오를 생성한 후 즉시 실행하거나 향후 특정 시간에 실행되도록 예약할 수 있습니다. 실행 중인 테스트로 이동하면 콘솔에 실시간 작업 상태 및 지표와 함께 시나리오 세부 정보 탭이 표시됩니다.



시나리오 세부 정보 보기

시나리오 세부 정보 탭에는 테스트에 대한 주요 정보가 표시됩니다. 모든 리전에 대한 작업 상태 테이블 실시간 정보입니다.

작업 상태 테이블

작업 상태 테이블에는 각 리전에 대한 실시간 정보가 표시됩니다.

- 리전 - 작업이 실행 중인 AWS 리전
- 작업 수 - 리전에 대해 구성된 총 작업 수
- 동시성 - 작업당 가상 사용자 수
- 실행 중 - 현재 테스트를 실행 중인 작업 수
- 보류 중 - 시작 대기 중인 작업 수
- 프로비저닝 - 프로비저닝되는 작업 수

테스트 실행 워크플로

테스트가 시작되면 다음 워크플로가 발생합니다.

1. 작업 프로비저닝 - 솔루션은 지정된 AWS 리전에 컨테이너(작업)를 프로비저닝합니다. 작업은 "프로비저닝" 열에 표시됩니다.
2. 작업 시작 - 솔루션은 각 리전에서 대상 작업 수에 도달할 때까지 작업을 계속 프로비저닝합니다. 작업은 "프로비저닝"에서 "보류 중"에서 "실행"으로 이동합니다.
3. 트래픽 생성 - 솔루션이 리전의 모든 작업을 프로비저닝한 후 대상 엔드포인트로 트래픽을 전송하기 시작합니다.
4. 테스트 실행 - 구성된 기간(램프업 + 대기 시간) 동안 테스트가 실행됩니다.
5. 결과 구문 분석 - 테스트가 종료되면 백그라운드 구문 분석 작업은 모든 리전의 결과를 집계하고 처리합니다.

테스트 실행 상태

테스트 실행의 상태는 다음과 같습니다.

- 예약됨 - 향후 테스트가 실행되도록 예약됩니다.

- 실행 중 - 테스트가 현재 진행 중입니다.
- 취소됨 - 사용자가 진행 중인 테스트 실행을 취소했습니다.
- 오류 - 테스트 실행에 오류가 발생했습니다.
- 완료 - 테스트 실행이 성공적으로 완료되었으며 결과가 준비되었습니다.

라이브 데이터로 모니터링

테스트 시나리오를 생성할 때 라이브 데이터를 활성화한 경우 테스트가 실행되는 동안 실시간 지표를 볼 수 있습니다. 실시간 지표 섹션에는 테스트가 진행됨에 따라 지속적으로 업데이트되는 그래프 4개가 표시되며 데이터는 1초 간격으로 집계됩니다.

그래프 설명

평균 응답 시간

각 리전에서 처리된 요청에 대한 평균 응답 시간을 초 단위로 표시합니다. Y축은 응답 시간을 초 단위로 표시하고 X축은 시간을 표시합니다. 각 리전은 범례에서 다른 색상으로 표시됩니다.

가상 사용자

각 리전에서 로드를 적극적으로 생성하는 동시 가상 사용자 수를 표시합니다. 그래프에는 가상 사용자가 테스트 중에 어떻게 증가하며 대상 동시성 수준을 유지하는지 표시됩니다.

성공적인 요청

각 리전에 대해 시간 경과에 따른 성공한 요청의 누적 수를 표시합니다. 그래프는 성공적인 요청이 처리되는 속도를 보여줍니다.

실패한 요청

각 리전에 대해 시간 경과에 따른 실패한 요청의 누적 수를 표시합니다. 개수가 낮거나 0이면 정상 테스트 실행을 나타냅니다.

다중 리전 시각화

여러 리전에서 테스트를 실행할 때 각 그래프는 모든 리전에 대한 데이터를 동시에 표시합니다. 각 그래프 하단의 범례는 각 리전을 나타내는 색상을 식별합니다(예: us-west-2 및 us-east-1).

기술 구현

Fargate 태스크의 CloudWatch 로그 그룹에는 테스트 결과를 캡처하는 구독 필터가 포함되어 있습니다. 패턴이 감지되면 Lambda 함수가 데이터를 구조화하여 AWS IoT Core 주제에 게시합니다. 웹 콘솔은 이 주제를 구독하고 지표를 실시간으로 표시합니다.

> **Note**
>
> 라이브 데이터는 임시 데이터이며 테스트가 실행되는 동안에만 사용할 수 있습니다. 웹 콘솔은 최대 5,000개의 데이터 포인트를 유지하며, 그 이후에는 가장 오래된 데이터가 최신 데이터로 대체됩니다. 페이지가 새로 고쳐지면 그래프가 비어 있고 사용 가능한 다음 데이터 포인트에서 시작됩니다. 테스트가 완료되면 솔루션은 결과 데이터를 DynamoDB 및 Amazon S3에 저장합니다. 아직 사용할 수 있는 데이터가 없는 경우 그래프에 “사용 가능한 데이터가 없습니다.”가 표시됩니다.

테스트 취소

웹 콘솔에서 실행 중인 테스트를 취소할 수 있습니다. 테스트를 취소하면 다음 워크플로가 발생합니다.

1. 취소 요청이 `microservices` API로 전송됩니다.
2. `microservices` API는 현재 시작된 모든 작업을 중지하는 `task-canceler` Lambda 함수를 호출합니다.
3. 초기 취소 호출 후에도 `task-runner` Lambda 함수가 계속 실행되는 경우 작업이 잠시 계속 시작될 수 있습니다.

4. `task-runner` Lambda 함수가 완료되면 AWS Step Functions는 `Cancel Test` 단계로 진행하여 `task-canceler` Lambda 함수를 다시 실행하여 나머지 작업을 중지합니다.

> **Note**
>
> 취소된 테스트는 솔루션이 모든 컨테이너를 종료하므로 종료 프로세스를 완료하는 데 시간이 걸립니다. 모든 리소스가 정리되면 테스트 상태가 “취소됨”으로 변경됩니다.

테스트 결과 탐색

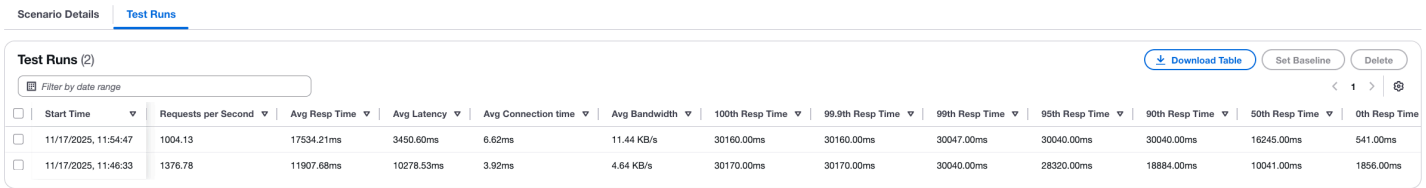
구문 분석 작업이 완료되면 테스트 결과를 분석에 사용할 수 있습니다. 이 솔루션은 로드 시 애플리케이션의 성능을 이해하는 데 도움이 되는 포괄적인 지표와 도구를 제공합니다.

테스트 실행 요약 지표

테스트가 완료되면 솔루션은 다음 지표를 포함하는 요약을 생성합니다.

- 평균 응답 시간 - 테스트에서 생성된 모든 요청에 대한 평균 응답 시간입니다.
- 평균 지연 시간 - 테스트에서 생성된 모든 요청에 대한 초 단위의 평균 지연 시간입니다.
- 평균 연결 시간 - 모든 요청에 대해 호스트에 연결하는 데 걸리는 초 단위의 평균 시간입니다.
- 평균 대역폭 - 테스트에서 생성된 모든 요청의 평균 대역폭입니다.
- 총 수 - 총 요청 수입니다.
- 성공 수 - 성공한 총 요청 수입니다.
- 오류 수 - 총 오류 수입니다.
- 초당 요청 수 - 테스트에서 생성된 모든 요청에 대한 초당 평균 요청 수입니다.
- 백분위수 - 모든 요청에 대한 응답 시간 분포를 보여주는 p50(중앙값), p90, p95 및 p99를 포함한 응답 시간 백분위수입니다.

테스트 실행 테이블



테스트 실행 테이블에는 시나리오에 대한 모든 과거 테스트 실행이 표시됩니다. 다음을 수행할 수 있습니다.

- 각 테스트 실행에 대한 요약 지표를 봅니다.
- 성능 비교를 위한 기준 테스트 실행을 설정합니다.
- 테이블을 CSV 파일로 다운로드합니다.
- 열을 전환하여 보기를 사용자 지정합니다.
- 테스트 실행을 선택하여 자세한 결과를 봅니다.

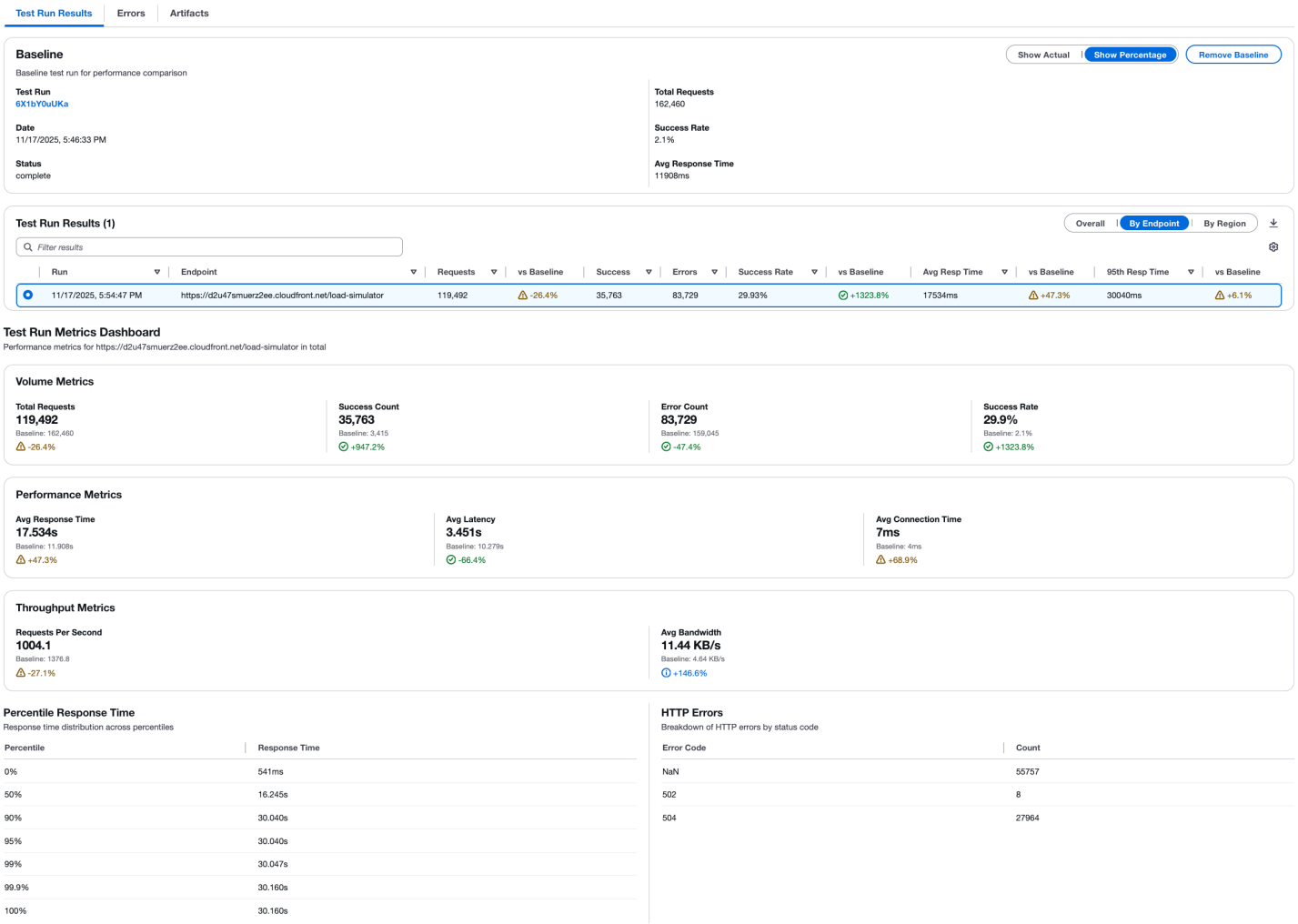
기준 비교

테스트 실행을 기준으로 지정하여 향후 테스트 실행과 비교할 수 있습니다. 기준이 설정된 경우:

- 테스트 실행 표에는 각 지표의 기준과 비교한 백분율 차이 +/-%)가 나와 있습니다.
- 기준 지표는 성능 개선 또는 회귀를 빠르게 식별하는 데 도움이 됩니다.
- 언제든지 기준을 변경하거나 지울 수 있습니다.

