Panduan Implementasi

Pengujian Beban Terdistribusi di AWS

Pengujian Beban Terdistribusi di AWS: Panduan Implementasi

Table of Contents

Ikhtisar solusi 1
Fitur 2
Manfaat 4
Kasus penggunaan 5
Konsep dan definisi 6
Gambaran umum arsitektur 7
Diagram arsitektur 7
Pertimbangan desain AWS Well-Architected 9
Keunggulan operasional 9
Keamanan 10
Keandalan 11
Efisiensi kinerja 11
Optimalisasi biaya 11
Keberlanjutan 12
Detail arsitektur 13
Ujung depan 13
Muat pengujian API 13
Konsol web 14
MCP Server (Opsional) 14
Backend 14
Pipa gambar kontainer 14
Infrastruktur pengujian 15
Mesin uji beban 15
MCP Server 16
AgentCore Gerbang AWS 16
Lambda Server MCP DLT 16
Integrasi otentikasi 17
Layanan AWS dalam solusi ini 17
Cara Kerja Pengujian Beban Terdistribusi di AWS 18
Alur kerja MCP Server (Opsional) 21
Pertimbangan desain 22
Aplikasi-aplikasi yang didukung 22
Jenis tes 23
Tes penjadwalan 25

Tes bersamaan 26
Manajemen pengguna 26
Penyebaran regional 26
Rencanakan penyebaran Anda 27
Biaya 27
Biaya tambahan MCP Server (Opsional) 28
Keamanan 29
Peran IAM 29
Amazon CloudFront 29
Amazon API Gateway 30
Grup keamanan AWS Fargate 30
Amazon VPC 30
Tes stress jaringan 32
Membatasi akses ke antarmuka pengguna publik 33
Keamanan MCP Server (Opsional) 33
Wilayah AWS yang Didukung 33
MCP Server mendukung AWS Regions (Opsional) 34
Kuota 35
Kuota untuk layanan AWS dalam solusi ini 35
CloudFormation Kuota AWS 35
Kuota pengujian beban 35
Tes bersamaan 26
Kebijakan pengujian Amazon EC2 36
Kebijakan pengujian CloudFront beban Amazon 36
Memantau solusi pasca-penerapan 36
Menyiapkan CloudWatch alarm 37
Libatkan Ahli 37
AWS Countdown Premium Keterlibatan Jangka Pendek untuk Pengujian Beban Terdistribusi di AWS 37
Terapkan solusinya 40
Ikhtisar proses penyebaran 40
Terapkan menggunakan AWS Launch Wizard 41
Terapkan menggunakan AWS CloudFormation 41
CloudFormation Templat AWS 41
Luncurkan tumpukan 42
Penyebaran Multi-Wilayah 45

Perbarui solusinya 49
Perbarui menggunakan AWS Launch Wizard 49
Perbarui menggunakan AWS CloudFormation 49
Pemecahan masalah pembaruan dari versi sebelum v3.3.0 50
Memperbarui tumpukan regional 51
Manajer Aplikasi AWS Systems Manager 51
Pemecahan masalah 53
Resolusi masalah yang diketahui 53
Hubungi AWS Support 55
Buat kasus 55
Bagaimana kami bisa membantu? 55
Informasi tambahan 56
Bantu kami menyelesaikan kasus Anda lebih cepat 56
Selesaikan sekarang atau hubungi kami 56
Copot pemasangan solusinya 57
Menggunakan Konsol Manajemen AWS 57
AWS CloudFormation 57
AWS Launch Wizard 57
Menggunakan AWS Command Line Interface 57
Menghapus bucket Amazon S3 58
Gunakan solusinya 59
Buat skenario pengujian 59
Langkah 1: Pengaturan umum 59
Langkah 2: Konfigurasi skenario 61
Langkah 3: Bentuk lalu lintas 63
Langkah 4: Tinjau dan buat 67
Jalankan skenario pengujian 67
Tampilan detail skenario 68
Alur kerja eksekusi uji 68
Status uji coba 69
Pemantauan dengan data langsung 69
Membatalkan tes 71
Jelajahi hasil tes 72
Metrik ringkasan uji coba 72
Tabel uji coba 73
Perbandingan dasar 73

Hasil tes terperinci 73
Tab kesalahan 75
Tab artefak 75
Struktur hasil S3 75
Integrasi server MCP 76
Langkah 1: Dapatkan titik akhir MCP dan token akses 76
Langkah 2: Uji dengan MCP Inspector 77
Langkah 3: Konfigurasikan klien pengembangan AI 79
Contoh petunjuk 85
Panduan developer 88
Kode sumber 88
Maintenance 88
Versi 88
Kustomisasi gambar kontainer 89
API pengujian beban terdistribusi 97
DAPATKAN /tumpukan-info 98
DAPATKAN /skenario 99
POST/skenario 100
PILIHAN/skenario 101
DAPATKAN /scenarios/ {teSid} 102
POSTING /scenarios/ {TESid} 104
HAPUS /scenarios/ {teSid} 105
PILIHAN /scenarios/ {TESid} 105
DAPATKAN /scenarios/ {teSTid} /testruns 106
DAPATKAN /scenarios/ {teSid} /testruns/ {} testRunId 109
HAPUS /scenarios/ {teSTid} /testruns/ {} testRunId 111
DAPATKAN /scenarios/ {teSid} /baseline 112
PUT /scenarios/ {teSid} /baseline 114
HAPUS /scenarios/ {teSid} /baseline 115
DAPATKAN /tugas 116
PILIHAN/tugas 116
DAPATKAN /wilayah 116
PILIHAN/wilayah 117
Meningkatkan sumber daya kontainer 118
Buat revisi definisi tugas baru 118
Perbarui tabel DynamoDB 119

Spesifikasi alat MCP 120
list_scenario 120
get_scenario_details 121
list_test_runs 122
get_test_run 123
get_latest_test_run 124
get_baseline_test_run 125
get_test_run_artefak 126
Referensi 128
Pengumpulan data 128
Kontributor 128
Glosarium 129
Protokol dan Format Teknis 129
Ketentuan Pengujian dan Database 130
AWS dan Ketentuan Sistem 131
Ketentuan Pengujian Beban 132
Revisi 133
Pemberitahuan 134
..... cxxxv

Otomatiskan pengujian aplikasi perangkat lunak Anda dalam skala besar

Tanggal publikasi: Desember 2025

Pengujian Beban Terdistribusi di AWS membantu Anda mengotomatiskan pengujian kinerja aplikasi perangkat lunak dalam skala besar untuk mengidentifikasi kemacetan sebelum merilis aplikasi. Solusi ini mensimulasikan ribuan pengguna terhubung yang menghasilkan permintaan HTTP dengan kecepatan berkelanjutan tanpa perlu menyediakan server.

Solusi ini memanfaatkan Amazon Elastic Container Service (Amazon ECS) Container Service (Amazon ECS) di AWS Fargate untuk menerapkan kontainer yang menjalankan simulasi uji beban Anda dan menawarkan kemampuan berikut:

- Terapkan Amazon ECS di kontainer AWS Fargate yang berjalan secara independen untuk menguji kapasitas muat aplikasi Anda.
- Simulasikan puluhan ribu pengguna bersamaan di beberapa Wilayah AWS yang menghasilkan permintaan secara terus menerus.
- Sesuaikan pengujian aplikasi Anda menggunakan JMeter, K6, skrip pengujian Locust, atau konfigurasi titik akhir HTTP sederhana.
- Jadwalkan tes beban untuk segera dijalankan, pada tanggal dan waktu yang akan datang, atau pada jadwal berulang.
- Jalankan beberapa tes beban secara bersamaan di berbagai skenario dan wilayah.

Panduan implementasi ini memberikan gambaran umum tentang solusi Distributed Load Testing on AWS, arsitektur referensi dan komponennya, pertimbangan untuk merencanakan penerapan, dan langkah-langkah konfigurasi untuk menerapkan solusi ke Amazon Web Services (AWS) Cloud. Ini mencakup tautan ke CloudFormation templat AWS yang meluncurkan dan mengonfigurasi layanan AWS yang diperlukan untuk menerapkan solusi ini menggunakan praktik terbaik AWS untuk keamanan dan ketersediaan.

Audiens yang dituju untuk menggunakan fitur dan kemampuan solusi ini di lingkungan mereka mencakup arsitek infrastruktur TI, administrator, dan DevOps profesional yang memiliki pengalaman praktis dalam merancang di AWS Cloud.

Gunakan tabel navigasi ini untuk menemukan jawaban atas pertanyaan-pertanyaan ini dengan cepat:

Jika kau mau.	Baca.
Ketahui biaya untuk menjalankan solusi ini. Perkiraan biaya untuk menjalankan solusi ini di Wilayah AS Timur (Virginia N.) adalah USD $30,90 per bulan untuk sumber daya AWS.	Biaya
Pahami pertimbangan keamanan untuk solusi ini.	Keamanan
Ketahui cara merencanakan kuota untuk solusi ini.	Kuota
Ketahui Wilayah AWS mana yang mendukung solusi ini.	Wilayah AWS yang Didukung
Pelajari tentang Server MCP opsional untuk analisis pengujian beban berbantuan AI.	Integrasi server MCP
Lihat atau unduh CloudFormation templat AWS yang disertakan dalam solusi ini untuk secara otomatis menerapkan sumber daya infrastruktur (“tumpukan”) untuk solusi ini.	CloudFormation Templat AWS
Akses kode sumber dan secara opsional gunakan AWS Cloud Development Kit (AWS CDK) untuk menerapkan solusi.	GitHub repositori

Fitur

Solusinya menyediakan fitur-fitur berikut:

Beberapa Test Framework Support

Mendukung JMeter, skrip pengujian K6, dan Locust, serta pengujian titik akhir HTTP sederhana tanpa memerlukan skrip khusus. Untuk informasi selengkapnya, lihat Jenis pengujian di bagian Detail arsitektur.

Simulasi Beban Pengguna Tinggi

Mensimulasikan puluhan ribu pengguna virtual bersamaan untuk menguji stres aplikasi Anda dalam kondisi beban yang realistis.

Distribusi Beban Multi-Wilayah

Mendistribusikan pengujian beban di beberapa Wilayah AWS untuk mensimulasikan lalu lintas pengguna yang terdistribusi secara geografis dan menilai kinerja global.

Penjadwalan Uji Fleksibel

Menjadwalkan tes untuk segera dijalankan, pada tanggal dan waktu masa depan tertentu, atau pada jadwal berulang menggunakan ekspresi cron untuk pengujian regresi otomatis.

Pemantauan Waktu Nyata

Menyediakan streaming data langsung opsional untuk memantau kemajuan pengujian dengan metrik waktu nyata termasuk waktu respons, jumlah pengguna virtual, dan tingkat keberhasilan permintaan.

Hasil Tes Komprehensif

Menampilkan hasil pengujian terperinci dengan metrik kinerja, persentil (p50, p90, p95, p99), analisis kesalahan, dan artefak yang dapat diunduh untuk analisis offline.

Perbandingan Dasar

Menetapkan uji dasar berjalan untuk perbandingan kinerja untuk melacak peningkatan atau regresi dari waktu ke waktu.