세부 테스트 결과

테스트 실행을 선택하면 테스트 실행 결과, 오류 및 아티팩트의 세 가지 탭이 있는 세부 결과 보기가 열립니다.



기준 정보

기준 테스트 실행이 설정된 경우 페이지 상단에 표시됩니다. 실제 표시, 백분율 표시 또는 기준 제거를 선택하여 기준 비교가 표시되는 방식을 제어할 수 있습니다.

테스트 실행 결과 테이블

결과 표는 다음 기능과 함께 자세한 지표를 제공합니다.

차원 보기

차원 버튼을 사용하여 세 보기 간에 전환합니다.

- 전체 - 모든 엔드포인트 및 리전에서 집계된 결과
- 엔드포인트별 - 개별 엔드포인트별로 결과 분류
- 리전별 - AWS 리전별로 결과 분류

작업 버튼

- 실제 표시 - 실제 지표 값 표시
- 백분율 표시 - 기준과의 백분율 차이 표시
- 기준 제거 - 기준 비교 지우기

데이터 내보내기 및 사용자 지정

- 결과 테이블을 CSV 파일로 다운로드
- 열을 전환하여 보기를 사용자 지정합니다.
- 특정 지표에 초점을 맞추기 위한 데이터 필터링 및 정렬
- 특정 지표에 초점을 맞추도록 데이터를 필터링하고 정렬합니다.

오류 탭

오류 탭은 자세한 오류 분석을 제공합니다.

- 유형별로 오류 수를 확인합니다.
- 전체 테스트 또는 엔드포인트별로 집계된 오류를 확인합니다.
- 실패한 요청의 패턴을 식별합니다.
- 특정 엔드포인트 또는 리전 관련 문제를 해결합니다.

아티팩트 탭

아티팩트 탭을 사용하면 테스트 실행 중에 생성된 모든 파일에 액세스할 수 있습니다.

- 개별 아티팩트(로그, 결과 파일)를 봅니다.
- 오프라인 분석을 위한 특정 아티팩트를 다운로드합니다.
- 모든 테스트 실행 아티팩트를 단일 아카이브로 다운로드합니다.

S3 결과 구조

버전 4.0에서는 조직 개선을 위해 S3 결과 구조가 변경되었습니다.

- 새 구조 - `scenario-id/test-run-id/results-files`.

- 레거시 구조 - 버전 4.0 이전에 실행되는 테스트는 시나리오 ID 수준에서 모든 결과 파일을 표시합니다.

> **Note**
>
> 테스트 결과가 콘솔에 표시됩니다. `Results` 폴더 아래의 Amazon S3 버킷에서 직접 원시 테스트 결과에 액세스할 수도 있습니다. Taurus 테스트 결과에 대한 자세한 내용은 Taurus 사용 설명서의 테스트 보고서 생성을 참조하세요.

MCP 서버 통합

솔루션 배포 중에 선택적 MCP 서버 구성 요소를 배포한 경우 Distributed Load Testing 솔루션을 모델 컨텍스트 프로토콜을 지원하는 AI 개발 도구와 통합할 수 있습니다. MCP 서버는 AI 어시스턴트를 통해 로드 테스트를 검색, 관리 및 분석할 수 있는 프로그래밍 방식의 액세스를 제공합니다.

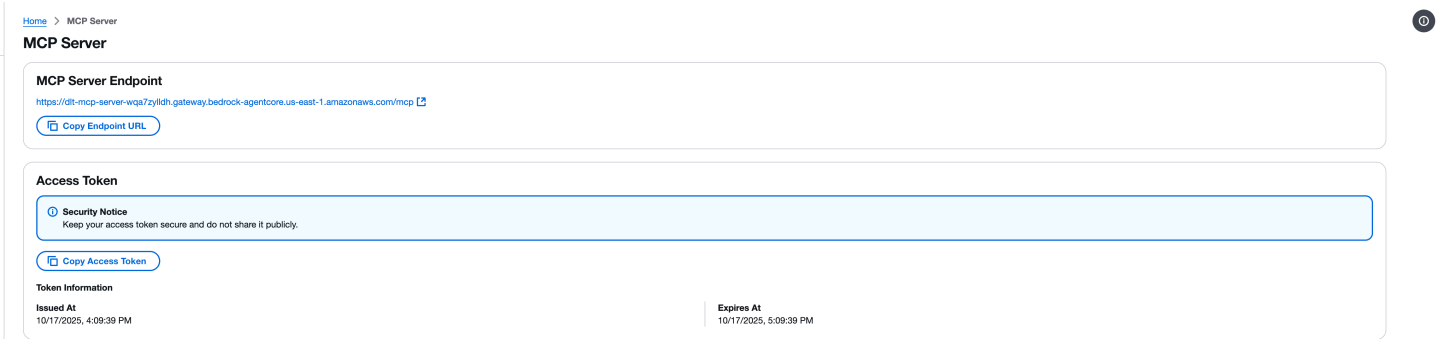
고객은 선택한 클라이언트(Amazon Q, Claude 등)를 사용하여 DLT MCP 서버에 연결할 수 있으며, 각 클라이언트에는 약간 다른 구성 지침이 있습니다. 이 섹션에서는 MCP Inspector, Amazon Q CLI, Cline 및 Amazon Q Suite에 대한 설정 지침을 제공합니다.

1단계: MCP 엔드포인트 및 액세스 토큰 가져오기

MCP 클라이언트를 구성하기 전에 DLT 웹 콘솔에서 MCP 서버 엔드포인트와 액세스 토큰을 검색해야 합니다.

1. 분산 로드 테스트 웹 콘솔에서 MCP 서버 페이지로 이동합니다.
2. MCP 서버 엔드포인트 섹션을 찾습니다.
3. 엔드포인트 URL 복사 버튼을 사용하여 엔드포인트 URL을 복사합니다. 엔드포인트 URL은 형식을 따릅니다. `https://{gateway-id}.gateway.bedrock-agentcore.{region}.amazonaws.com/mcp`
4. 액세스 토큰 섹션을 찾습니다.
5. 액세스 토큰 복사 버튼을 사용하여 액세스 토큰을 복사합니다.

> **Important**
>
> 액세스 토큰을 안전하게 유지하고 공개적으로 공유하지 마세요. 토큰은 MCP 인터페이스를 통해 분산 로드 테스트 솔루션에 대한 읽기 전용 액세스를 제공합니다.



2단계: MCP Inspector로 테스트

모델 컨텍스트 프로토콜은 MCP 서버에 직접 연결하고 도구를 호출하는 도구인 MCP Inspector를 제공합니다. 이를 통해 AI 클라이언트를 구성하기 전에 MCP 서버 연결을 테스트하기 위한 편리한 UI 및 샘플 네트워크 요청이 제공됩니다.

> **Note**
>
> MCP Inspector에는 버전 0.17 이상이 필요합니다. 모든 요청은 JSON RPC로 직접 수행할 수도 있지만 MCP Inspector는 보다 사용자 친화적인 인터페이스를 제공합니다.

MCP Inspector 설치 및 시작

1. 필요한 경우 npm을 설치합니다.
2. 다음 명령을 실행하여 MCP Inspector를 시작합니다.

```
npx @modelcontextprotocol/inspector
```

연결 구성

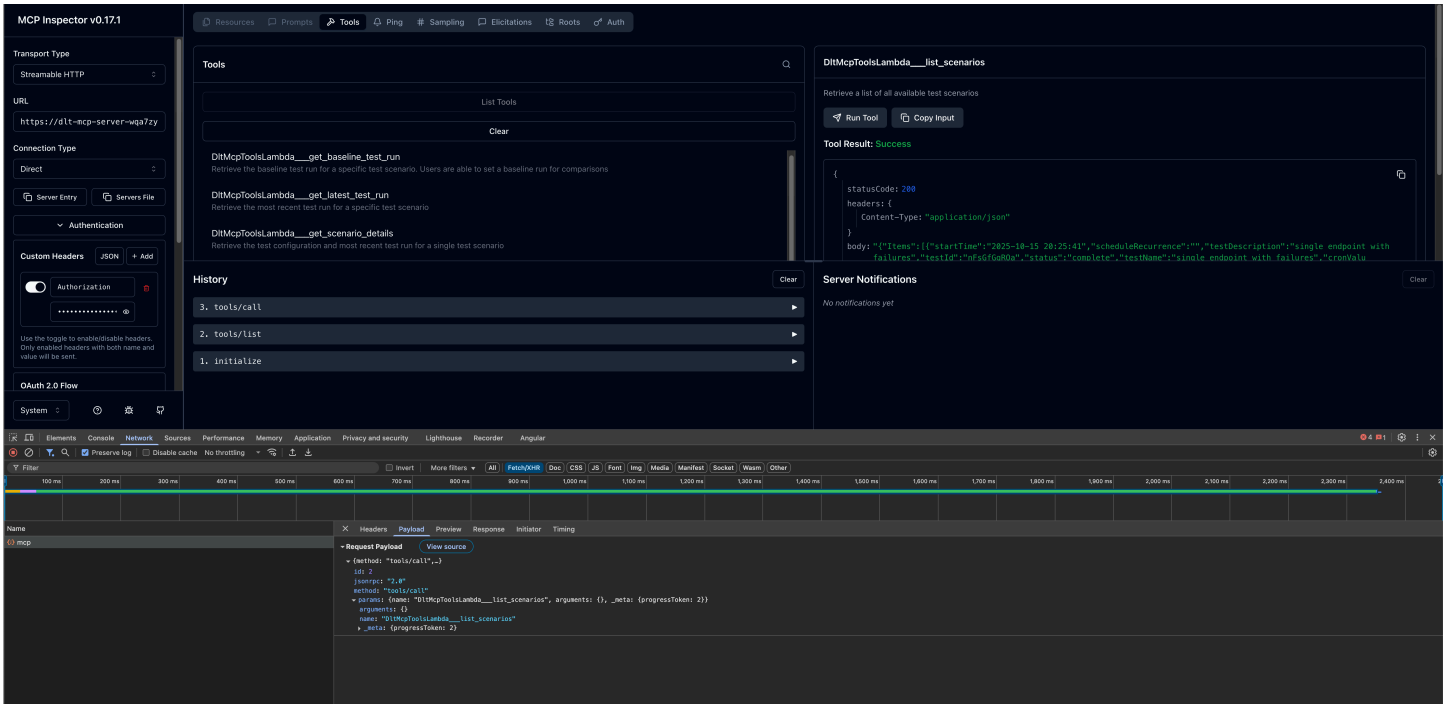
1. MCP Inspector 인터페이스에서 MCP 서버 엔드포인트 URL을 입력합니다.
2. 액세스 토큰과 함께 권한 부여 헤더를 추가합니다.

3. 연결을 클릭하여 연결을 설정합니다.

도구 호출

연결되면 사용 가능한 MCP 도구를 테스트할 수 있습니다.

1. 왼쪽 패널에서 사용 가능한 도구 목록을 찾습니다.
2. 도구를 선택합니다(예: `list_scenarios`).
3. 필요한 파라미터를 제공합니다.
4. 호출을 클릭하여 도구를 실행하고 응답을 봅니다.



3단계: AI 개발 클라이언트 구성

MCP Inspector와의 MCP 서버 연결을 확인한 후 원하는 AI 개발 클라이언트를 구성할 수 있습니다.

Amazon Q CLI

Amazon Q CLI는 MCP 서버 통합을 통해 AI 지원 개발에 대한 명령줄 액세스를 제공합니다.

구성 단계

1. `mcp.json` 구성 파일을 편집합니다. 구성 파일 위치에 대한 자세한 내용은 Amazon Q Developer 사용 설명서의 원격 MCP 서버 구성을 참조하세요.
2. DLT MCP 서버 구성을 추가합니다.

```
{
  "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"
      }
```

```
    }
  }
}
```

구성 확인

1. 터미널에서 q를 입력하여 Amazon Q CLI를 시작합니다.
2. 사용 가능한 모든 MCP 서버를 보려면 /mcp를 입력합니다.
3. dlt-mcp 및 기타 구성된 MCP 서버에서 제공하는 사용 가능한 도구를 보려면 /tools를 입력합니다.
4. 가 dlt-mcp 성공적으로 초기화되는지 확인합니다.

클라인

Cline은 MCP 서버 통합을 지원하는 AI 코딩 도우미입니다.