Fleksibilitas Titik Akhir

Menguji titik akhir HTTP atau HTTPS di seluruh Wilayah AWS, lingkungan lokal, atau penyedia cloud lainnya.

Konsol Web Intuitif

Menyediakan konsol berbasis web untuk membuat, mengelola, dan memantau tes tanpa memerlukan interaksi baris perintah.

Analisis Berbantuan AI (Opsional)

Terintegrasi dengan alat pengembangan AI melalui server Model Context Protocol (MCP) untuk analisis cerdas data pengujian beban.

Dukungan Multiple Protocol

Mendukung berbagai protokol termasuk HTTP, HTTPS,, JDBC WebSocket, JMS, FTP, dan gRPC melalui skrip uji kustom.

Manfaat

Solusinya memberikan manfaat sebagai berikut:

Pengujian Kinerja Komprehensif

Mendukung pengujian beban, pengujian stres, dan pengujian ketahanan untuk mengevaluasi kinerja aplikasi secara menyeluruh dalam berbagai kondisi.

Deteksi Masalah Dini

Mengidentifikasi kemacetan kinerja, kebocoran memori, dan masalah skalabilitas sebelum penerapan produksi, mengurangi risiko pemadaman.

Simulasi Penggunaan Dunia Nyata

Secara akurat mensimulasikan perilaku pengguna dunia nyata dan pola lalu lintas untuk memvalidasi kinerja aplikasi dalam kondisi realistis.

Performance Insights yang Dapat Ditindaklanjuti

Menyediakan metrik terperinci, persentil, dan analisis kesalahan untuk memahami perilaku aplikasi dan memandu upaya pengoptimalan.

Alur Kerja Pengujian Otomatis

Memungkinkan pengujian terjadwal dan berulang untuk pemantauan kinerja berkelanjutan dan pengujian regresi tanpa intervensi manual.

Infrastruktur Hemat Biaya

Menggunakan kontainer AWS Fargate tanpa server dengan pay-per-use harga, menghilangkan kebutuhan akan infrastruktur pengujian khusus dan biaya berlangganan yang sedang berlangsung.

Penerapan Tes Cepat

Menyebarkan dan menskalakan infrastruktur pengujian dalam hitungan menit tanpa menyediakan atau mengelola server.

Interogasi Hasil Tes yang Mudah

Terintegrasi dengan alat pengembangan AI melalui server Model Context Protocol (MCP) opsional, memungkinkan kueri bahasa alami dan analisis cerdas data pengujian beban untuk wawasan dan pemecahan masalah yang lebih cepat.

Kasus penggunaan

Validasi Pra-Produksi

Uji aplikasi web dan seluler dalam kondisi pemuatan seperti produksi sebelum meluncurkan versi baru untuk memvalidasi kinerja dan mengidentifikasi masalah.

Perencanaan Kapasitas

Tentukan jumlah maksimum pengguna bersamaan yang dapat didukung aplikasi Anda dengan infrastruktur saat ini dan identifikasi kapan penskalaan diperlukan.

Verifikasi Beban Puncak

Pastikan infrastruktur Anda dapat menangani beban puncak, lonjakan lalu lintas musiman, atau lonjakan permintaan yang tidak terduga tanpa penurunan kinerja.

Optimalisasi Kinerja

Identifikasi kemacetan kinerja seperti kueri basis data yang lambat, eksekusi kode yang tidak efisien, latensi jaringan, atau kendala sumber daya.

Pengujian Regresi

Jadwalkan pengujian beban berulang untuk mendeteksi regresi kinerja yang diperkenalkan oleh penerapan kode baru atau perubahan infrastruktur.

Penilaian Kinerja Global

Evaluasi kinerja aplikasi dari berbagai wilayah geografis untuk memastikan pengalaman pengguna yang konsisten bagi khalayak global.

Pengujian Beban API

Uji REST APIs, titik akhir GraphQL, atau layanan mikro untuk memvalidasi waktu respons, throughput, dan tingkat kesalahan yang sedang dimuat.

Integrasi Pipa CI/CD

Integrasikan pengujian kinerja otomatis ke dalam pipeline integrasi dan penerapan berkelanjutan untuk menangkap masalah kinerja di awal siklus pengembangan.

Pengujian Layanan Pihak Ketiga

Uji kinerja dan keandalan pihak ketiga APIs atau layanan yang bergantung pada aplikasi Anda dalam berbagai kondisi pemuatan.

Konsep dan definisi

Bagian ini menjelaskan konsep-konsep kunci dan mendefinisikan terminologi khusus untuk solusi ini:

skenario

Definisi pengujian termasuk nama pengujian, deskripsi, jumlah tugas, konkurensi, Wilayah AWS, ramp-up, penahanan, jenis pengujian, tanggal jadwal, dan konfigurasi pengulangan.

jumlah tugas

Jumlah kontainer yang akan diluncurkan di cluster Fargate untuk menjalankan skenario pengujian. Tugas tambahan tidak akan dibuat setelah batas akun pada sumber daya Fargate tercapai. Namun, tugas yang sudah berjalan akan terus berlanjut.

konkurensi

Konkurensi (jumlah pengguna virtual bersamaan per tugas). Konkurensi yang disarankan berdasarkan pengaturan default adalah 200. Konkurensi dibatasi oleh CPU dan memori. Untuk tes berdasarkan Apache JMeter, konkurensi yang lebih tinggi meningkatkan memori yang digunakan oleh JVM pada tugas ECS. Definisi Tugas ECS default membuat tugas dengan memori 4 GB. Disarankan untuk memulai dengan nilai konkurensi yang lebih rendah untuk 1 tugas dan memantau CloudWatch metrik ECS untuk Task Cluster. Lihat metrik pemanfaatan klaster Amazon ECS.

ramp-up

Periode waktu untuk secara bertahap meningkat dari nol ke tingkat konkurensi target.

tahan untuk

Periode waktu untuk mempertahankan level konkurensi target setelah ramp-up selesai.

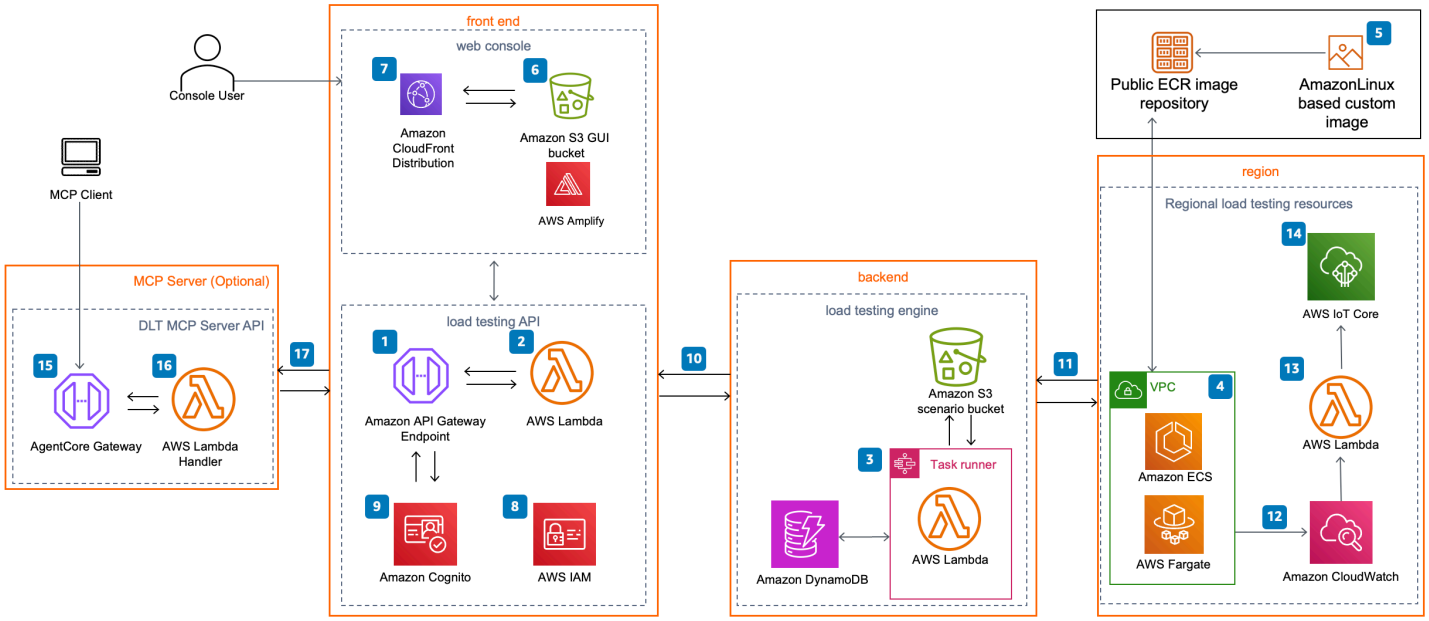
Untuk referensi umum istilah AWS, lihat Daftar Istilah AWS.

Gambaran umum arsitektur

Diagram arsitektur

Menerapkan solusi ini dengan parameter default akan menerapkan komponen berikut di akun AWS Anda.

Pengujian Beban Terdistribusi pada arsitektur AWS di AWS



> **Note**
>
> CloudFormation Sumber daya AWS dibuat dari konstruksi AWS Cloud Development Kit (AWS CDK).

Alur proses tingkat tinggi untuk komponen solusi yang digunakan dengan CloudFormation template AWS adalah sebagai berikut:

1. API penguji beban terdistribusi memanfaatkan Amazon API Gateway untuk menjalankan layanan mikro solusi (fungsi AWS Lambda).
2. Layanan mikro menyediakan logika bisnis untuk mengelola data pengujian dan menjalankan tes.

3. Layanan mikro ini berinteraksi dengan Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB, dan AWS Step Functions untuk menyimpan detail dan hasil skenario pengujian serta mengatur eksekusi pengujian.
4. Topologi jaringan Amazon Virtual Private Cloud (Amazon VPC) disebarkan berisi kontainer Amazon Elastic Container Service (Amazon ECS) solusi yang berjalan di AWS Fargate.
5. Wadah menggunakan gambar dasar Amazon Linux 2023 dengan kerangka pengujian beban Taurus diinstal. Taurus adalah kerangka kerja otomatisasi pengujian open-source yang mendukung JMeter, K6, Locust, dan alat pengujian lainnya. Gambar kontainer sesuai dengan Open Container Initiative (OCI) dan dihosting oleh AWS di repositori publik Amazon Elastic Container Registry (Amazon ECR). Untuk informasi selengkapnya, lihat kustomisasi gambar Container.
6. Konsol web yang didukung oleh AWS Amplify diterapkan ke dalam bucket S3 yang dikonfigurasi untuk hosting web statis.
7. Amazon CloudFront menyediakan akses publik yang aman ke konten bucket situs web solusi.
8. Selama konfigurasi awal, solusi membuat peran administrator default (peran IAM) dan mengirimkan undangan akses ke alamat email pengguna yang ditentukan pelanggan.
9. Kumpulan pengguna Amazon Cognito mengelola akses pengguna ke konsol, API penguji beban terdistribusi, dan Server MCP.
10. Setelah menerapkan solusi ini, Anda dapat menggunakan konsol web atau APIs untuk membuat dan menjalankan skenario pengujian yang menentukan serangkaian tugas.
11. Layanan mikro menggunakan skenario pengujian ini untuk menjalankan tugas ECS di Fargate di Wilayah yang ditentukan.
12. Ketika pengujian selesai, solusi menyimpan hasil di S3 dan DynamoDB dan mencatat output di Amazon. CloudWatch
13. Jika Anda mengaktifkan opsi data langsung, solusi akan mengirimkan CloudWatch log dari tugas Fargate ke fungsi Lambda selama pengujian untuk setiap Wilayah tempat pengujian dijalankan.
14. Fungsi Lambda menerbitkan data ke topik yang sesuai di AWS IoT Core di Wilayah tempat tumpukan utama digunakan. Konsol web berlangganan topik dan menampilkan data waktu nyata saat pengujian berjalan.