구성 단계

1. 클라인에서 MCP 서버 관리 > 구성 > MCP 서버 구성으로 이동합니다.
2. cline_mcp_settings.json 파일을 다음과 같이 업데이트 합니다.

```
{
  "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. 구성 파일을 저장합니다.
4. Cline을 다시 시작하여 변경 사항을 적용합니다.

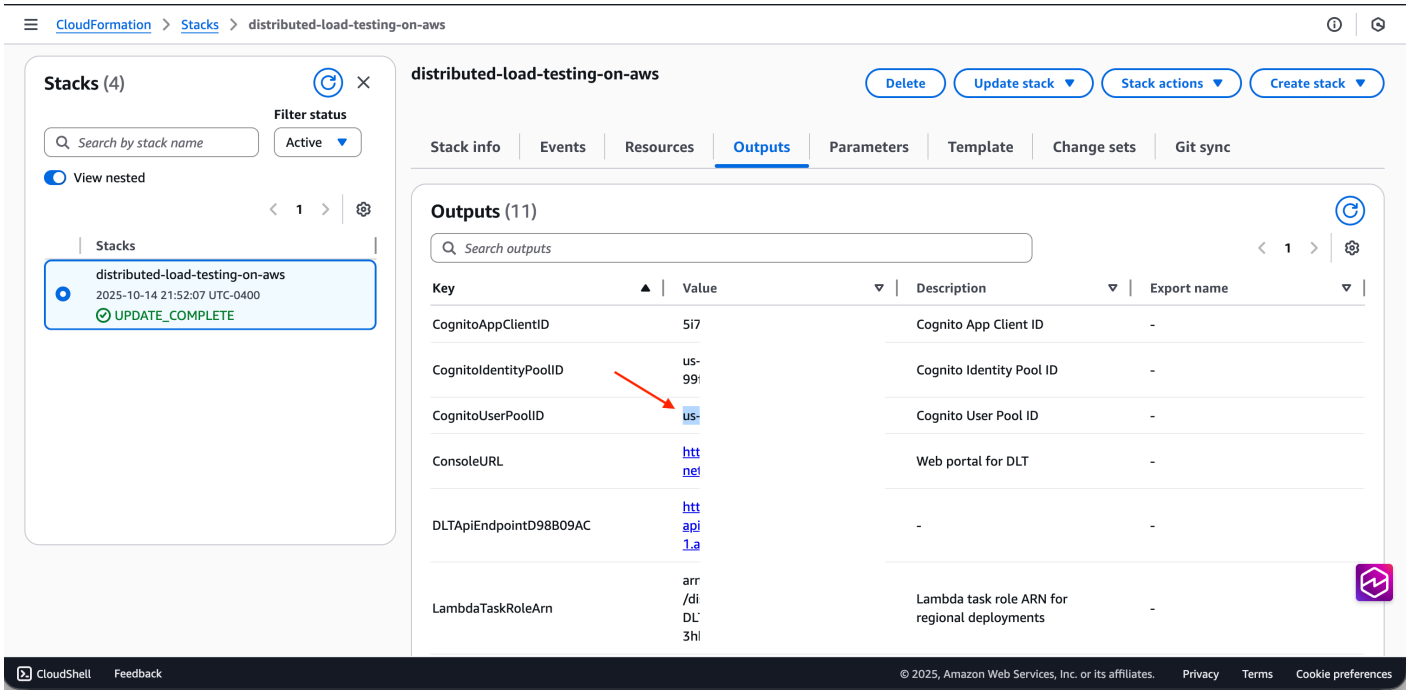
Amazon Q Suite

Amazon Q Suite는 MCP 서버 작업을 지원하는 포괄적인 AI 어시스턴트 플랫폼을 제공합니다.

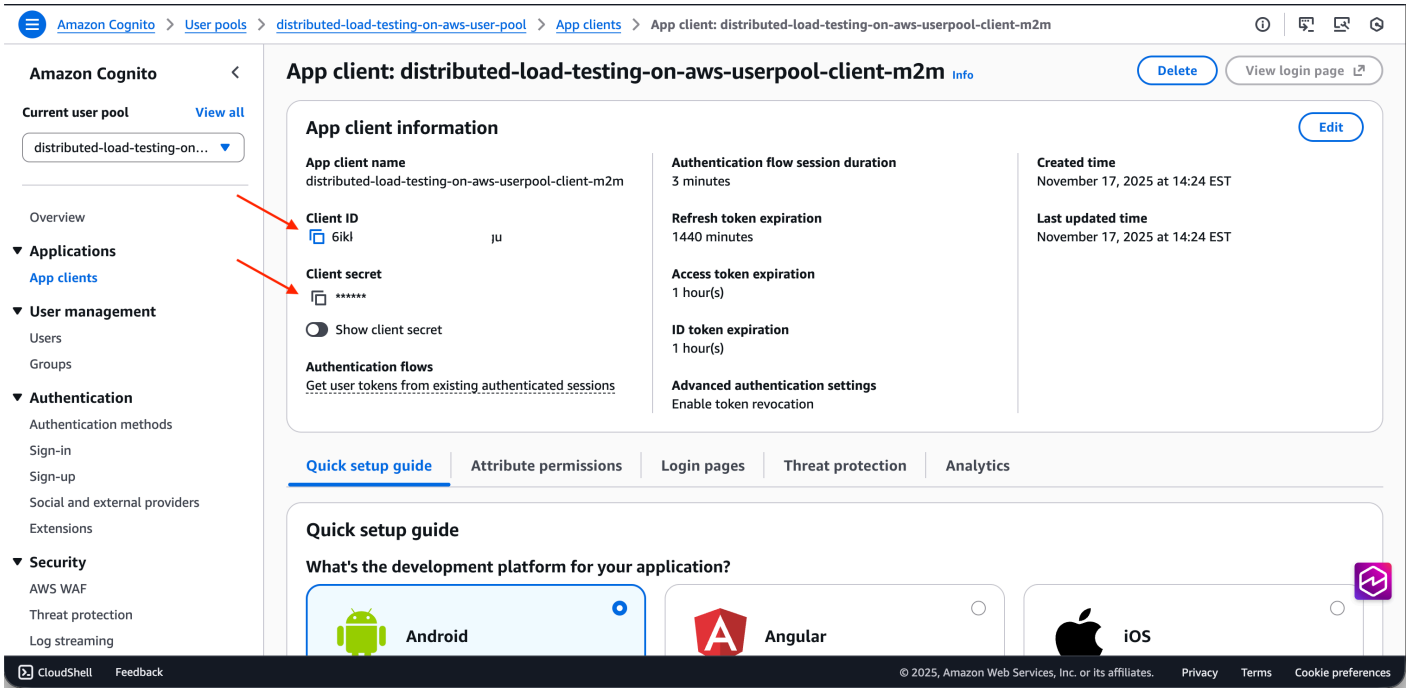
사전 조건

Amazon Q Suite에서 MCP 서버를 구성하기 전에 DLT 배포의 Cognito 사용자 풀에서 OAuth 자격 증명을 검색해야 합니다.

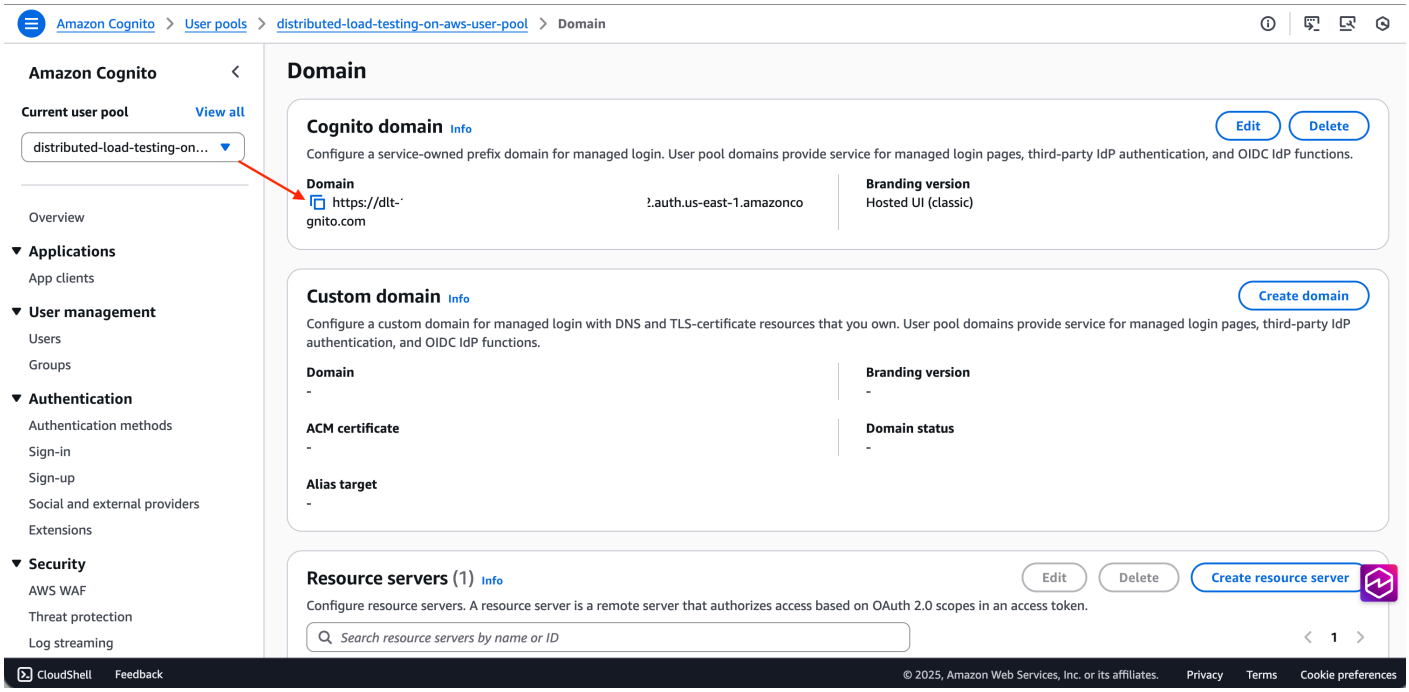
1. AWS CloudFormation 콘솔로 이동합니다.
2. 분산 로드 테스트 스택을 선택합니다.
3. 출력 탭에서 DLT 배포와 연결된 Cognito 사용자 풀 ID를 찾아 복사합니다.



4. Amazon Cognito 콘솔로 이동합니다.
5. CloudFormation 출력에서 사용자 풀 ID를 사용하여 사용자 풀을 선택합니다.
6. 왼쪽 탐색 창에서 앱 통합 > 앱 클라이언트를 선택합니다.



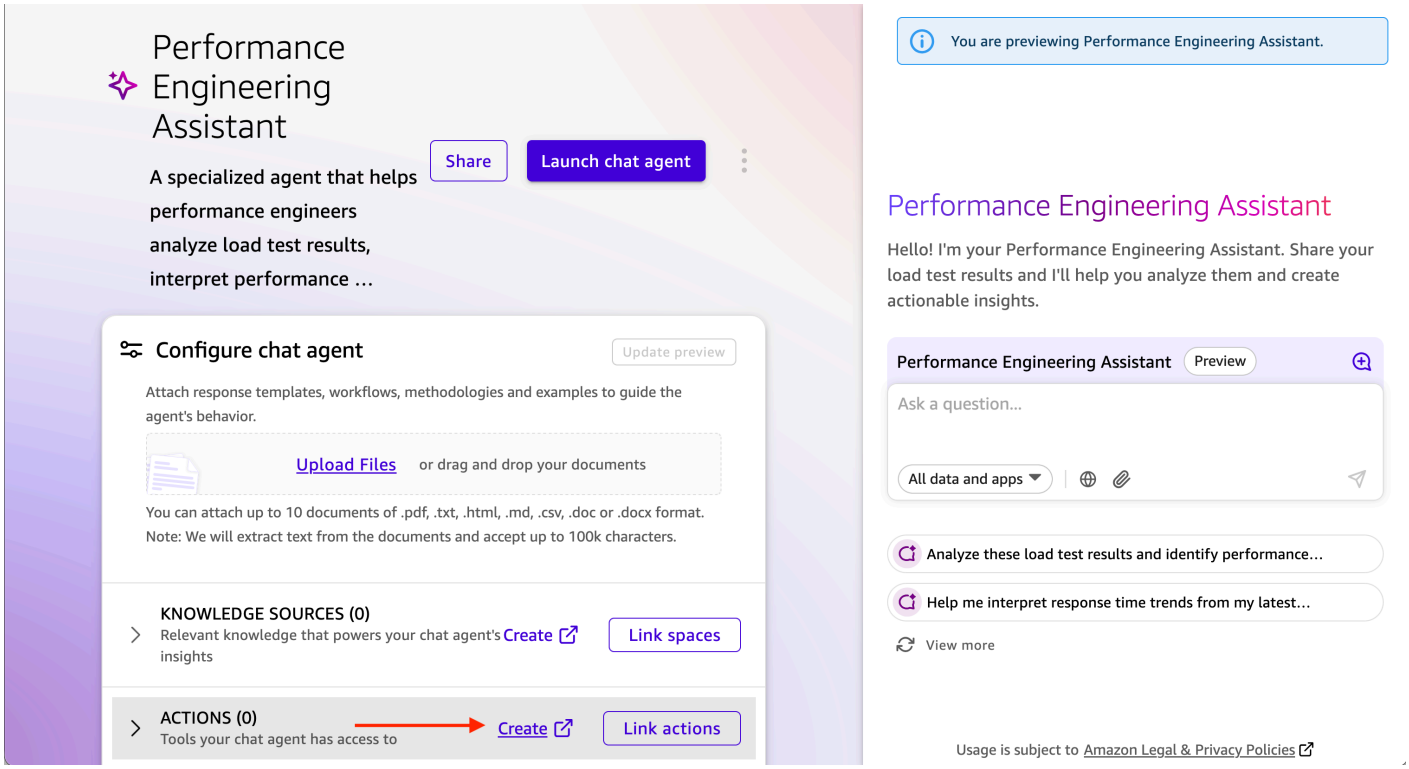
7. 이름이 m2m (machine-to-machine)로 끝나는 앱 클라이언트를 찾습니다.
8. 클라이언트 ID와 클라이언트 보안 암호를 복사합니다.
9. 도메인 탭에서 사용자 풀 도메인을 가져옵니다.

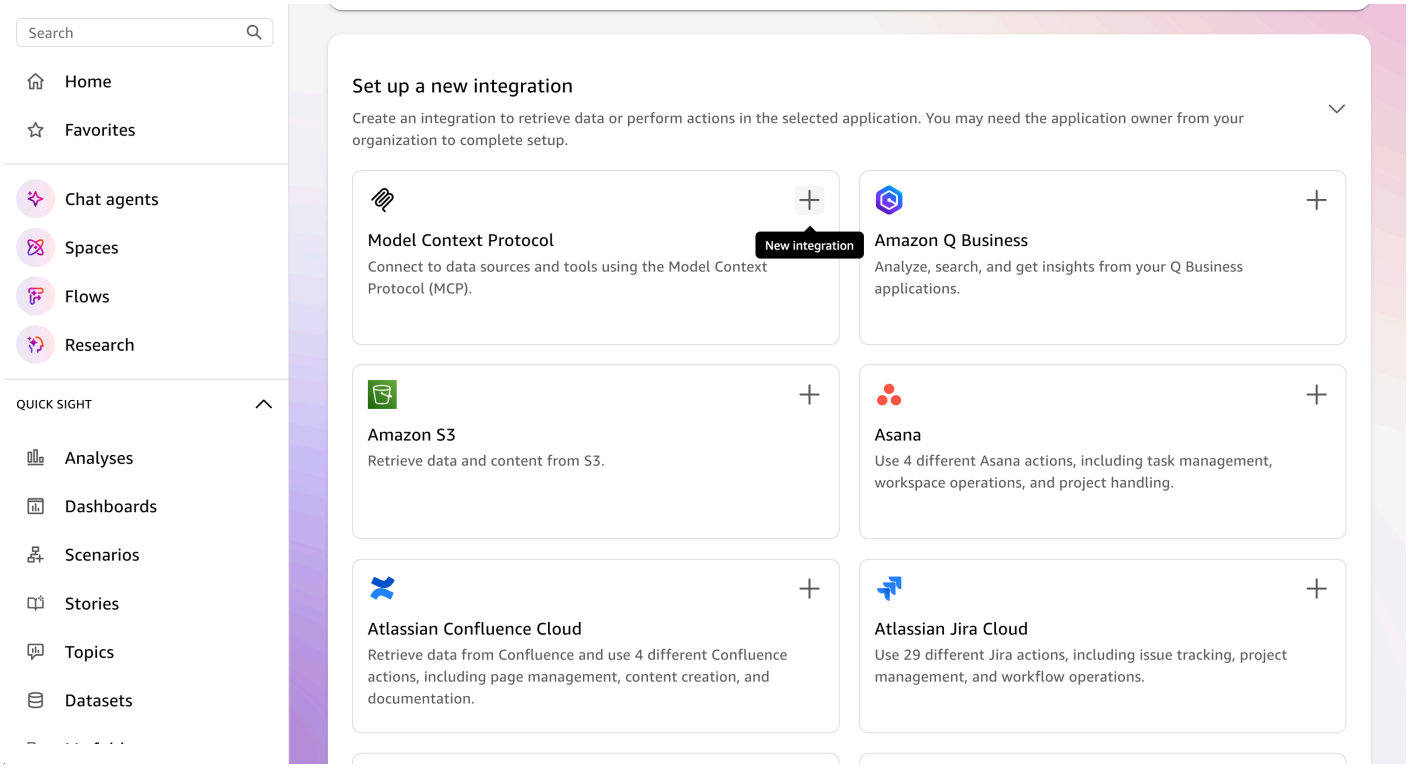


10도메인 /oauth2/token 끝에를 추가하여 토큰 엔드포인트 URL을 구성합니다.

구성 단계

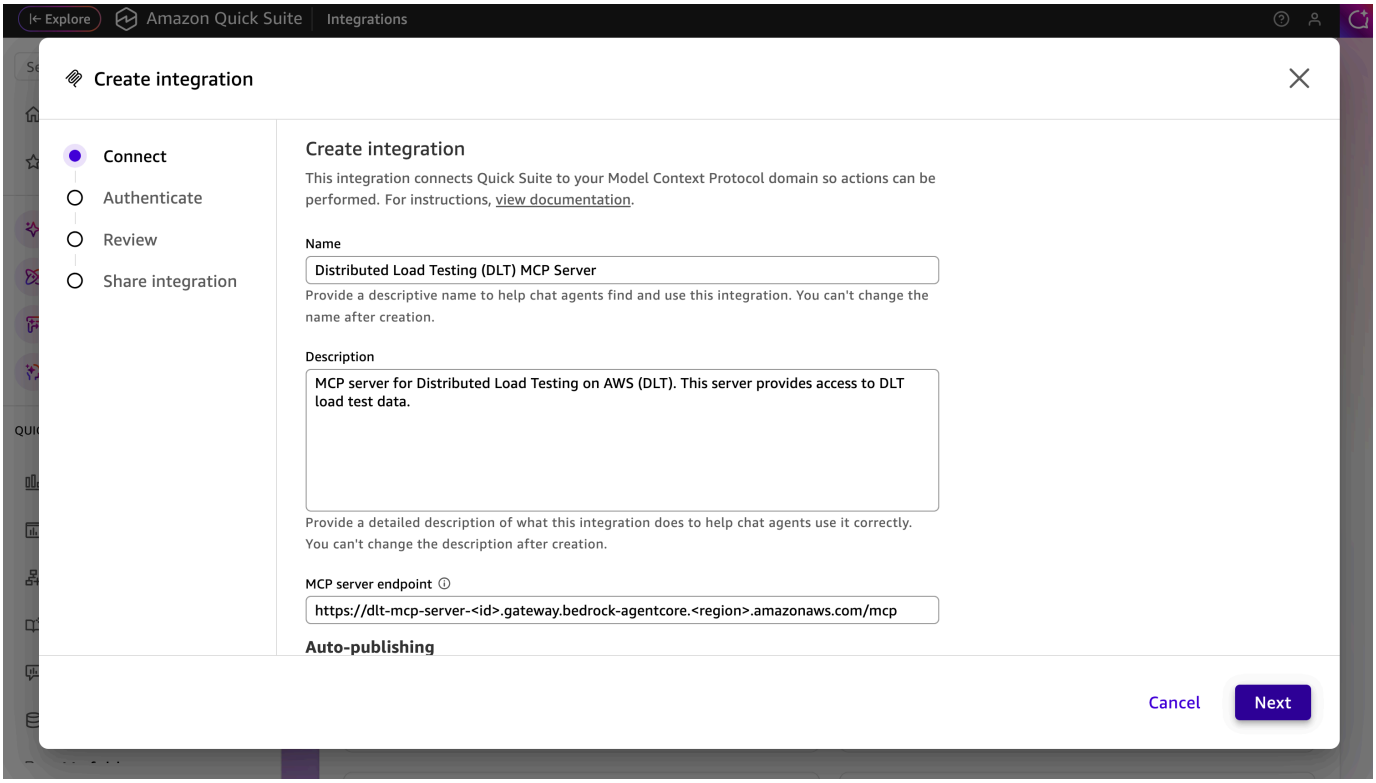
1. Amazon Q Suite에서 새 에이전트를 생성하거나 기존 에이전트를 선택합니다.
2. DLT MCP 서버와 상호 작용하는 방법을 설명하는 에이전트 프롬프트를 추가합니다.
3. 새 작업을 추가하고 MCP 서버 작업을 선택합니다.



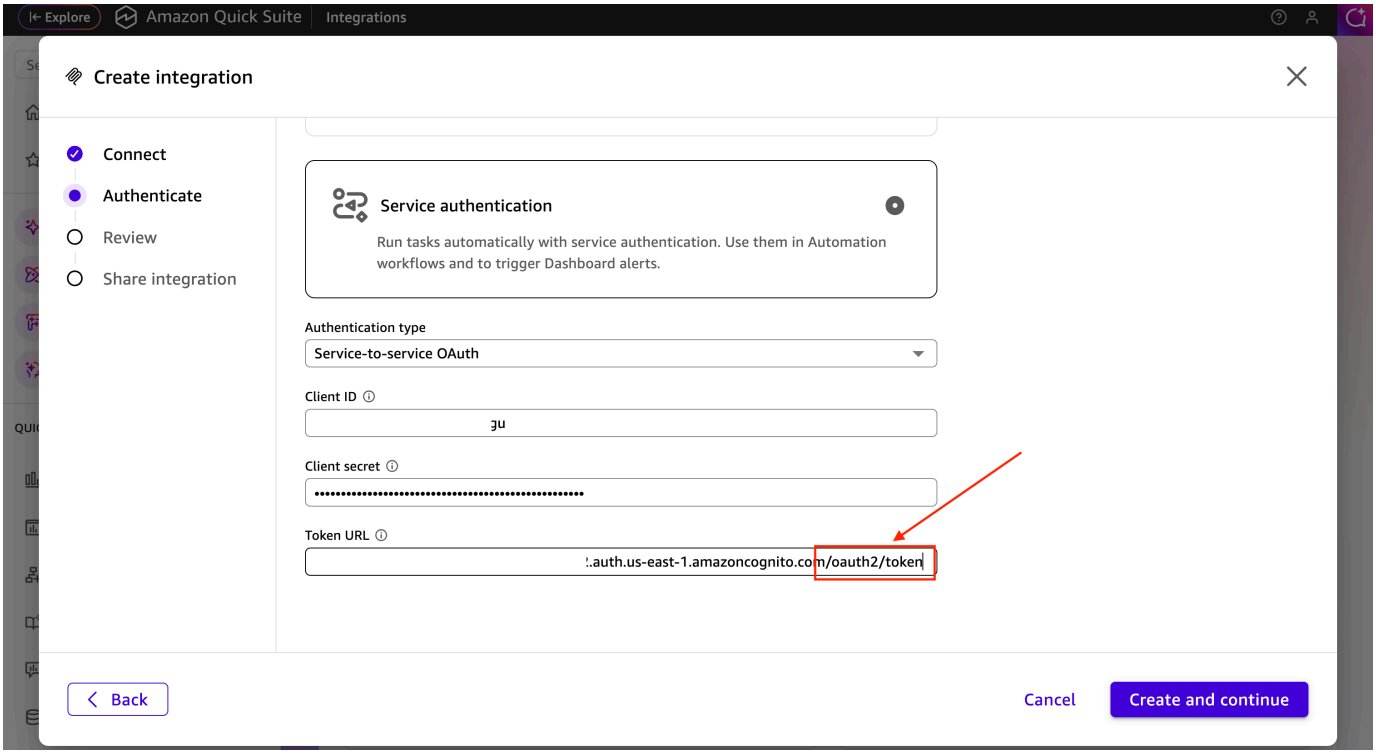


4. MCP 서버 세부 정보를 구성합니다.

- MCP 서버 URL: DLT MCP 엔드포인트



- 인증 유형: 서비스 기반 인증
- 토큰 엔드포인트: Cognito 토큰 엔드포인트 URL
- 클라이언트 ID: m2m 앱 클라이언트의 클라이언트 ID
- 클라이언트 보안 암호: m2m 앱 클라이언트의 클라이언트 보안 암호



5. MCP 서버 작업 구성을 저장합니다.
6. 에이전트에 새 MCP 서버 작업을 추가합니다.

에이전트 시작 및 테스트

1. Amazon Q Suite에서 에이전트를 시작합니다.
2. 자연어 프롬프트를 사용하여 에이전트와 대화를 시작합니다.
3. 에이전트는 MCP 도구를 사용하여 로드 테스트 데이터를 검색하고 분석합니다.

프롬프트 예제

다음 예제에서는 AI 어시스턴트와 상호 작용하여 MCP 인터페이스를 통해 로드 테스트 데이터를 분석하는 방법을 보여줍니다. 특정 테스트 요구 사항에 맞게 테스트 IDs, 날짜 범위 및 기준을 사용자 지정합니다.

사용 가능한 MCP 도구 및 해당 파라미터에 대한 자세한 내용은 개발자 안내서의 MCP 도구 사양을 참조하세요.

간단한 테스트 결과 쿼리

MCP 서버와의 상호 작용은 다음과 같이 간단`Show me the load tests that have completed in the last 24 hours with their associated completion status`하거나 더 설명적일 수 있습니다.