> **Note**
>
> Langkah-langkah berikut menjelaskan integrasi MCP Server opsional untuk analisis pengujian beban berbantuan AI. Komponen ini hanya digunakan jika Anda memilih opsi MCP Server selama penerapan solusi.

15. Klien MCP (alat pengembangan AI) terhubung ke titik akhir AWS AgentCore Gateway untuk mengakses data solusi Pengujian Beban Terdistribusi melalui Protokol Konteks Model. AgentCore Gateway memvalidasi token otentikasi Cognito pengguna untuk memastikan akses resmi ke server MCP.
16. Setelah otentikasi berhasil, AgentCore Gateway meneruskan permintaan alat MCP ke fungsi Lambda Server MCP DLT. Fungsi Lambda mengembalikan data terstruktur ke AgentCore Gateway, yang mengirimkannya kembali ke klien MCP untuk analisis dan wawasan yang dibantu AI.
17. Fungsi Lambda memproses permintaan dan menanyakan sumber daya AWS yang sesuai (tabel DynamoDB, bucket S3, atau CloudWatch log) untuk mengambil data pengujian beban yang diminta.

Pertimbangan desain AWS Well-Architected

Solusi ini menggunakan praktik terbaik dari AWS Well-Architected Framework, yang membantu pelanggan merancang dan mengoperasikan beban kerja yang andal, aman, efisien, dan hemat biaya di cloud.

Bagian ini menjelaskan bagaimana prinsip-prinsip desain dan praktik terbaik dari Well-Architected Framework menguntungkan solusi ini.

Keunggulan operasional

Bagian ini menjelaskan bagaimana kami merancang solusi ini menggunakan prinsip dan praktik terbaik dari pilar keunggulan operasional.

- Semua sumber daya didefinisikan sebagai infrastruktur sebagai kode menggunakan CloudFormation templat AWS yang dihasilkan dari konstruksi AWS CDK.
- Solusi ini mendorong metrik ke berbagai CloudWatch tahap untuk memberikan pengamatan ke dalam fungsi Lambda, tugas ECS, bucket S3, dan komponen solusi lainnya.

Keamanan

Bagian ini menjelaskan bagaimana kami merancang solusi ini menggunakan prinsip dan praktik terbaik pilar keamanan.

- Cognito mengautentikasi dan mengotorisasi pengguna konsol web dan permintaan API.
- Semua komunikasi antar layanan menggunakan peran AWS Identity and Access Management (IAM) dengan akses hak istimewa paling sedikit, hanya berisi izin minimum yang diperlukan.
- Semua penyimpanan data, termasuk bucket S3 dan tabel DynamoDB, mengenkripsi data saat istirahat menggunakan kunci yang dikelola AWS.
- Pencatatan, penelusuran, dan pembuatan versi diaktifkan jika berlaku untuk tujuan audit dan kepatuhan.
- Akses jaringan bersifat pribadi secara default dengan titik akhir VPC diaktifkan jika tersedia untuk menjaga lalu lintas dalam jaringan AWS.

Note

Solusi ini membuat beberapa grup CloudWatch log dengan periode retensi yang bervariasi berdasarkan volume log dan pertimbangan biaya:

Jenis Log	Periode Retensi
Wawasan kontainer ECS	1 hari
Step Functions, log kustom ECS, log akses API Gateway	1 tahun
Log runtime Lambda	2 tahun
Log eksekusi API Gateway	Tidak pernah kedaluwarsa

Anda dapat mengubah periode retensi ini di CloudWatch konsol berdasarkan kebutuhan Anda.

Keandalan

Bagian ini menjelaskan bagaimana kami merancang solusi ini menggunakan prinsip dan praktik terbaik dari pilar keandalan.

- Solusinya menggunakan layanan tanpa server AWS sedapat mungkin (contoh: Lambda, API Gateway, Amazon S3, AWS Step Functions, Amazon DynamoDB, dan AWS Fargate) untuk memastikan ketersediaan dan pemulihan yang tinggi dari kegagalan layanan.
- Semua pemrosesan komputasi menggunakan fungsi Lambda atau Amazon ECS di AWS Fargate.
- Data disimpan di DynamoDB dan Amazon S3, sehingga tetap ada di beberapa Availability Zone secara default.

Efisiensi kinerja

Bagian ini menjelaskan bagaimana kami merancang solusi ini menggunakan prinsip dan praktik terbaik dari pilar efisiensi kinerja.

- Solusinya menggunakan arsitektur tanpa server dengan kemampuan untuk menskalakan secara horizontal sesuai kebutuhan.
- Solusi ini dapat diluncurkan di Wilayah mana pun yang mendukung layanan AWS dalam solusi ini, seperti: AWS Lambda, Amazon API Gateway, Amazon S3, AWS Step Functions, Amazon DynamoDB, Amazon ECS, AWS Fargate, dan Amazon Cognito.
- Solusi ini menggunakan layanan terkelola secara keseluruhan untuk mengurangi beban operasional penyediaan dan manajemen sumber daya.
- Solusi ini secara otomatis diuji dan diterapkan setiap hari untuk mencapai konsistensi seiring perubahan layanan AWS, serta ditinjau oleh arsitek solusi dan pakar materi pelajaran untuk area yang dapat dicoba dan ditingkatkan.

Optimalisasi biaya

Bagian ini menjelaskan bagaimana kami merancang solusi ini menggunakan prinsip dan praktik terbaik dari pilar pengoptimalan biaya.

- Solusinya menggunakan arsitektur tanpa server; oleh karena itu, pelanggan hanya dikenakan biaya untuk apa yang mereka gunakan.

- Amazon DynamoDB menskalakan kapasitas sesuai permintaan, jadi Anda hanya membayar untuk kapasitas yang Anda gunakan.
- AWS ECS di AWS Fargate memungkinkan Anda membayar hanya untuk sumber daya komputasi yang Anda gunakan, tanpa biaya di muka.
- AWS AgentCore Gateway berfungsi sebagai proxy berbasis Lambda yang hemat biaya untuk API pengujian beban terdistribusi, menghilangkan kebutuhan akan infrastruktur khusus dan mengurangi biaya melalui penetapan harga tanpa server. pay-per-request

Keberlanjutan

Bagian ini menjelaskan bagaimana kami merancang solusi ini menggunakan prinsip dan praktik terbaik pilar keberlanjutan.

- Solusi ini menggunakan layanan tanpa server terkelola untuk meminimalkan dampak lingkungan dari layanan backend dibandingkan dengan layanan lokal yang terus beroperasi.
- Layanan tanpa server memungkinkan Anda untuk meningkatkan atau menurunkan skala sesuai kebutuhan.

Detail arsitektur

Bagian ini menjelaskan komponen dan layanan AWS yang membentuk solusi ini dan detail arsitektur tentang cara komponen ini bekerja sama.

Solusi Pengujian Beban Terdistribusi pada AWS terdiri dari tiga komponen tingkat tinggi: ujung depan, backend, dan Server MCP opsional.

Ujung depan

Ujung depan menyediakan antarmuka untuk berinteraksi dengan solusi dan mencakup:

- API pengujian beban untuk akses terprogram
- Konsol web untuk membuat, menjadwalkan, dan menjalankan tes kinerja
- Server MCP opsional untuk analisis hasil tes dan kesalahan yang dibantu AI

Muat pengujian API

Pengujian Beban Terdistribusi di AWS mengonfigurasi Amazon API Gateway untuk meng-host RESTful API solusi. Pengguna dapat berinteraksi dengan sistem pengujian beban secara aman melalui konsol web, RESTful API, dan Server MCP opsional yang disertakan. API bertindak sebagai “pintu depan” untuk akses ke data pengujian yang disimpan di Amazon DynamoDB. Anda juga dapat menggunakan APIs untuk mengakses fungsionalitas tambahan apa pun yang Anda buat ke dalam solusi.

Solusi ini memanfaatkan fitur otentikasi pengguna kumpulan pengguna Amazon Cognito. Setelah berhasil mengautentikasi pengguna, Amazon Cognito mengeluarkan token web JSON yang digunakan untuk mengizinkan konsol mengirimkan permintaan ke solusi (titik akhir Amazon API APIs Gateway). Permintaan HTTPS dikirim oleh konsol ke APIs header otorisasi yang menyertakan token.

Berdasarkan permintaan tersebut, API Gateway memanggil fungsi AWS Lambda yang sesuai untuk melakukan tugas yang diperlukan pada data yang disimpan dalam tabel DynamoDB, menyimpan skenario pengujian sebagai objek JSON di Amazon S3, mengambil gambar metrik CloudWatch Amazon, dan mengirimkan skenario pengujian ke mesin status AWS Step Functions.

Untuk informasi selengkapnya tentang API solusi, lihat bagian API pengujian beban terdistribusi dari panduan ini.

Konsol web

Solusi ini mencakup konsol web yang dapat Anda gunakan untuk mengonfigurasi dan menjalankan pengujian, memantau pengujian yang sedang berjalan, dan melihat hasil pengujian terperinci. Konsol adalah aplikasi ReactJS yang dibangun dengan Cloudscape, sistem desain sumber terbuka untuk membangun aplikasi web yang intuitif. Konsol ini di-host di Amazon S3 dan diakses melalui Amazon. CloudFront Aplikasi ini memanfaatkan AWS Amplify untuk berintegrasi dengan Amazon Cognito untuk mengautentikasi pengguna. Konsol web juga berisi opsi untuk melihat data langsung untuk pengujian yang sedang berjalan, di mana ia berlangganan topik yang sesuai di AWS IoT Core.

URL konsol web adalah nama domain CloudFront distribusi yang dapat ditemukan di CloudFormation output sebagai Konsol. Setelah Anda meluncurkan CloudFormation template, Anda juga akan menerima email yang berisi URL konsol web dan kata sandi satu kali untuk masuk ke dalamnya.

MCP Server (Opsional)

Server Model Context Protocol (MCP) opsional menyediakan antarmuka tambahan untuk alat pengembangan AI untuk mengakses dan menganalisis data pengujian beban melalui interaksi bahasa alami. Komponen ini hanya digunakan jika Anda memilih opsi MCP Server selama penerapan solusi.