```
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.
```

점진적 공개를 통한 대화형 성능 분석

```
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.
```

프로덕션 준비 확인

```
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]%.
```

성능 추세 분석

```
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.
```

실패한 테스트 문제 해결

```
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].
```

개발자 안내서

이 섹션에서는 솔루션의 소스 코드와 추가 사용자 지정을 제공합니다.

소스 코드

GitHub 리포지토리를 방문하여이 솔루션의 템플릿과 스크립트를 다운로드하고 사용자 지정을 다른 사용자와 공유합니다.

정비

이 솔루션은 각 솔루션 릴리스에 해당하는 고정 버전의 Docker 이미지를 사용합니다. 기본적으로 자동 업데이트가 비활성화되어 배포에 적용되는 버전 업데이트의 시기와 종류를 완벽하게 제어할 수 있습니다. AWS 솔루션 팀은 Amazon ECR 향상된 스캔을 사용하여 기본 이미지 및 설치된 패키지에서 일반적인 취약성 및 노출(CVEs)을 감지합니다. 가능하면 팀은 동일한 버전 태그로 패치된 이미지를 게시하여 릴리스된 솔루션 버전과의 호환성을 손상시키지 않고 CVEs를 해결합니다.

이미지가 동일한 마이너 버전에 패치되면 안정적인 태그가 자동으로 업데이트되고 형식으로 추가 이미지 태그가 생성됩니다`<solution-version>_<date-of-fix>`. 메이저 또는 마이너 버전이 릴리스된 경우 솔루션 버전과 일치하도록 안정적인 태그가 증가하므로 전체 스택 업데이트를 수행하여 최신 이미지 버전을 가져와야 합니다.

배포 중에 자동 업데이트를 옵트인하면 CVE 패치 및 마이너 버그 수정을 포함한 이미지 변경 사항이 일치하는 최신 마이너 릴리스까지 자동으로 적용됩니다.

버전

기본적으로이 솔루션은 자동 업데이트가 비활성화된 상태로 배포됩니다. 즉, 컨테이너 이미지 버전이 배포된 솔루션 버전과 일치하는 특정 버전으로 잠겨 버전 업데이트를 완전히 제어할 수 있습니다.

CloudFormation 배포 중에 예를 선택하여 자동 업데이트를 활성화하면 솔루션은 일치하는 최신 마이너 버전까지 보안 패치와 사소한 버그 수정을 자동으로 수신합니다. 예를 들어 자동 업데이트가 활성화된 버전 4.0.0을 배포하면 4.1.0 이상이 아닌 최대 4.0.x의 업데이트가 수신됩니다.

컨테이너 이미지 버전을 수동으로 제어하려면 태그가 지정된 버전 형식을 사용하여 특정 이미지 버전을 지정하도록 작업 정의를 편집할 수 있습니다. 이렇게 하면 자동 업데이트 설정에 관계없이 특정 이미지 버전에 고정할 수 있습니다.

컨테이너 이미지 사용자 지정

이 솔루션은 AWS에서 관리하는 퍼블릭 Amazon Elastic Container Registry(Amazon ECR) 이미지 리포지토리를 사용하여 구성된 테스트를 실행하는 데 사용되는 이미지를 저장합니다. 컨테이너 이미지를 사용자 지정하려면 이미지를 다시 빌드하여 자체 AWS 계정의 ECR 이미지 리포지토리에 푸시할 수 있습니다.

이 솔루션을 사용자 지정하려면 기본 컨테이너 이미지를 사용하거나 필요에 맞게이 컨테이너를 편집할 수 있습니다. 솔루션을 사용자 지정하는 경우 사용자 지정 솔루션을 빌드하기 전에 다음 코드 샘플을 사용하여 환경 변수를 선언합니다.

```
#!/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
```

컨테이너 이미지를 사용자 지정하도록 선택한 경우 프라이빗 이미지 리포지토리 또는 AWS 계정의 퍼블릭 이미지 리포지토리에서 호스팅할 수 있습니다. 이미지 리소스는 코드 베이스에 있는 `deployment/ecr/distributed-load-testing-on-aws-load-tester` 디렉터리에 있습니다.

이미지를 빌드하여 호스트 대상으로 푸시할 수 있습니다.

- 프라이빗 Amazon ECR 리포지토리 및 이미지는 Amazon ECR 사용 설명서의 Amazon ECR 프라이빗 리포지토리 및 프라이빗 이미지를 참조하세요.
- 퍼블릭 Amazon ECR 리포지토리 및 이미지는 Amazon ECR 퍼블릭 사용 설명서의 Amazon ECR 퍼블릭 리포지토리 및 퍼블릭 이미지를 참조하세요.

자체 이미지를 생성한 후에는 사용자 지정 솔루션을 빌드하기 전에 다음 환경 변수를 선언할 수 있습니다.

```
#!/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
```

다음 예제에서는 컨테이너 파일을 보여줍니다.

```
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"]
```

이 디렉터리에는 컨테이너 파일 외에도 Taurus/Blazemeter 프로그램을 실행하기 전에 Amazon S3에서 테스트 구성을 다운로드하는 다음 bash 스크립트가 포함되어 있습니다.