Server MCP memungkinkan agen AI untuk menanyakan hasil pengujian, menganalisis metrik kinerja, dan mendapatkan wawasan tentang data pengujian beban Anda menggunakan alat seperti Amazon Q, Claude, dan asisten AI lain yang kompatibel dengan MCP. Untuk informasi rinci tentang arsitektur dan konfigurasi MCP Server, lihat MCP Server di bagian ini.

Backend

Backend terdiri dari pipa gambar kontainer dan mesin pengujian beban yang Anda gunakan untuk menghasilkan beban untuk pengujian. Anda berinteraksi dengan backend melalui ujung depan. Selain itu, Amazon ECS pada tugas AWS Fargate yang diluncurkan untuk setiap pengujian ditandai dengan pengenal pengujian (ID) unik. Tag ID uji ini dapat digunakan untuk membantu Anda memantau biaya untuk solusi ini. Untuk informasi tambahan, lihat Tag Alokasi Biaya yang Ditentukan Pengguna di Panduan Pengguna AWS Billing and Cost Management.

Pipa gambar kontainer

Solusi ini menggunakan gambar kontainer yang dibangun dengan Amazon Linux 2023 sebagai gambar dasar dengan kerangka pengujian beban Taurus diinstal. Taurus adalah kerangka kerja

otomatisasi pengujian open-source yang mendukung JMeter, K6, Locust, dan alat pengujian lainnya. AWS menghosting gambar ini di repositori publik Amazon Elastic Container Registry (Amazon ECR). Solusinya menggunakan gambar ini untuk menjalankan tugas di Amazon ECS di klaster AWS Fargate.

Untuk informasi selengkapnya, lihat bagian kustomisasi gambar Container dari panduan ini.

Infrastruktur pengujian

Selain CloudFormation templat utama, solusinya menyediakan templat regional untuk meluncurkan sumber daya yang diperlukan untuk menjalankan pengujian di beberapa Wilayah. Solusinya menyimpan template ini di Amazon S3 dan menyediakan tautan ke sana di konsol web. Setiap tumpukan regional mencakup VPC, kluster AWS Fargate, dan fungsi Lambda untuk memproses data langsung.

Untuk informasi selengkapnya tentang cara menerapkan infrastruktur pengujian di Wilayah tambahan, lihat bagian penyebaran Multi-Region dari panduan ini.

Mesin uji beban

Solusi Pengujian Beban Terdistribusi menggunakan Amazon Elastic Container Service (Amazon ECS) Container Service (Amazon ECS) dan AWS Fargate untuk mensimulasikan ribuan pengguna bersamaan di beberapa Wilayah, menghasilkan permintaan HTTP dengan kecepatan berkelanjutan.

Anda menentukan parameter pengujian menggunakan konsol web yang disertakan. Solusinya menggunakan parameter ini untuk menghasilkan skenario pengujian JSON dan menyimpannya di Amazon S3. Untuk informasi selengkapnya tentang skrip pengujian dan parameter pengujian, lihat Jenis pengujian di bagian ini.

Mesin status AWS Step Functions menjalankan dan memantau tugas Amazon ECS di klaster AWS Fargate. Mesin status AWS Step Functions mencakup fungsi AWS Lambda ecr-checker, fungsi AWS Lambda, fungsi AWS Lambda yang menjalankan tugas, fungsi task-status-checker AWS Lambda pembatal tugas, dan fungsi AWS Lambda parser hasil. Untuk informasi selengkapnya tentang alur kerja, lihat bagian Alur kerja Uji panduan ini. Untuk informasi lebih lanjut tentang hasil tes, lihat bagian Hasil tes dari panduan ini. Untuk informasi selengkapnya tentang alur kerja pembatalan pengujian, lihat bagian Alur kerja pembatalan pengujian pada panduan ini.

Jika Anda memilih data langsung, solusi akan memulai fungsi real-time-data-publisher Lambda di setiap Wilayah dengan CloudWatch log yang sesuai dengan tugas Fargate di Wilayah tersebut.

Solusinya kemudian memproses dan menerbitkan data ke topik di AWS IoT Core dalam Wilayah tempat Anda meluncurkan tumpukan utama. Untuk informasi selengkapnya, lihat bagian Data langsung dari panduan ini.

MCP Server

Integrasi Server Model Context Protocol (MCP) opsional memungkinkan agen AI mengakses dan menganalisis data pengujian beban Anda secara terprogram melalui interaksi bahasa alami. Komponen ini hanya digunakan jika Anda memilih opsi MCP Server selama penerapan solusi.

MCP Server bertindak sebagai jembatan antara alat pengembangan AI dan penyebaran DLT Anda, menyediakan antarmuka standar untuk analisis cerdas hasil pengujian kinerja. Arsitektur ini mengintegrasikan beberapa layanan AWS untuk menciptakan antarmuka yang aman dan dapat diskalakan untuk interaksi agen AI:

AgentCore Gerbang AWS

AWS AgentCore Gateway adalah layanan yang dikelola sepenuhnya yang menyediakan hosting standar dan manajemen protokol untuk server MCP. Dalam solusi ini, AgentCore Gateway berfungsi sebagai titik akhir publik yang terhubung dengan agen AI saat meminta akses ke data pengujian beban Anda.

Layanan ini menangani semua komunikasi protokol MCP, termasuk penemuan alat, validasi token otentikasi, dan perutean permintaan. AgentCore Gateway beroperasi sebagai layanan multi-penyewa dengan perlindungan keamanan bawaan terhadap ancaman umum terhadap titik akhir publik, sambil memvalidasi tanda tangan token Cognito dan klaim untuk setiap permintaan.

Lambda Server MCP DLT

Fungsi DLT MCP Server Lambda adalah komponen tanpa server khusus yang memproses permintaan MCP dari agen AI dan menerjemahkannya ke dalam kueri terhadap sumber daya DLT Anda.

Fungsi Lambda ini bertindak sebagai lapisan intelijen integrasi MCP, mengambil hasil pengujian dari tabel DynamoDB, mengakses artefak kinerja yang disimpan dalam bucket S3, dan menanyakan log untuk informasi eksekusi terperinci. CloudWatch Fungsi Lambda mengimplementasikan pola akses hanya-baca dan mengubah data DLT mentah menjadi format terstruktur dan ramah AI yang dapat dengan mudah ditafsirkan dan dianalisis oleh agen.

Integrasi otentikasi

Sistem otentikasi memanfaatkan infrastruktur kumpulan pengguna Cognito yang ada untuk mempertahankan kontrol akses yang konsisten di konsol web dan antarmuka Server MCP.

Integrasi ini menggunakan OAuth otentikasi berbasis token 2.0. Pengguna mengautentikasi sekali melalui proses login Cognito dan menerima token yang berfungsi untuk interaksi UI dan akses Server MCP. Sistem mempertahankan batas izin dan kontrol akses yang sama dengan antarmuka web, memastikan bahwa pengguna hanya dapat mengakses melalui agen AI data pengujian beban yang sama yang dapat mereka akses melalui konsol.

Layanan AWS dalam solusi ini

Layanan AWS berikut disertakan dalam solusi ini:

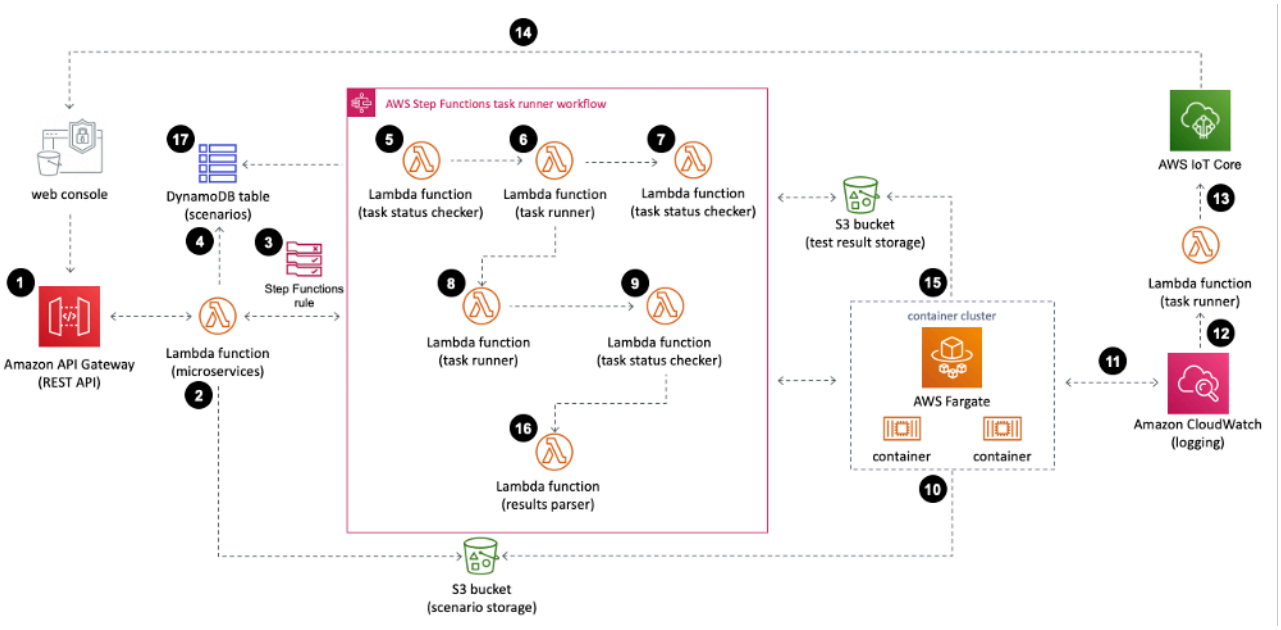
AWS service	Deskripsi
Amazon API Gateway	Inti. Menghosting titik akhir REST API dalam solusi.
AWS CloudFormation	Inti. Mengelola penyebaran untuk infrastruktur solusi.
Amazon CloudFront	Inti. Melayani konten web yang dihosting di Amazon S3.
Amazon CloudWatch	Inti. Menyimpan log dan metrik solusi.
Amazon Cognito	Inti. Menangani manajemen pengguna dan otentikasi untuk API.
Amazon DynamoDB	Inti. Menyimpan informasi penyebaran dan menguji detail skenario dan hasil.
Layanan Kontainer Elastis Amazon	Inti. Menerapkan dan mengelola tugas Amazon ECS independen di kontainer AWS Fargate.
AWS Fargate	Inti. Wadah Amazon ECS solusi host
AWS Identity and Access Management	Inti. Menangani peran pengguna dan manajemen izin.
AWS Lambda	Inti. Menyediakan logika untuk APIs implementasi, menguji hasil parsing, dan meluncurkan workers/leader tugas.