```
#!/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
```

Dockerfile 및 bash 스크립트 외에도 디렉터리에 두 개의 Python 스크립트가 포함됩니다. 각 작업은 bash 스크립트 내에서 Python 스크립트를 실행합니다. 작업자 작업은 `ecslistener.py` 스크립트를 실행하는 반면 리더 작업은 `ecscontroller.py` 스크립트를 실행합니다. `ecslistener.py` 스크립트는 포트 50000에 소켓을 생성하고 메시지를 기다립니다. `ecscontroller.py` 스크립트는 소켓에 연결하여 테스트 시작 메시지를 작업자 작업에 전송하므로 동시에 시작할 수 있습니다.

분산 로드 테스트 API

이 로드 테스트 솔루션은 테스트 결과 데이터를 안전한 방식으로 노출하는 데 도움이 됩니다. API는 Amazon DynamoDB에 저장된 테스트 데이터에 액세스하기 위한 "정문" 역할을 합니다. APIs를 사용하여 솔루션에 빌드하는 모든 확장 기능에 액세스할 수도 있습니다.

이 솔루션은 식별 및 권한 부여를 위해 Amazon API Gateway와 통합된 Amazon Amazon Cognito 사용자 풀을 사용합니다. 사용자 풀을 API와 함께 사용하는 경우 클라이언트는 유효한 자격 증명 토큰을 제공한 후에만 사용자 풀 활성화 메서드를 호출할 수 있습니다.

API를 통해 직접 테스트를 실행하는 방법에 대한 자세한 내용은 Amazon API Gateway REST API 참조 설명서의 서명 요청을 참조하세요.

솔루션의 API에서 다음 작업을 사용할 수 있습니다.

> **Note**
>
> `testScenario` 및 기타 파라미터에 대한 자세한 내용은 GitHub 리포지토리의 시나리오 및 페이로드 예제를 참조하세요.

스택 정보

- GET/stack-info

시나리오

- GET/시나리오
- POST/시나리오
- 옵션/시나리오
- GET /시나리오/{testId}
- POST/시나리오/{testId}
- DELETE/시나리오/{testId}
- 옵션/시나리오/{testId}

테스트 실행

- GET /scenarios/{testId}/testruns
- GET /scenarios/{testId}/testruns/{testRunId}
- DELETE /scenarios/{testId}/testruns/{testRunId}

기준

- GET /scenarios/{testId}/Baseline
- PUT /시나리오/{testId}/기준
- DELETE /scenarios/{testId}/Baseline

업무

- GET/작업
- 옵션/작업

리전

- GET/리전
- 옵션/리전

GET/stack-info

설명

`GET /stack-info` 작업은 생성 시간, 리전 및 버전을 포함하여 배포된 스택에 대한 정보를 검색합니다. 이 엔드포인트는 프런트 엔드에서 사용됩니다.

응답

200 - 성공

이름	설명
`created_time`	스택 생성 시 ISO 8601 타임스탬프(예: `2025-09-09T19:40:22Z`)
`region`	스택이 배포되는 AWS 리전(예: `us-east-1`)
`version`	배포된 솔루션의 버전(예: `v4.0.0`)

오류 응답

- `403` - 금지됨: 스택 정보에 액세스할 수 있는 권한이 부족함

- 404 - 찾을 수 없음: 스택 정보를 사용할 수 없음
- 500 - 내부 서버 오류

GET/시나리오

설명

GET /scenarios 작업을 통해 테스트 시나리오 목록을 검색할 수 있습니다.

응답

이름	설명
data	각 테스트의 ID, 이름, 설명, 상태, 실행 시간, 태그, 총 실행 및 마지막 실행을 포함한 시나리오 목록

POST/시나리오

설명

POST /scenarios 작업을 통해 테스트 시나리오를 생성하거나 예약할 수 있습니다.

요청 본문

이름	설명
testName	테스트의 이름입니다.
testDescription	테스트에 대한 설명
testTaskConfigs	시나리오에 region 대해 concurrency (병렬 실행 수), taskCount (테스트를 실행하는 데 필요한 작업 수) 및를 지정하는 객체입니다.
testScenario	동시성, 테스트 시간, 호스트 및 테스트 방법을 포함한 테스트 정의

이름	설명
testType	테스트 유형(예: , simplejmeter)
fileType	업로드 파일 유형(예: , nonescript, zip)
tags	테스트를 분류하기 위한 문자열 배열입니다. 최대 길이가 5인 선택적 필드(예: ["blue", "3.0", "critical"])
scheduleDate	테스트를 실행할 날짜입니다. 테스트를 예약하는 경우에만 제공됩니다(예: 2021-02-28).
scheduleTime	테스트를 실행하는 시간입니다. 테스트를 예약하는 경우에만 제공됩니다(예: 21:07).
scheduleStep	일정 프로세스의 단계입니다. 반복 테스트를 예약하는 경우에만 제공됩니다. (사용 가능한 단계에는 create 및가 포함됩니다start.)
cronvalue	반복 예약 사용자 지정을 위한 cron 값입니다. 사용되는 경우 scheduleDate 및 scheduleTime을 생략합니다.
cronExpiryDate	cron이 만료되고 무기한 실행되지 않는 데 필요한 날짜입니다.
recurrence	예약된 테스트의 반복입니다. 반복 테스트를 예약하는 경우에만 제공됩니다(예: , dailyweeklybiweekly, 또는 monthly).

응답

이름	설명
testId	테스트의 고유 ID
testName	테스트의 이름입니다.

이름	설명
status	테스트 상태

옵션/시나리오

설명

OPTIONS /scenarios 작업은 요청에 대한 응답을 올바른 CORS 응답 헤더와 함께 제공합니다.

응답

이름	설명
testId	테스트의 고유 ID
testName	테스트의 이름입니다.
status	테스트 상태

GET /시나리오/{testId}

설명

GET /scenarios/{testId} 작업을 통해 특정 테스트 시나리오의 세부 정보를 검색할 수 있습니다.

요청 파라미터

testId

- 테스트의 고유 ID

  유형: 문자열

  필수 항목 여부: 예

latest

- 파라미터를 쿼리하여 최신 테스트 실행만 반환합니다. 기본값은 true입니다.

  유형: 부울

필수 항목 여부: 아니요

`history`

- 응답에 테스트 실행 기록을 포함하는 쿼리 파라미터입니다. 기본값은 `true`입니다. 기록을 제외`false`하려면 로 설정

  유형: 부울

  필수 항목 여부: 아니요

응답

이름	설명
`testId`	테스트의 고유 ID
`testName`	테스트의 이름입니다.
`testDescription`	테스트에 대한 설명
`testType`	실행 중인 테스트 유형(예: , `simplejmeter`)
`fileType`	업로드되는 파일 유형(예: , `nonescript`, `zip`)
`tags`	테스트를 분류하기 위한 문자열 배열
`status`	테스트 상태
`startTime`	마지막 테스트가 시작된 시간 및 날짜
`endTime`	마지막 테스트가 종료된 시간 및 날짜
`testScenario`	동시성, 테스트 시간, 호스트 및 테스트 방법을 포함한 테스트 정의
`taskCount`	테스트를 실행하는 데 필요한 작업 수
`taskIds`	테스트 실행을 위한 작업 IDs 목록
`results`	테스트의 최종 결과

이름	설명
history	과거 테스트의 최종 결과 목록(일 때 제외됨history=false)
totalRuns	이 시나리오에 대한 총 테스트 실행 수
lastRun	마지막 테스트 실행의 타임스탬프
errorReason	오류가 발생할 때 생성되는 오류 메시지
nextRun	예약된 다음 실행(예: 2017-04-22 17:18:00)
scheduleRecurrence	테스트의 반복(예: , dailyweekly, biweekly, monthly)

POST/시나리오/{testId}

설명

POST /scenarios/{testId} 작업을 통해 특정 테스트 시나리오를 취소할 수 있습니다.

요청 파라미터

testId

- 테스트의 고유 ID

  유형: 문자열

  필수 항목 여부: 예

응답

이름	설명
status	테스트 상태

DELETE /scenarios/{testId}

설명

`DELETE /scenarios/{testId}` 작업을 통해 특정 테스트 시나리오와 관련된 모든 데이터를 삭제할 수 있습니다.

요청 파라미터

`testId`

- 테스트의 고유 ID

  유형: 문자열

  필수 항목 여부: 예

응답

이름	설명
`status`	테스트 상태

옵션/시나리오/{testId}

설명

`OPTIONS /scenarios/{testId}` 작업은 요청에 대한 응답을 올바른 CORS 응답 헤더와 함께 제공합니다.

응답

이름	설명
`testId`	테스트의 고유 ID
`testName`	테스트의 이름입니다.

이름	설명
testDescription	테스트에 대한 설명
testType	실행 중인 테스트 유형(예: , simplejmeter)
fileType	업로드되는 파일 유형(예: , nonescript, zip)
status	테스트 상태
startTime	마지막 테스트가 시작된 시간 및 날짜
endTime	마지막 테스트가 종료된 시간 및 날짜
testScenario	동시성, 테스트 시간, 호스트 및 테스트 방법을 포함한 테스트 정의
taskCount	테스트를 실행하는 데 필요한 작업 수
taskIds	테스트 실행을 위한 작업 IDs 목록
results	테스트의 최종 결과
history	과거 테스트의 최종 결과 목록
errorReason	오류가 발생할 때 생성되는 오류 메시지

GET /scenarios/{testId}/testruns

설명

GET /scenarios/{testId}/testruns 작업은 선택적으로 시간 범위별로 필터링된 특정 테스트 시나리오의 테스트 실행 IDs를 검색합니다. latest=true인 경우는 가장 최근의 단일 테스트 실행만 반환합니다.

요청 파라미터

testId

- 테스트 시나리오 ID

유형: 문자열

필수 항목 여부: 예

`latest`

- 가장 최근 테스트 실행 ID만 반환

  유형: Boolean

  기본값: `false`

  필수 여부: 아니요

`start_timestamp`

- (포함)에서 테스트 실행을 필터링하기 위한 ISO 8601 타임스탬프입니다. 예: `2024-01-01T00:00:00Z`

  유형: 문자열(날짜-시간 형식)

  필수 여부: 아니요

`end_timestamp`

- (포함)까지 테스트 실행을 필터링하는 ISO 8601 타임스탬프입니다. 예: `2024-12-31T23:59:59Z`

  유형: 문자열(날짜-시간 형식)

  필수 여부: 아니요

`limit`

- 반환할 최대 테스트 실행 수(일 때 무시됨`latest=true`)

  유형: 정수(최소: 1, 최대: 100)

  기본값: `20`

  필수 여부: 아니요

`next_token`

- 다음 페이지를 가져오기 위한 이전 응답의 페이지 매김 토큰

  유형: 문자열

필수 항목 여부: 아니요

응답

200 - 성공

이름	설명
testRuns	각각 testRunId (문자열) 및 startTime (ISO 8601 날짜-시간)을 포함하는 테스트 실행 객체 배열
pagination	limit (정수) 및 next_token (문자열 또는 null)을 포함하는 객체입니다. 더 이상 결과가 없는 경우 토큰은 null입니다.

오류 응답

- 400 - 잘못된 타임스탬프 형식 또는 파라미터
- 404 - 테스트 시나리오를 찾을 수 없음
- 500 - 내부 서버 오류

사용 예시

- 최신 테스트 실행만 해당: GET /scenarios/test123/testruns?latest=true
- 시간 범위 내 최신: GET /scenarios/test123/testruns?latest=true&start_timestamp=2024-01-01T00:00:00Z
- 다음 페이지 요청: GET /scenarios/test123/testruns?limit=20&next_token=eyJ0ZXN0SWQiOiJzZVFVeTEyTEtMIiwic3RhcnRUaW1lIjoiMjAyNC0wMS

GET /scenarios/{testId}/testruns/{testRunId}

설명

GET /scenarios/{testId}/testruns/{testRunId} 작업은 특정 테스트 실행에 대한 전체 결과 및 지표를 검색합니다. 선택적으로 더 빠른 응답을 history=false 위해를 사용하여 기록 결과를 생략합니다.

요청 파라미터

testId

- 테스트 시나리오 ID

  유형: 문자열

  필수 항목 여부: 예

testRunId

- 특정 테스트 실행 ID

  유형: 문자열

  필수 항목 여부: 예

history

- 응답에 기록 배열을 포함합니다. 더 빠른 응답을 false 위해 기록을 생략하려면 로 설정

  유형: Boolean

  기본값: true

  필수 여부: 아니요

응답

200 - 성공

이름	설명
testId	테스트의 고유 ID(예: seQUy12LKL)

이름	설명
testRunId	특정 테스트 실행 ID(예: 2DEwHItEne)
testDescription	로드 테스트에 대한 설명
testType	테스트 유형(예: , simplejmeter)
status	테스트 실행 상태: complete, runningfailed, 또는 cancelled
startTime	테스트가 시작된 시간 및 날짜(예: 2025-09-09 21:01:00)
endTime	테스트가 종료된 시간 및 날짜(예: 2025-09-09 21:18:29)
succPercent	성공률(예: 100.00)
testTaskConfigs	region, 및 taskCount 를 포함하는 작업 구성 객체 배열 concurrency
completeTasks	완료된 작업 수에 대한 객체 매핑 리전
results	avg_lt (평균 지연 시간), 백분위수(p0_0, , p50_0, p90_0, p95_0 p99_0p99_9, p100_0), avg_rt (평균 응답 시간), avg_ct (평균 연결 시간), stdev_rt (표준 편차 응답 시간), concurrency , throughput , succ (성공 횟수), fail (실패 횟수), bytes, testDuration , metricS3Location , rc (응답 코드 배열) 및 labels 배열을 포함한 세부 지표가 포함된 객체
testScenario	execution , reporting 및 scenarios 속성을 사용하여 테스트 구성을 포함하는 객체
history	과거 테스트 결과 배열(일 때 제외됨history=false)

오류 응답

- `400` - 잘못된 testId 또는 testRunId
- `404` - 테스트 실행을 찾을 수 없음
- `500` - 내부 서버 오류

DELETE /scenarios/{testId}/testruns/{testRunId}

설명

`DELETE /scenarios/{testId}/testruns/{testRunId}` 작업은 특정 테스트 실행과 관련된 모든 데이터와 아티팩트를 삭제합니다. 테스트 실행 데이터는 DynamoDB에서 제거되지만 S3의 실제 테스트 데이터는 변경되지 않습니다.

요청 파라미터

`testId`

- 테스트 시나리오 ID

  유형: 문자열

  필수 항목 여부: 예

`testRunId`

- 삭제할 특정 테스트 실행 ID

  유형: 문자열

  필수 항목 여부: 예

응답

204 - 성공

테스트 실행이 성공적으로 삭제되었습니다(콘텐츠가 반환되지 않음).

오류 응답

- `400` - 잘못된 testId 또는 testRunId
- `403` - 금지됨: 테스트 실행을 삭제할 수 있는 권한이 부족함

- 404 - 테스트 실행을 찾을 수 없음
- 409 - 충돌: 테스트 실행이 현재 실행 중이며 삭제할 수 없음
- 500 - 내부 서버 오류

GET /scenarios/{testId}/Baseline

설명

GET /scenarios/{testId}/baseline 작업은 시나리오에 대해 지정된 기준 테스트 결과를 검색합니다. data 파라미터에 따라 기준 테스트 실행 ID 또는 전체 기준 결과를 반환합니다.

요청 파라미터

testId

- 테스트 시나리오 ID

  유형: 문자열

  필수 항목 여부: 예

data

- 인 경우 전체 기준 테스트 실행 데이터를 반환하고true, 그렇지 않으면 testRunId만 반환합니다.

  유형: Boolean

  기본값: false

  필수 여부: 아니요

응답

200 - 성공

Whendata=false(기본값):

이름	설명
testId	테스트 시나리오 ID(예: seQUy12LKL)

이름	설명
baselineTestRunId	기준 테스트 실행 ID(예: 2DEwHItEne)

인 경우data=true:

이름	설명
testId	테스트 시나리오 ID(예: seQUy12LKL)
baselineTestRunId	기준 테스트 실행 ID(예: 2DEwHItEne)
baselineData	전체 테스트 실행 결과 객체(와 동일한 구조GET /scenarios/{testId}/testruns/{testRunId})

오류 응답

- 400 - 잘못된 testId 파라미터
- 404 - 테스트 시나리오를 찾을 수 없거나 기준 세트를 찾을 수 없음
- 500 - 내부 서버 오류

PUT /시나리오/{testId}/기준

설명

PUT /scenarios/{testId}/baseline 작업은 특정 테스트 실행을 성능 비교의 기준으로 지정합니다. 시나리오당 하나의 기준만 설정할 수 있습니다.

요청 파라미터

testId

- 테스트 시나리오 ID

  유형: 문자열

  필수 항목 여부: 예

요청 본문

이름	설명
`testRunId`	기준으로 설정할 테스트 실행 ID(예: `2DEwHItEne`)

응답

200 - 성공

이름	설명
`message`	확인 메시지(예: `Baseline set successfully`)
`testId`	테스트 시나리오 ID(예: `seQUy12LKL`)
`baselineTestRunId`	설정된 기준 테스트 실행 ID(예: `2DEwHItEne`)

오류 응답

- `400` - 잘못된 testId 또는 testRunId
- `404` - 테스트 시나리오 또는 테스트 실행을 찾을 수 없음
- `409` - 충돌: 테스트 실행을 기준으로 설정할 수 없음(예: 테스트 실패)
- `500` - 내부 서버 오류

DELETE /scenarios/{testId}/Baseline

설명

`DELETE /scenarios/{testId}/baseline` 작업은 시나리오를 빈 문자열로 설정하여 시나리오의 기준 값을 지웁니다.

요청 파라미터

`testId`

- 테스트 시나리오 ID

  유형: 문자열

  필수 항목 여부: 예

응답

204 - 성공

기준이 성공적으로 지워짐(콘텐츠가 반환되지 않음)

오류 응답

- `400` - 잘못된 testId
- `500` - 내부 서버 오류

GET/작업

설명

`GET /tasks` 작업을 통해 실행 중인 Amazon Elastic Container Service(Amazon ECS) 작업 목록을 검색할 수 있습니다.

응답

이름	설명
`tasks`	테스트 실행을 위한 작업 IDs 목록

옵션/작업

설명

`OPTIONS /tasks` 작업 작업은 요청에 대한 응답을 올바른 CORS 응답 헤더와 함께 제공합니다.

응답

이름	설명
taskIds	테스트 실행을 위한 작업 IDs 목록

GET/리전

설명

GET /regions 작업을 통해 해당 리전에서 테스트를 실행하는 데 필요한 리전 리소스 정보를 검색할 수 있습니다.

응답

이름	설명
testId	리전 ID
ecsCloudWatchLogGroup	리전의 Amazon Fargate 작업에 대한 Amazon CloudWatch 로그 그룹의 이름입니다.
region	테이블의 리소스가 있는 리전
subnetA	리전에 있는 서브넷 중 하나의 ID입니다.
subnetB	리전에 있는 서브넷 중 하나의 ID입니다.
taskCluster	리전에 있는 AWS Fargate 클러스터의 이름
taskDefinition	리전에 있는 작업 정의의 ARN
taskImage	리전에 있는 작업 이미지의 이름입니다.
taskSecurityGroup	리전에 있는 보안 그룹의 ID입니다.

옵션/리전

설명

`OPTIONS /regions` 작업은 요청에 대한 응답을 올바른 CORS 응답 헤더와 함께 제공합니다.

응답

이름	설명
`testId`	리전 ID
`ecsCloudWatchLogGroup`	리전의 Amazon Fargate 작업에 대한 Amazon CloudWatch 로그 그룹의 이름입니다.
`region`	테이블의 리소스가 있는 리전
`subnetA`	리전에 있는 서브넷 중 하나의 ID입니다.
`subnetB`	리전에 있는 서브넷 중 하나의 ID입니다.
`taskCluster`	리전에 있는 AWS Fargate 클러스터의 이름
`taskDefinition`	리전에 있는 작업 정의의 ARN
`taskImage`	리전에 있는 작업 이미지의 이름입니다.
`taskSecurityGroup`	리전에 있는 보안 그룹의 ID입니다.

컨테이너 리소스 증가

로드 테스트가 시뮬레이션할 수 있는 동시 가상 사용자 수(동시성)를 늘리려면 각 Amazon ECS 작업에 할당된 CPU 및 메모리 리소스를 늘려야 합니다. 여기에는 리소스 제한이 더 높은 새 작업 정의 개정을 생성한 다음 향후 테스트 실행에 새 작업 정의를 사용하도록 솔루션의 DynamoDB 구성을 업데이트하는 작업이 포함됩니다.

새 작업 정의 개정 생성

CPU 및 메모리 리소스가 증가한 새 작업 정의를 생성하려면 다음 단계를 따르세요.

1. Amazon Elastic Container Service 콘솔에 로그인합니다.
2. 왼쪽 탐색 메뉴에서 작업 정의를 선택합니다.
3. 이 솔루션에 해당하는 작업 정의 옆의 확인란을 선택합니다. 예: [replaceable]<stackName>`-EcsTaskDefinition-*<system-generated-random-Hash>*.
4. Create new revision(새 수정 생성)을 선택합니다.
5. 새 개정 생성 페이지에서 다음 작업을 수행합니다.
   a. 작업 크기에서 작업 메모리와 작업 CPU를 원하는 값으로 수정합니다. 값이 높을수록 작업당 더 많은 동시 가상 사용자가 허용됩니다.
   b. 컨테이너 정의에서 하드/소프트 메모리 제한을 검토합니다. 이 제한이 원하는 메모리보다 낮으면 컨테이너를 선택합니다.
   c. 컨테이너 편집 대화 상자에서 메모리 제한으로 이동하여 하드 제한을 작업 메모리 할당과 일치하거나 더 작게 업데이트합니다.
   d. 업데이트를 선택합니다.
6. 새 개정 생성 페이지에서 생성을 선택합니다.
7. 작업 정의가 성공적으로 생성되면 버전 번호를 포함하여 전체 작업 정의 ARN을 기록합니다. 예: [replaceable]<stackName>`-EcsTaskDefinition-*<system-generated-random-Hash>*: [replaceable]<system-generated-versionNumber>

DynamoDB 테이블 업데이트

새 작업 정의 개정을 생성한 후에는 향후 테스트 실행에서 새 작업 정의를 사용하도록 솔루션의 DynamoDB 테이블을 업데이트해야 합니다. 업데이트된 작업 정의를 사용하려는 각 AWS 리전에 대해 다음 단계를 반복합니다.

1. DynamoDB 콘솔로 이동합니다.
2. 왼쪽 탐색 창에서 테이블 아래의 항목 탐색을 선택합니다.
3. 이 솔루션과 연결된 scenarios-table DynamoDB 테이블을 선택합니다. 예: [replaceable]<stackName>`-DLTTestRunnerStorageDLTScenariosTable-*<system-generated-random-Hash>*.
4. 새 작업 정의 개정을 생성한 리전에 해당하는 항목을 선택합니다. 예: region-[replaceable]<region-name>`.
5. 항목 편집기에서 taskDefinition 속성을 찾고 이전 섹션에서 기록한 전체 작업 정의 ARN(버전 번호 포함)으로 값을 업데이트합니다.
6. 변경 사항 저장을 선택합니다.

> **Note**
>
> 업데이트된 작업 정의는 새 테스트 실행에만 사용됩니다. 현재 실행 중이거나 예약된 모든 테스트는 이전 작업 정의를 계속 사용합니다.

MCP 도구 사양

분산 로드 테스트 솔루션은 AI 에이전트가 테스트 시나리오 및 결과와 상호 작용할 수 있는 일련의 MCP 도구를 제공합니다. 이러한 도구는 AI 에이전트가 정보를 처리하는 방식에 맞는 상위 수준의 추상화된 기능을 제공하므로 세부 API 계약이 아닌 분석 및 인사이트에 집중할 수 있습니다.

> **Note**
>
> 모든 MCP 도구는 솔루션 데이터에 대한 읽기 전용 액세스를 제공합니다. MCP 인터페이스를 통해 테스트 시나리오 또는 구성을 수정할 수 없습니다.

list_scenarios

설명

이 `list_scenarios` 도구는 기본 메타데이터를 사용하여 사용 가능한 모든 테스트 시나리오 목록을 검색합니다.

엔드포인트