AWS service	Deskripsi
AWS Step Functions	Inti. Mengatur penyediaan kontainer Amazon ECS pada tugas AWS Fargate di wilayah yang ditentukan
AWS Amplify	Mendukung. Menyediakan konsol web yang didukung oleh AWS Amplify.
CloudWatch Acara Amazon	Mendukung. Menjadwalkan tes untuk secara otomatis dimulai pada tanggal tertentu atau pada tanggal berulang.
Amazon Elastic Container Registry	Mendukung. Menghosting gambar kontainer di repositori ECR publik.
AWS IoT Core	Mendukung. Memungkinkan melihat data langsung untuk pengujian yang sedang berjalan dengan berlangganan topik terkait di AWS IoT Core.
AWS Systems Manager	Mendukung. Menyediakan pemantauan sumber daya tingkat aplikasi dan visualisasi operasi sumber daya dan data biaya.
Amazon S3	Mendukung. Menghosting konten web statis, log, metrik, dan data pengujian.
Amazon Virtual Private Cloud	Mendukung. Berisi wadah Amazon ECS solusi yang berjalan di AWS Fargate.
Amazon Bedrock AgentCore	Mendukung, Opsional. Menghosting Server Remote Model Context Protocol (MCP) opsional solusi untuk integrasi agen AI dengan API.

Cara Kerja Pengujian Beban Terdistribusi di AWS

Rincian rinci berikut menunjukkan langkah-langkah yang terlibat dalam menjalankan skenario pengujian.

Alur kerja uji



1. Anda menggunakan konsol web untuk mengirimkan skenario pengujian yang menyertakan detail konfigurasi ke API solusi.
2. Konfigurasi skenario pengujian diunggah ke Amazon Simple Storage Service (Amazon S3) sebagai file JSON (). `s3://<bucket-name>/test-scenarios/<$TEST_ID>/<$TEST_ID>.json`
3. Mesin status AWS Step Functions berjalan menggunakan ID pengujian, jumlah tugas, jenis pengujian, dan jenis file sebagai input mesin status AWS Step Functions. Jika pengujian dijadwalkan, pengujian akan membuat aturan CloudWatch Acara terlebih dahulu, yang memicu AWS Step Functions pada tanggal yang ditentukan. Untuk detail selengkapnya tentang alur kerja penjadwalan, lihat bagian Alur kerja penjadwalan pengujian pada panduan ini.
4. Detail konfigurasi disimpan dalam tabel skenario Amazon DynamoDB.
5. Dalam alur kerja runner tugas AWS Step Functions, fungsi AWS task-status-checker Lambda memeriksa apakah tugas Amazon Elastic Container Service (Amazon ECS) Container Service (Amazon ECS) sudah berjalan untuk ID pengujian yang sama. Jika tugas dengan ID pengujian yang sama ditemukan berjalan, itu menyebabkan kesalahan. Jika tidak ada tugas Amazon ECS yang berjalan di klaster AWS Fargate, fungsi akan mengembalikan ID pengujian, jumlah tugas, dan jenis pengujian.
6. Fungsi AWS Lambda pelari tugas mendapatkan detail tugas dari langkah sebelumnya dan menjalankan tugas pekerja Amazon ECS di klaster AWS Fargate. Amazon ECS API menggunakan RunTask tindakan untuk menjalankan tugas pekerja. Tugas pekerja ini diluncurkan dan kemudian menunggu pesan awal dari tugas pemimpin untuk memulai tes. RunTask Tindakan

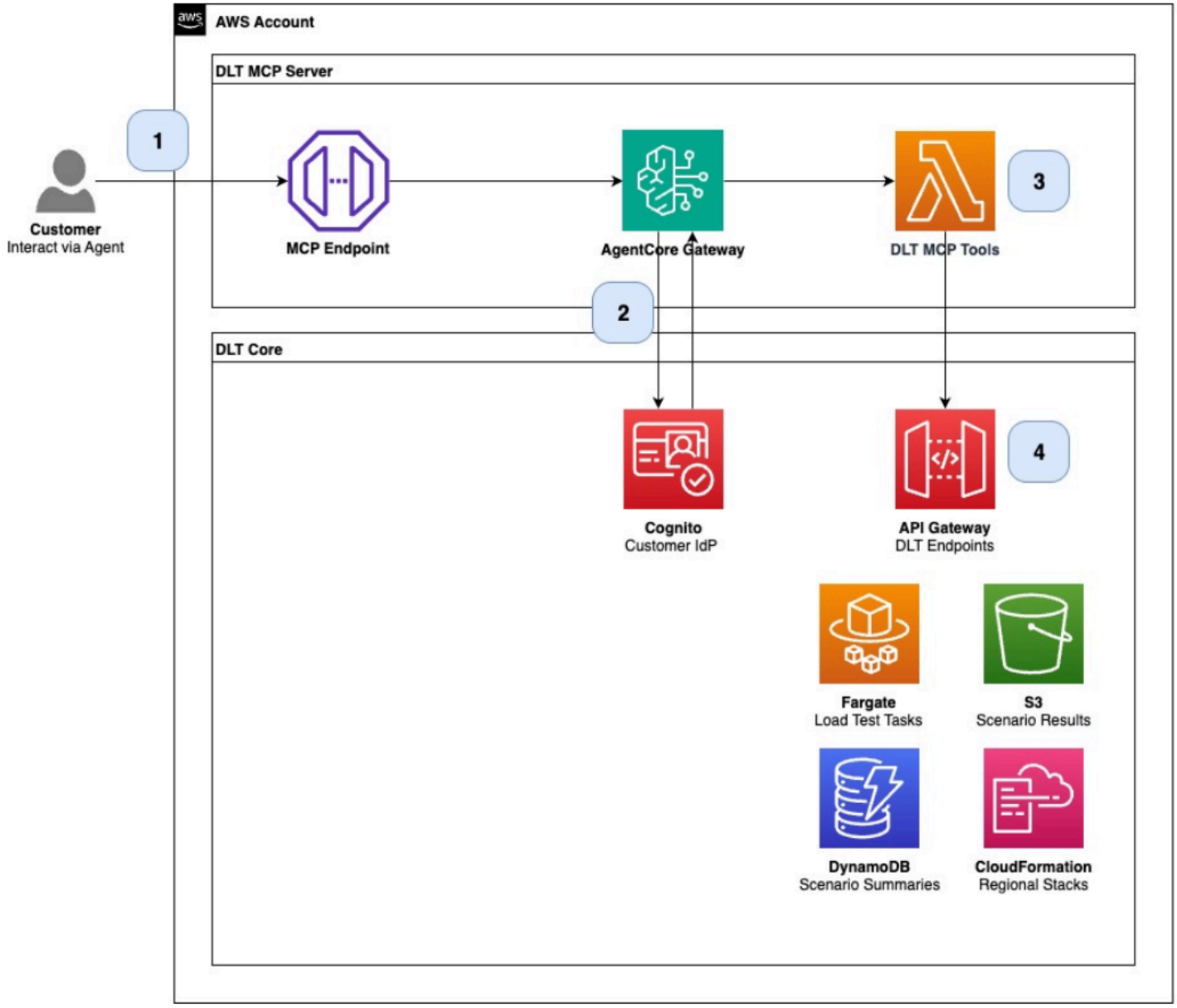
ini dibatasi hingga 10 tugas per definisi. Jika jumlah tugas Anda lebih dari 10, definisi tugas akan berjalan beberapa kali hingga semua tugas pekerja dimulai. Fungsi ini juga menghasilkan awalan untuk membedakan pengujian saat ini dalam fungsi AWS Lambda yang mengurai hasil.

7. Fungsi task-status-checker AWS Lambda memeriksa apakah semua tugas pekerja Amazon ECS berjalan dengan ID pengujian yang sama. Jika tugas masih disediakan, ia menunggu selama satu menit dan memeriksa lagi. Setelah semua tugas Amazon ECS berjalan, ia mengembalikan ID pengujian, jumlah tugas, jenis pengujian, semua tugas IDs dan awalan dan meneruskannya ke fungsi task-runner.
8. Fungsi AWS Lambda pelari tugas berjalan lagi, kali ini meluncurkan satu tugas Amazon ECS untuk bertindak sebagai node pemimpin. Tugas ECS ini mengirimkan pesan uji awal ke setiap tugas pekerja untuk memulai pengujian secara bersamaan.
9. Fungsi task-status-checker AWS Lambda kembali memeriksa apakah tugas Amazon ECS berjalan dengan ID pengujian yang sama. Jika tugas masih berjalan, ia menunggu selama satu menit dan memeriksa lagi. Setelah tidak ada tugas Amazon ECS yang berjalan, ia mengembalikan ID pengujian, jumlah tugas, jenis pengujian, dan awalan.
10. Saat fungsi AWS Lambda pelari tugas menjalankan tugas Amazon ECS di klaster AWS Fargate, setiap tugas mengunduh konfigurasi pengujian dari Amazon S3 dan memulai pengujian.
11. Setelah pengujian berjalan, waktu respons rata-rata, jumlah pengguna bersamaan, jumlah permintaan yang berhasil, dan jumlah permintaan yang gagal untuk setiap tugas dicatat di Amazon CloudWatch dan dapat dilihat di CloudWatch dasbor.
12. Jika Anda menyertakan data langsung dalam pengujian, solusi akan memfilter hasil pengujian real-time CloudWatch menggunakan filter langganan. Kemudian solusinya meneruskan data ke fungsi Lambda.
13. Fungsi Lambda kemudian menyusun data yang diterima dan menerbitkannya ke topik AWS IoT Core.
14. Konsol web berlangganan topik AWS IoT Core untuk pengujian dan menerima data yang dipublikasikan ke topik untuk membuat grafik data waktu nyata saat pengujian sedang berjalan.
15. Saat pengujian selesai, gambar kontainer mengekspor laporan terperinci sebagai file XHTML ke Amazon S3. Setiap file diberi UUID untuk nama file. Misalnya, s3://dlte-bucket/test-scenarios/ **<** $TEST_ID**>** /results/ **<**$UUID**>** .json.
16. Saat file XHTML diunggah ke Amazon S3, fungsi AWS Lambda parser hasil membaca hasil dalam file XMLyang dimulai dengan awalan dan mem-parsing dan menggabungkan semua hasil menjadi satu hasil yang diringkas.
17. Fungsi AWS Lambda parser hasil menulis hasil agregat ke tabel Amazon DynamoDB.

Alur kerja MCP Server (Opsional)

Jika Anda menerapkan integrasi MCP Server opsional, agen AI dapat mengakses dan menganalisis data pengujian beban Anda melalui alur kerja berikut:

Arsitektur MCP Server



1. Interaksi pelanggan - Pelanggan berinteraksi dengan MCP DLT melalui MCP Endpoint yang diselenggarakan oleh AWS Gateway. AgentCore Agen AI terhubung ke titik akhir ini untuk meminta akses memuat data pengujian.

2. Otorisasi - AgentCore Gateway menangani otorisasi terhadap klien aplikasi kumpulan pengguna Cognito Solusi. Gateway memvalidasi token Cognito pengguna untuk memastikan mereka memiliki izin untuk mengakses server DLT MCP. Pengguna yang berwenang diberikan akses dengan akses alat agen terbatas pada operasi hanya-baca.
3. Spesifikasi alat - AgentCore Gateway terhubung ke fungsi Lambda Server MCP DLT. Spesifikasi alat mendefinisikan alat yang tersedia yang dapat digunakan agen AI untuk berinteraksi dengan data pengujian beban Anda.
4. Akses API hanya-baca - Fungsi Lambda dicakup untuk akses API hanya-baca melalui titik akhir DLT API Gateway yang ada. Fungsi ini menyediakan empat operasi utama:
   - Daftar skenario - Mengambil daftar skenario pengujian dari tabel skenario DynamoDB
   - Dapatkan hasil tes skenario - Akses hasil pengujian terperinci untuk skenario tertentu dari DynamoDB dan S3
   - Dapatkan pelari uji beban Fargate - Kueri informasi tentang menjalankan tugas Fargate di cluster ECS
   - Dapatkan tumpukan Regional yang tersedia - Ambil informasi tentang infrastruktur regional yang digunakan dari CloudFormation

Integrasi MCP Server memanfaatkan infrastruktur DLT yang ada (API Gateway, Cognito, DynamoDB, S3) untuk menyediakan akses yang aman dan hanya-baca untuk menguji data untuk analisis dan wawasan yang didukung AI.