```
GET /scenarios
```

파라미터

없음

응답

이름	설명
`testId`	테스트 시나리오의 고유 식별자

이름	설명
testName	테스트 시나리오의 이름
status	테스트 시나리오의 현재 상태
startTime	테스트가 생성되거나 마지막으로 실행된 시간
testDescription	테스트 시나리오에 대한 설명

get_scenario_details

설명

get_scenario_details 도구는 단일 테스트 시나리오에 대한 테스트 구성과 최신 테스트 실행을 검색합니다.

엔드포인트

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

요청 파라미터

test_id

- 테스트 시나리오의 고유 식별자

  유형: 문자열

  필수 항목 여부: 예

응답

이름	설명
testTaskConfigs	각 리전에 대한 작업 구성
testScenario	테스트 정의 및 파라미터
status	현재 테스트 상태

이름	설명
startTime	테스트 시작 타임스탬프
endTime	테스트 종료 타임스탬프(완료된 경우)

list_test_runs

설명

이 list_test_runs 도구는 최신에서 오래된 것으로 정렬된 특정 테스트 시나리오에 대한 테스트 실행 목록을 검색합니다. 최대 30개의 결과를 반환합니다.

엔드포인트

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

또는

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

요청 파라미터

test_id

- 테스트 시나리오의 고유 식별자

  유형: 문자열

  필수 항목 여부: 예

limit

- 반환할 최대 테스트 실행 수

  유형: 정수

  기본값: 20

  최대: 30

  필수 여부: 아니요

`start_date`

- 특정 날짜부터 실행을 필터링하는 ISO 8601 타임스탬프

  유형: 문자열(날짜-시간 형식)

  필수 여부: 아니요

`end_date`

- 특정 날짜까지 실행을 필터링하는 ISO 8601 타임스탬프

  유형: 문자열(날짜-시간 형식)

  필수 여부: 아니요

응답

이름	설명
`testRuns`	각 실행에 대한 성능 지표 및 백분위수가 포함된 테스트 실행 요약 배열

get_test_run

설명

이 `get_test_run` 도구는 리전 및 엔드포인트 분석을 통해 단일 테스트 실행에 대한 자세한 결과를 검색합니다.

엔드포인트

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

요청 파라미터

`test_id`

- 테스트 시나리오의 고유 식별자

  유형: 문자열

필수 항목 여부: 예

`test_run_id`

- 특정 테스트 실행의 고유 식별자

  유형: 문자열

  필수 항목 여부: 예

응답

이름	설명
`results`	리전별 결과 분석, 엔드포인트별 지표, 성능 백분위수(p50, p90, p95, p99), 성공 및 실패 수, 응답 시간 및 지연 시간, 실행에 사용되는 테스트 구성을 포함한 전체 테스트 실행 데이터

get_latest_test_run

설명

`get_latest_test_run` 도구는 특정 테스트 시나리오에 대한 최신 테스트 실행을 검색합니다.

엔드포인트

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

> **Note**
>
> 결과는 글로벌 보조 인덱스(GSI)를 사용하여 시간별로 정렬되므로 가장 최근의 테스트 실행이 반환됩니다.

요청 파라미터

`test_id`

- 테스트 시나리오의 고유 식별자

유형: 문자열

필수 항목 여부: 예

응답

이름	설명
`results`	와 동일한 형식의 최신 테스트 실행 데이터 `get_test_run`

get_baseline_test_run

설명

`get_baseline_test_run` 도구는 특정 테스트 시나리오에 대한 기준 테스트 실행을 검색합니다. 기준은 성능 비교 목적으로 사용됩니다.

엔드포인트

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

요청 파라미터

`test_id`

- 테스트 시나리오의 고유 식별자

  유형: 문자열

  필수 항목 여부: 예

응답

이름	설명
`baselineData`	지정된 기준 실행의 모든 지표 및 구성을 포함하여 비교를 위한 기준 테스트 실행 데이터

get_test_run_artifacts

설명

get_test_run_artifacts 도구는 로그, 오류 파일 및 결과를 포함한 테스트 아티팩트에 액세스하기 위한 Amazon S3 버킷 정보를 검색합니다.

엔드포인트

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

요청 파라미터

test_id

- 테스트 시나리오의 고유 식별자

  유형: 문자열

  필수 항목 여부: 예

test_run_id

- 특정 테스트 실행의 고유 식별자

  유형: 문자열

  필수 항목 여부: 예

응답

이름	설명
bucketName	아티팩트가 저장되는 S3 버킷 이름
testRunPath	현재 아티팩트 스토리지의 경로 접두사(버전 4.0 이상)
testScenarioPath	레거시 아티팩트 스토리지의 경로 접두사(버전 4.0 이전)

> **Note**
>
> 모든 MCP 도구는 기존 API 엔드포인트를 활용합니다. MCP 기능을 지원하기 위해 기본 APIs를 수정할 필요가 없습니다.

레퍼런스

이 섹션에는 데이터 수집, 관련 리소스에 대한 포인터,이 솔루션에 기여한 빌더 목록에 대한 정보가 포함되어 있습니다.

데이터 수집

이 솔루션은이 솔루션 사용에 대한 운영 지표("데이터")를 AWS로 전송합니다. 당사는 고객이이 솔루션과 관련 서비스 및 제품을 사용하는 방법을 더 잘 이해하기 위해이 데이터를 사용합니다. AWS의이 데이터 수집에는 AWS 개인 정보 보호 고지가 적용됩니다.

기여자

- Tom Nightingale
- 페르난도 딩글러
- 이범석
- George Lenz
- Erin McGill
- Dimitri Lopez
- 카미아 자바리
- Bassem Wanis
- Garvit Singh
- Nikhil Reddy
- 사이몬 크롤
- Ahern Knox
- Ian Downard
- Owen Brady
- 짐 타리오
- Thyag Ramachandran
- 양킨
- 제임스 왕

용어집

이 용어집은 AWS 구현 가이드의 Distributed Load Testing 전체에서 사용되는 약어와 약어를 정의합니다.

기술 프로토콜 및 형식

AGPL

Affero 일반 공개 라이선스. K6에서 사용하는 오픈 소스 소프트웨어 라이선스입니다.

API

애플리케이션 프로그래밍 인터페이스. 소프트웨어 애플리케이션을 구축하고 서로 다른 시스템 간의 통신을 활성화하기 위한 프로토콜 및 도구 세트입니다.

CLI

명령줄 인터페이스. 소프트웨어 및 운영 체제와 상호 작용하기 위한 텍스트 기반 인터페이스입니다.

CORS

교차 오리진 리소스 공유. 한 오리진에서 실행되는 웹 애플리케이션이 다른 오리진의 리소스에 액세스하도록 허용하거나 제한하는 보안 기능입니다.

CSV

쉼표로 구분된 값입니다. 테이블 형식 데이터를 일반 텍스트로 저장하는 데 사용되는 파일 형식으로, 일반적으로 데이터 내보내기에 사용됩니다.

gRPC

gRPC 원격 프로시저 호출. 원격 프로시저 호출을 위한 고성능 오픈 소스 프레임워크입니다.

HTTP

하이퍼텍스트 전송 프로토콜. 월드 와이드 웹에서 데이터를 전송하는 데 사용되는 파운데이션 프로토콜입니다.

HTTPS

HTTP 보안. 네트워크를 통한 보안 통신을 위해 암호화를 사용하는 HTTP의 확장입니다.

JSON

JavaScript Object Notation(JSON)입니다. 사람이 쉽게 읽고 쓸 수 있고 머신이 쉽게 구문 분석하고 생성할 수 있는 경량 데이터 교환 형식입니다.

JWT

JSON 웹 토큰. 인증 및 권한 부여를 위해 두 당사자 간에 전송될 클레임을 나타내는 간단한 URL 보호 수단입니다.

OAuth

권한 부여를 엽니다. 토큰 기반 인증 및 권한 부여에 일반적으로 사용되는 액세스 위임의 공개 표준입니다.

REST

대표 상태 전송. 상태 비저장 통신 및 표준 HTTP 메서드를 사용하여 네트워크 애플리케이션을 설계하기 위한 아키텍처 스타일입니다.

SSE

서버 전송 이벤트. 클라이언트가 HTTP 연결을 통해 서버에서 자동 업데이트를 수신할 수 있는 서버 푸시 기술입니다.

UI

사용자 인터페이스. 시각적 요소 및 컨트롤은 사용자가 소프트웨어 애플리케이션과 상호 작용하는 방식을 제어합니다.

URL

균일한 리소스 로케이터입니다. 인터넷의 리소스에 액세스하는 데 사용되는 주소입니다.

XML

확장 가능한 마크업 언어. 사람이 읽을 수 있고 기계가 읽을 수 있는 형식으로 문서를 인코딩하기 위한 규칙을 정의하는 마크업 언어입니다.

테스트 및 데이터베이스 용어

FTP

파일 전송 프로토콜. 클라이언트와 서버 간에 파일을 전송하는 데 사용되는 표준 네트워크 프로토콜입니다.

쿼리

글로벌 보조 인덱스. 대체 키를 사용하여 데이터를 쿼리할 수 있는 DynamoDB 기능입니다.

JDBC

Java 데이터베이스 연결. 데이터베이스와 쿼리를 연결하고 실행하기 위한 Java API입니다.

JMS

Java 메시지 서비스. 둘 이상의 클라이언트 간에 메시지를 보내기 위한 Java API입니다.

TPS

초당 트랜잭션 수입니다. 시스템이 1초 동안 처리할 수 있는 트랜잭션 수의 측정치입니다.

AWS 및 시스템 약관

ARN

Amazon 리소스 이름() AWS 서비스에서 리소스를 지정하는 데 사용되는 AWS 리소스의 고유 식별자입니다.

ISO

국제 표준화 기구. 국제 표준을 개발하는 독립적인 비정부 조직입니다. 이 가이드에서 ISO 8601 타임스탬프 형식에 대해 참조되었습니다.

SLA

서비스 수준 계약. 예상되는 서비스 수준을 정의하는 서비스 공급자와 고객 간의 약정입니다.

UUID

범용 고유 식별자. 컴퓨터 시스템에서 정보를 고유하게 식별하는 데 사용되는 128비트 숫자입니다.

vCPU

가상 중앙 처리 장치. 물리적 CPU 처리 능력의 일부를 나타내는 가상 머신 또는 컨테이너에 할당된 가상 프로세서입니다.

로드 테스트 용어

동시성

작업당 동시 가상 사용자 수입니다. 이 파라미터는 각 Fargate 태스크가 로드 테스트 중에 생성하는 시뮬레이션된 사용자 수를 제어합니다.

리전 스택

다중 리전 로드 테스트를 위한 테스트 인프라를 제공하기 위해 AWS 리전에 배포된 CloudFormation 스택입니다.

작업 수

테스트 시나리오를 실행하기 위해 시작된 Fargate 컨테이너(작업) 수입니다. 생성된 총 로드는 작업 수에 동시성을 곱한 값과 같습니다.

테스트 시나리오

테스트 유형, 대상 엔드포인트, 작업 수, 동시성, 기간 및 기타 파라미터를 포함하여 구성된 로드 테스트입니다.

개정

버전별 개선 사항 및 수정 사항을 추적하려면 GitHub 리포지토리의 CHANGELOG.md를 방문하세요.

고지 사항

고객은 본 문서의 정보를 독립적으로 평가할 책임이 있습니다. 이 문서는 (a) 정보 제공 목적으로만 사용되며, (b) 예고 없이 변경될 수 있는 AWS의 현재 제품 제공 및 관행을 나타내며, (c) AWS 및 그 계열사, 공급업체 또는 라이선스 제공자로부터 어떠한 약속이나 보장도 생성하지 않습니다. AWS 제품 또는 서비스는 명시적이든 묵시적이든 어떠한 종류의 보증, 표현 또는 조건 없이 "있는 그대로" 제공됩니다. 고객에 대한 AWS의 책임과 책임은 AWS 계약의 적용을 받으며,이 문서는 AWS와 고객 간의 계약의 일부이거나 수정하지 않습니다.

AWS의 분산 로드 테스트는 Apache Software Foundation에서 제공되는 Apache 라이선스 버전 2.0의 약관에 따라 라이선스가 부여됩니다.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.