Pertimbangan desain

Bagian ini menjelaskan keputusan desain penting dan opsi konfigurasi untuk solusi Pengujian Beban Terdistribusi pada AWS, termasuk aplikasi yang didukung, jenis pengujian, opsi penjadwalan, dan pertimbangan penerapan.

Aplikasi-aplikasi yang didukung

Solusi ini mendukung pengujian aplikasi berbasis cloud dan aplikasi lokal selama Anda memiliki konektivitas jaringan dari akun AWS ke aplikasi Anda. Solusinya mendukung APIs yang menggunakan protokol HTTP atau HTTPS.

Jenis tes

Pengujian Beban Terdistribusi di AWS mendukung beberapa jenis pengujian: pengujian titik akhir HTTP sederhana,, K6 JMeter, dan Locust.

Tes titik akhir HTTP sederhana

Konsol web menyediakan antarmuka Konfigurasi Titik Akhir HTTP yang memungkinkan Anda menguji titik akhir HTTP atau HTTPS apa pun tanpa menulis skrip khusus. Anda menentukan URL endpoint, pilih metode HTTP (GET, POST, PUT, DELETE, dll.) Dari menu dropdown, dan secara opsional menambahkan header permintaan kustom dan muatan tubuh. Konfigurasi ini memungkinkan Anda untuk menguji APIs dengan token otorisasi kustom, jenis konten, atau header HTTP lainnya dan badan permintaan yang diperlukan oleh aplikasi Anda.

JMeter tes

Saat membuat skenario pengujian menggunakan konsol web, Anda dapat mengunggah skrip JMeter pengujian. Solusinya mengunggah skrip ke bucket skenario S3. Saat tugas Amazon ECS berjalan, mereka mengunduh JMeter skrip dari S3 dan menjalankan pengujian.

> **Important**
>
> Meskipun JMeter skrip Anda dapat menentukan konkurensi (pengguna virtual), tingkat transaksi (TPS), waktu peningkatan, dan parameter pemuatan lainnya, solusinya akan mengganti konfigurasi ini dengan nilai yang Anda tentukan di layar Bentuk Lalu Lintas selama pembuatan pengujian. Konfigurasi Bentuk Lalu Lintas mengontrol jumlah tugas, konkurensi (pengguna virtual per tugas), durasi peningkatan, dan durasi penahanan untuk eksekusi pengujian.

Jika Anda memiliki file JMeter input, Anda dapat zip file input bersama dengan JMeter skrip. Anda dapat memilih file zip saat membuat skenario pengujian.

Jika Anda ingin menyertakan plugin, file.jar apa pun yang disertakan dalam subdirektori /plugins dalam file zip yang dibundel akan disalin ke direktori JMeter ekstensi dan tersedia untuk pengujian beban.

> **Note**
>
> Jika Anda menyertakan file JMeter input dengan file JMeter skrip Anda, Anda harus menyertakan jalur relatif dari file input dalam file JMeter skrip Anda. Selain itu, file input harus berada di jalur relatif. Misalnya, ketika file JMeter input dan file skrip Anda berada di/home/user directory and you refer to the input files in the JMeter script file, the path of input files must be ./INPUT_FILES. If you use /home/user/INPUT_FILES sebagai gantinya, pengujian akan gagal karena tidak akan dapat menemukan file input.

Jika Anda menyertakan JMeter plugin, file.jar harus dibundel dalam subdirektori bernama /plugins dalam root file zip. Sehubungan dengan root file zip, jalur ke file jar harus. /plugins/bundled_plugin.jar.

Untuk informasi selengkapnya tentang cara menggunakan JMeter skrip, lihat Panduan JMeter Pengguna.

Tes K6

Solusinya mendukung pengujian berbasis kerangka kerja K6. K6 dirilis di bawah lisensi AGPL-3.0. Solusinya menampilkan pesan pengakuan lisensi saat membuat tes K6 baru. Anda dapat mengunggah file uji K6 bersama dengan file input yang diperlukan dalam file arsip.

> **Important**
>
> Meskipun skrip K6 Anda dapat menentukan konkurensi (pengguna virtual), tahapan, ambang batas, dan parameter beban lainnya, solusinya akan mengganti konfigurasi ini dengan nilai yang Anda tentukan di Bentuk Lalu Lintas layar selama pembuatan pengujian. Konfigurasi Bentuk Lalu Lintas mengontrol jumlah tugas, konkurensi (pengguna virtual per tugas), durasi peningkatan, dan durasi penahanan untuk eksekusi pengujian.

Tes belalang

Solusinya mendukung pengujian berbasis kerangka kerja Locust. Anda dapat mengunggah file uji Locust bersama dengan file input yang diperlukan dalam file arsip.

> **Important**
>
> Meskipun skrip Locust Anda dapat menentukan konkurensi (jumlah pengguna), laju spawn, dan parameter pemuatan lainnya, solusinya akan mengganti konfigurasi ini dengan nilai yang Anda tentukan di Layar Bentuk Lalu Lintas selama pembuatan pengujian. Konfigurasi Bentuk Lalu Lintas mengontrol jumlah tugas, konkurensi (pengguna virtual per tugas), durasi peningkatan, dan durasi penahanan untuk eksekusi pengujian.

Tes penjadwalan

Solusinya menyediakan tiga opsi waktu eksekusi untuk menjalankan tes beban:

- Jalankan Sekarang - Jalankan uji beban segera setelah pembuatan
- Jalankan Once - Jalankan tes pada tanggal dan waktu tertentu di masa depan
- Jalankan pada Jadwal - Buat tes berulang menggunakan ekspresi cron untuk menentukan jadwal

Ketika Anda memilih Run Once, Anda menentukan waktu berjalan dalam format 24 jam dan tanggal berjalan ketika uji beban harus mulai berjalan.

Ketika Anda memilih Run on a Schedule, Anda dapat memasukkan ekspresi cron secara manual atau memilih dari pola cron umum (seperti setiap jam, setiap hari pada waktu tertentu, hari kerja, atau bulanan). Ekspresi cron menggunakan format jadwal berbutir halus dengan bidang untuk menit, jam, hari bulan, bulan, hari dalam seminggu, dan tahun. Anda juga harus menentukan tanggal kedaluwarsa, yang menentukan kapan tes terjadwal harus berhenti berjalan. Untuk informasi selengkapnya tentang cara kerja penjadwalan, lihat bagian Alur kerja penjadwalan pengujian pada panduan ini.

> **Note**
>
> - Durasi tes: Pertimbangkan total durasi tes saat menjadwalkan. Misalnya, tes dengan waktu ramp-up 10 menit dan waktu penahanan 40 menit akan memakan waktu sekitar 80 menit untuk menyelesaikannya.
> - Interval minimum: Pastikan interval antara tes terjadwal lebih lama dari perkiraan durasi tes. Misalnya, jika tes memakan waktu sekitar 80 menit, jadwalkan untuk berjalan tidak lebih sering dari setiap 3 jam.

- Batasan per jam: Sistem tidak mengizinkan tes dijadwalkan dengan perbedaan hanya satu jam meskipun perkiraan durasi tes kurang dari satu jam.

Tes bersamaan

Solusi ini membuat CloudWatch dasbor Amazon untuk setiap pengujian yang menampilkan output gabungan dari semua tugas yang berjalan di cluster Amazon ECS secara real time. CloudWatch Dasbor menunjukkan waktu respons rata-rata, jumlah pengguna bersamaan, jumlah permintaan yang berhasil, dan jumlah permintaan yang gagal. Solusinya menggabungkan setiap metrik per detik dan memperbarui dasbor setiap menit.

Manajemen pengguna

Selama konfigurasi awal, Anda memberikan nama pengguna dan alamat email yang digunakan Amazon Cognito untuk memberi Anda akses ke konsol web solusi. Konsol tidak menyediakan administrasi pengguna. Untuk menambahkan pengguna tambahan, Anda harus menggunakan konsol Amazon Cognito. Untuk informasi selengkapnya, lihat Mengelola Pengguna di Kumpulan Pengguna di Panduan Pengembang Amazon Cognito.

Untuk memigrasikan pengguna yang ada ke kumpulan pengguna Amazon Cognito, lihat Pendekatan blog AWS untuk memigrasikan pengguna ke kumpulan pengguna Amazon Cognito.

Penyebaran regional

Solusi ini menggunakan Amazon Cognito yang hanya tersedia di Wilayah AWS tertentu. Oleh karena itu, Anda harus menerapkan solusi ini di wilayah tempat Amazon Cognito tersedia. Untuk ketersediaan layanan terbaru menurut Wilayah, lihat Daftar Layanan Regional AWS.

Rencanakan penyebaran Anda

Bagian ini menjelaskan biaya, keamanan, Wilayah yang didukung, kuota, dan pertimbangan lain yang harus Anda tinjau sebelum menerapkan solusi.

Biaya

Anda bertanggung jawab atas biaya layanan AWS yang digunakan saat menjalankan solusi ini. Total biaya tergantung pada jumlah uji beban yang dijalankan, durasi pengujian tersebut, dan jumlah data yang dihasilkan. Pada revisi ini, perkiraan biaya untuk menjalankan solusi ini dengan pengaturan default di Wilayah AS Timur (Virginia N.) adalah sekitar $30,90 per bulan.

Tabel berikut memberikan rincian biaya sampel untuk menerapkan solusi ini dengan parameter default di Wilayah AS Timur (Virginia N.) selama satu bulan.

AWS service	Dimensi	Biaya [USD]
AWS Fargate	10 tugas sesuai permintaan (menggunakan dua memori v CPUs dan 4 GB) berjalan selama 30 jam	$29,62
Amazon DynamoDB	1.000 unit kapasitas tulis sesuai permintaan 1.000 unit kapasitas baca sesuai permintaan	$0.0015
AWS Lambda	1.000 permintaan Total durasi 10 menit	$1,25
AWS Step Functions	1.000 transisi negara	$0,025
Jumlah:		$30,90 per bulan

Sumber daya solusi ditandai dengan Key= SolutionId dan Value=. SO0062 Anda dapat mengaktifkan kunci tag SolutionId dengan mengikuti tag pengaktifan dokumentasi. Setelah tag diaktifkan, Anda

dapat membuat aturan kategori biaya dengan mengikuti dokumentasi untuk membuat kategori biaya. Anda dapat melihat biaya yang dikeluarkan untuk solusi dengan memantau konsol kategori biaya dan memilih nama kategori biaya.

Sebaiknya buat anggaran melalui AWS Cost Explorer untuk membantu mengelola biaya. Harga dapat berubah sewaktu-waktu. Untuk detail selengkapnya, lihat halaman web harga untuk setiap layanan AWS yang digunakan dalam solusi ini.

> **Note**
>
> Konfigurasi tugas default menggunakan 2 v CPUs dan 4 GB memori per tugas. Jika tes beban Anda tidak memerlukan sumber daya ini, Anda dapat menguranginya untuk menurunkan biaya. Sebaliknya, Anda dapat meningkatkan sumber daya untuk mendukung konkurensi yang lebih tinggi per tugas. Untuk informasi selengkapnya, lihat bagian Meningkatkan sumber daya kontainer dalam panduan ini.

> **Note**
>
> Solusi ini menyediakan opsi untuk menyertakan data langsung saat menjalankan pengujian. Fitur ini memerlukan fungsi AWS Lambda tambahan dan topik AWS IoT Core yang menimbulkan biaya tambahan.

Biaya tambahan MCP Server (Opsional)

Tabel berikut memberikan rincian biaya untuk integrasi MCP Server dengan harga di Wilayah AS Timur (Virginia N.) selama satu bulan.

Komponen layanan	Dimensi	Biaya [USD]
AgentCore Gateway - Pengindeksan Alat	10 alat × $0,02 per 100 alat	$0.002
AgentCore Gateway - Cari API	10,000 interaksi × $0.025 per 1,000	$0,25

Komponen layanan	Dimensi	Biaya [USD]
AgentCore Gateway - Pemanggilan API	50.000 doa × $0,005 per 1.000	$0,25
Fungsi AWS Lambda	Variabel berdasarkan penggunaan (beban kerja tipikal)	$5,00 - US$20,00
Total perkiraan biaya tambahan:		$5,50 - $20,50 per bulan

Harga dapat berubah sewaktu-waktu. Untuk detail selengkapnya tentang harga AgentCore Gateway, lihat Harga Amazon Bedrock (bagian AgentCore Gateway). Untuk harga Lambda, lihat Harga AWS Lambda.

Keamanan

Saat Anda membangun sistem pada infrastruktur AWS, tanggung jawab keamanan dibagi antara Anda dan AWS. Model tanggung jawab bersama ini mengurangi beban operasional Anda karena AWS mengoperasikan, mengelola, dan mengontrol komponen termasuk sistem operasi host, lapisan virtualisasi, dan keamanan fisik fasilitas tempat layanan beroperasi. Untuk informasi selengkapnya tentang keamanan AWS, kunjungi AWS Cloud Security.

Peran IAM

Peran AWS Identity and Access Management (IAM) memungkinkan pelanggan menetapkan kebijakan akses terperinci dan izin untuk layanan dan pengguna di AWS Cloud. Solusi ini menciptakan peran IAM yang memberikan akses fungsi AWS Lambda solusi untuk membuat sumber daya Regional.

Amazon CloudFront

Solusi ini menerapkan UI web yang dihosting di bucket Amazon S3, yang didistribusikan oleh Amazon. CloudFront Untuk membantu mengurangi latensi dan meningkatkan keamanan, solusi ini mencakup CloudFront distribusi dengan identitas akses asal, yaitu CloudFront pengguna yang menyediakan akses publik ke konten bucket situs web solusi. Secara default, CloudFront distribusi menggunakan TLS 1.2 untuk menegakkan protokol keamanan tingkat tertinggi. Untuk informasi

selengkapnya, lihat Membatasi akses ke asal Amazon S3 di Panduan Pengembang CloudFront Amazon.

CloudFront mengaktifkan mitigasi keamanan tambahan untuk menambahkan header keamanan HTTP ke setiap respons penampil. Untuk informasi selengkapnya, lihat Menambahkan atau menghapus header HTTP dalam CloudFront tanggapan.

Solusi ini menggunakan CloudFront sertifikat default, yang memiliki protokol keamanan minimum yang didukung TLS v1.0. Untuk menegakkan penggunaan TLS v1.2 atau TLS v1.3, Anda harus menggunakan sertifikat SSL kustom alih-alih sertifikat default. CloudFront Untuk informasi selengkapnya, lihat Bagaimana cara mengonfigurasi CloudFront distribusi saya untuk menggunakan SSL/TLS sertifikat.

Amazon API Gateway

Solusi ini menerapkan titik akhir Amazon API Gateway yang dioptimalkan RESTful APIs untuk menyediakan fungsionalitas pengujian beban menggunakan titik akhir API Gateway default, bukan domain kustom. Untuk dioptimalkan tepi APIs menggunakan titik akhir default, API Gateway menggunakan kebijakan keamanan TLS-1-0. Untuk informasi selengkapnya, lihat Bekerja dengan REST APIs di Panduan Pengembang Amazon API Gateway.

Solusi ini menggunakan sertifikat API Gateway default, yang memiliki protokol keamanan minimum yang didukung TLS v1.0. Untuk menerapkan penggunaan TLS v1.2 atau TLS v1.3, Anda harus menggunakan domain kustom dengan sertifikat SSL kustom, bukan sertifikat API Gateway default. Untuk informasi selengkapnya, lihat Menyiapkan nama domain khusus untuk REST APIs.

Grup keamanan AWS Fargate

Secara default, solusi ini membuka aturan keluar grup keamanan AWS Fargate kepada publik. Jika Anda ingin memblokir AWS Fargate agar tidak mengirim lalu lintas ke mana pun, ubah aturan keluar ke Perutean Antar-Domain Tanpa Kelas (CIDR) tertentu.

Grup keamanan ini juga menyertakan aturan masuk yang memungkinkan lalu lintas lokal di port 50.000 ke sumber apa pun yang termasuk dalam grup keamanan yang sama. Ini digunakan untuk memungkinkan wadah berkomunikasi satu sama lain.

Amazon VPC

VPC: Virtual Private Cloud (VPC) berbasis layanan Amazon VPC memberi Anda jaringan pribadi yang terisolasi secara logis di AWS Cloud.

Anda dapat menentukan VPC Anda sendiri dalam CloudFormation parameter AWS selama penerapan. VPC digunakan secara eksklusif oleh tugas ECS yang menghasilkan beban; konsol web dan API tidak digunakan dalam VPC ini. Jika Anda tidak menentukan VPC yang ada, solusinya akan membuat VPC baru dengan konfigurasi jaringan yang diperlukan. Jika Anda memilih untuk menggunakan VPC yang ada, VPC harus memenuhi persyaratan berikut untuk menjalankan tugas pengujian beban dengan sukses.

Persyaratan VPC

Persyaratan minimum untuk VPC yang akan digunakan dengan Pengujian Beban Terdistribusi di AWS tercantum di bawah ini.

- VPC harus berisi setidaknya dua AZs
- VPC harus berisi setidaknya dua subnet, masing-masing dalam AZ terpisah
- Subnet VPC dapat bersifat publik atau pribadi, tetapi mereka harus menggunakan konfigurasi yang sama (baik publik ATAU keduanya pribadi)
- VPC harus menyediakan akses ke titik akhir untuk ECR, CloudWatch Log, S3, dan IoT Core.
- VPC harus menyediakan akses ke layanan yang ditargetkan oleh uji beban.

> **Note**
>
> Jika Anda tidak memiliki VPC yang memenuhi kriteria ini, Anda dapat membuat VPC dengan wizard VPC dengan cepat. Untuk informasi selengkapnya, lihat Membuat VPC.

Subnet publik dapat memenuhi persyaratan ini dengan memasukkan yang berikut:

- Gateway internet yang terpasang pada VPC
- Rute ke gateway internet (0.0.0.0/0)

Subnet pribadi dapat memenuhi persyaratan ini melalui penggunaan Gateway NAT atau titik akhir VPC, seperti yang dijelaskan di bawah ini.

Opsi 1: NAT Gateway

- Terapkan NAT Gateway di setiap AZ dengan subnet pribadi
- Konfigurasikan tabel rute untuk merutekan lalu lintas ke internet (0.0.0.0/0) melalui NAT Gateway

Opsi 2: Titik Akhir VPC

Buat titik akhir VPC berikut di VPC Anda:

- Titik Akhir API Amazon ECR: `com.amazonaws.<region>.ecr.api`
- Titik Akhir Amazon ECR DKR: `com.amazonaws.<region>.ecr.dkr`
- Titik akhir CloudWatch Log Amazon: `com.amazonaws.<region>.logs`
- Titik akhir Gerbang Amazon S3: `com.amazonaws.<region>.s3`
- Titik akhir AWS IoT Core (diperlukan jika menggunakan bagan data langsung) `com.amazonaws.<region>.iot.data`

Konfigurasi VPC lainnya juga dapat berfungsi.

> **Important**
>
> Grup keamanan yang dilampirkan ke setiap antarmuka titik akhir VPC harus mengizinkan lalu lintas TCP masuk pada port 443 dari grup keamanan tugas ECS.

Konfigurasi Grup Keamanan

Selama penerapan, solusi akan membuat grup keamanan dalam VPC Anda untuk mengizinkan lalu lintas berikut dengan tugas di cluster ECS:

- Semua lalu lintas keluar
- Lalu lintas masuk di pelabuhan 50000 dari tugas lain dalam kelompok keamanan yang sama, untuk memfasilitasi koordinasi antara tugas pekerja dan pemimpin.

Tes stress jaringan

Anda bertanggung jawab untuk menggunakan solusi ini di bawah kebijakan Uji Stres Jaringan. Kebijakan ini mencakup situasi seperti saat Anda berencana menjalankan pengujian jaringan volume tinggi langsung dari instans Amazon EC2 ke lokasi lain seperti instans Amazon EC2 lainnya, properti/layanan AWS, atau titik akhir eksternal. Tes ini kadang-kadang disebut stress test, load test, atau gameday test. Sebagian besar pengujian pelanggan tidak akan termasuk dalam kebijakan ini; namun, lihat kebijakan ini jika Anda yakin Anda akan menghasilkan lalu lintas yang menopang,

secara agregat, selama lebih dari 1 menit, lebih dari 1 Gbps (1 miliar bit per detik) atau lebih dari 1 Gpps (1 miliar paket per detik).

Membatasi akses ke antarmuka pengguna publik

Untuk membatasi akses ke antarmuka pengguna yang menghadap publik di luar mekanisme autentikasi dan otorisasi yang disediakan oleh IAM dan Amazon Cognito, gunakan solusi Otomasi Keamanan AWS WAF (firewall aplikasi web).

Solusi ini secara otomatis menerapkan seperangkat aturan AWS WAF yang memfilter serangan berbasis web umum. Pengguna dapat memilih dari fitur pelindung yang telah dikonfigurasi sebelumnya yang menentukan aturan yang disertakan dalam daftar kontrol akses web AWS WAF (web ACL).

Keamanan MCP Server (Opsional)

Jika Anda menerapkan integrasi MCP Server opsional, solusinya menggunakan AWS AgentCore Gateway untuk menyediakan akses aman untuk memuat data pengujian untuk agen AI. AgentCore Gateway memvalidasi token otentikasi Amazon Cognito untuk setiap permintaan, memastikan bahwa hanya pengguna yang berwenang yang dapat mengakses Server MCP. Fungsi MCP Server Lambda mengimplementasikan pola akses hanya-baca, mencegah agen AI memodifikasi konfigurasi atau hasil pengujian. Semua interaksi MCP Server menggunakan batas izin dan kontrol akses yang sama seperti konsol web.

Wilayah AWS yang Didukung

Solusi ini menggunakan layanan Amazon Cognito, yang saat ini tidak tersedia di semua Wilayah AWS. Untuk ketersediaan terbaru layanan AWS menurut Wilayah, lihat Daftar Layanan Regional AWS.

Pengujian Beban Terdistribusi di AWS tersedia di Wilayah AWS berikut:

Nama wilayah	
AS Timur (Ohio)	Asia Pasifik (Tokyo)
AS Timur (Virginia Utara)	Kanada (Pusat)

Nama wilayah	
AS Barat (California Utara)	Eropa (Frankfurt)
AS Barat (Oregon)	Eropa (Irlandia)
Asia Pasifik (Mumbai)	Eropa (London)
Asia Pasifik (Seoul)	Eropa (Paris)
Asia Pasifik (Singapura)	Eropa (Stockholm)
Asia Pasifik (Sydney)	Amerika Selatan (Sao Paulo)

MCP Server mendukung AWS Regions (Opsional)

Jika Anda berencana untuk menerapkan integrasi MCP Server opsional, Anda harus menerapkan solusi di Wilayah AWS di mana AWS AgentCore Gateway tersedia. Fitur MCP Server hanya tersedia di Wilayah AWS berikut:

Nama wilayah	Kode Wilayah
US East (Northern Virginia)	us-east-1
AS Barat (Oregon)	us-west-2
Asia Pasifik (Singapura)	ap-southeast-1
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Eropa (Frankfurt)	eu-central-1
Europe (Ireland)	eu-west-1
Europe (London)	eu-west-2
Europe (Paris)	eu-west-3

Untuk ketersediaan AWS AgentCore Gateway terbaru menurut Wilayah, lihat titik akhir dan kuota AWS AgentCore Gateway di Panduan Pengembang AWS AgentCore Gateway.

Kuota

Service quotas, juga disebut batasan, adalah jumlah maksimum sumber daya layanan atau operasi untuk akun AWS Anda.

Kuota untuk layanan AWS dalam solusi ini

Pastikan Anda memiliki kuota yang cukup untuk setiap layanan yang diterapkan dalam solusi ini. Untuk informasi lebih lanjut, lihat Service quotas AWS.

Gunakan tautan berikut untuk membuka halaman untuk layanan itu. Untuk melihat kuota layanan untuk semua layanan AWS dalam dokumentasi tanpa berpindah halaman, lihat informasi di titik akhir Layanan dan halaman kuota di PDF sebagai gantinya.

CloudFormation Kuota AWS

Akun AWS Anda memiliki CloudFormation kuota AWS yang harus Anda ketahui saat meluncurkan tumpukan dalam solusi ini. Dengan memahami kuota ini, Anda dapat menghindari kesalahan pembatasan yang akan mencegah Anda menerapkan solusi ini dengan sukses. Untuk informasi selengkapnya, lihat CloudFormation kuota AWS di Panduan CloudFormation Pengguna AWS.

Kuota pengujian beban

Jumlah maksimum tugas yang dapat dijalankan di Amazon ECS menggunakan jenis peluncuran AWS Fargate didasarkan pada ukuran tugas vCPU. Ukuran tugas default dalam Pengujian Beban Terdistribusi di AWS adalah 2 vCPU. Untuk melihat kuota default saat ini, lihat kuota layanan Amazon ECS. Kuota akun saat ini mungkin berbeda dari kuota yang tercantum. Untuk memeriksa kuota khusus untuk akun, periksa kuota layanan untuk jumlah sumber daya vCPU sesuai permintaan Fargate di AWS Management Console. Untuk petunjuk tentang cara meminta peningkatan, lihat kuota layanan AWS di Panduan Referensi Umum AWS.

Gambar kontainer Amazon Linux 2023 (dengan Taurus diinstal) tidak membatasi koneksi bersamaan per tugas, tetapi itu tidak berarti dapat mendukung jumlah pengguna yang tidak terbatas. Untuk menentukan jumlah pengguna bersamaan yang dapat dihasilkan kontainer untuk pengujian, lihat bagian Tentukan jumlah pengguna dari panduan ini.

> **Note**
>
> Batas yang disarankan untuk pengguna bersamaan berdasarkan pengaturan default adalah 200 pengguna.

Tes bersamaan

Solusi ini membuat CloudWatch dasbor Amazon untuk setiap pengujian yang menampilkan output gabungan dari semua tugas yang berjalan di cluster Amazon ECS secara real time. CloudWatch Dasbor menunjukkan waktu respons rata-rata, jumlah pengguna bersamaan, jumlah permintaan yang berhasil, dan jumlah permintaan yang gagal. Solusinya menggabungkan setiap metrik per detik dan memperbarui dasbor setiap menit.

Kebijakan pengujian Amazon EC2

Anda tidak memerlukan persetujuan dari AWS untuk menjalankan pengujian beban menggunakan solusi ini selama lalu lintas jaringan Anda tetap di bawah 1 Gbps. Jika pengujian Anda akan menghasilkan lebih dari 1 Gbps, hubungi AWS. Untuk informasi selengkapnya, lihat Kebijakan Pengujian Amazon EC2.

Kebijakan pengujian CloudFront beban Amazon

Jika Anda berencana untuk menguji beban CloudFront titik akhir, lihat pedoman pengujian beban di Panduan CloudFront Pengembang Amazon. Kami juga merekomendasikan penyebaran lalu lintas di beberapa tugas dan Wilayah. Berikan setidaknya 30 menit waktu ramp-up untuk uji beban. Untuk tes beban yang mengirimkan lebih dari 500.000 permintaan per detik atau menuntut lebih dari 300 Gbps data, kami sarankan terlebih dahulu mendapatkan pra-persetujuan untuk mengirim lalu lintas. CloudFront dapat membatasi lalu lintas uji beban yang tidak disetujui yang memengaruhi CloudFront ketersediaan layanan.

Memantau solusi pasca-penerapan

Setelah menerapkan solusi, kami sarankan untuk terus memantau sumber daya solusi menggunakan CloudWatch alarm dan metrik Amazon.

Menyiapkan CloudWatch alarm

Anda dapat mengatur CloudWatch alarm untuk memantau metrik kunci dan menerima pemberitahuan saat ambang batas terlampaui. Pertimbangkan untuk menyiapkan alarm untuk sumber daya berikut:

Metrik CloudFront distribusi Amazon

Pantau kinerja dan kesalahan CloudFront distribusi. Untuk informasi selengkapnya, lihat metrik CloudFront distribusi di Panduan CloudFront Pengembang Amazon.

Metrik Amazon API Gateway

Pantau tingkat permintaan API, latensi, dan kesalahan. Untuk informasi selengkapnya, lihat dimensi dan metrik Amazon API Gateway di Panduan Pengembang Amazon API Gateway.

Metrik fungsi AWS Lambda

Pantau pemanggilan fungsi Lambda, durasi, kesalahan, dan throttle untuk layanan mikro solusi.

Metrik Amazon ECS dan AWS Fargate

Pantau CPU tugas dan pemanfaatan memori selama tes beban untuk memastikan sumber daya yang memadai.

Metrik Amazon DynamoDB

Pantau konsumsi kapasitas baca dan tulis, permintaan terbatas, dan latensi.

Libatkan Ahli

AWS Countdown Premium Keterlibatan Jangka Pendek untuk Pengujian Beban Terdistribusi di AWS

Insinyur AWS kami memberikan panduan ahli tentang dasar-dasar pengujian kinerja, pengembangan skrip, dan analisis hasil. Daftar sekarang.

Ikhtisar

Keterlibatan Jangka Pendek AWS Countdown Premium (CDP) memberikan panduan ahli bagi organisasi yang melakukan pengujian kinerja dalam skala besar. Melalui model “do-it-yourself”

kolaboratif, insinyur AWS memberikan pengawasan strategis dan keahlian teknis sementara tim Anda mempertahankan tanggung jawab eksekusi. Insinyur AWS ahli tersedia dalam waktu satu minggu setelah pendaftaran tanpa memerlukan kontrak jangka panjang.

Model Layanan

Insinyur CDP bekerja bersama tim Anda untuk memberikan panduan dan pengawasan selama implementasi pengujian kinerja Anda. Pendekatan lepas tangan ini memastikan Anda menerima arahan ahli sambil membangun kemampuan internal. Layanan ini ideal untuk organisasi dengan kemampuan pengujian yang ada yang membutuhkan keahlian AWS khusus untuk mengimplementasikan Pengujian Beban Terdistribusi di AWS secara efektif.

Apa yang Disediakan Insinyur CDP

Insinyur CDP memandu Anda melalui dasar-dasar pengujian kinerja dan Pengujian Beban Terdistribusi pada arsitektur AWS. Mereka memberikan panduan tentang JMeter, struktur skrip K6, dan Locust dan pengembangan skrip pengujian, membantu penerapan CloudFormation templat, dan mengevaluasi hasil pengujian dengan rekomendasi pengoptimalan kinerja. Support mencakup analisis pemanfaatan sumber daya, penyelarasan praktik terbaik, dan end-to-end panduan dari penyiapan awal hingga analisis hasil, memungkinkan transfer pengetahuan ke tim Anda.

Tanggung Jawab Pelanggan

Tim Anda menangani konfigurasi tingkat aplikasi, pengembangan skrip pengujian, dan verifikasi skenario pengujian. Anda bertanggung jawab atas pelaksanaan dan operasi pengujian yang sebenarnya, termasuk semua aktivitas pengujian sebelum, selama, dan setelah peristiwa pengujian kinerja.

Manfaat Utama

Keterlibatan Jangka Pendek CDP memberikan pengurangan risiko melalui pengawasan ahli, panduan kontekstual khusus untuk beban kerja Anda, rekomendasi pengoptimalan kinerja, penyelesaian masalah yang lebih cepat, penyelarasan praktik terbaik, dan dukungan komprehensif sambil mempertahankan kepemilikan dan pengembangan kemampuan tim Anda.

Arsitektur yang Didukung

Pengujian Beban Terdistribusi di AWS mendukung pengujian untuk aplikasi web APIs, layanan mikro, dan arsitektur tanpa server dalam skala besar, memanfaatkan Pengujian Beban Terdistribusi pada solusi AWS. Kemampuan pengujian jauh melampaui kasus penggunaan umum ini untuk

memasukkan database, TCP/UDP protokol, direktori LDAP, server email SMTP, dan banyak sistem dan protokol lain yang memerlukan validasi kinerja di bawah beban.

Memulai

Organizations yang tertarik dengan Keterlibatan Jangka Pendek CDP untuk Pengujian Beban Terdistribusi di AWS dapat mendaftar langsung melalui situs web AWS di sini dan memilih “Use Case Implementation” untuk Area Fokus Anda.

Keluar dari Lingkup

CDP tidak menyediakan pengembangan skrip pengujian khusus (hanya panduan), mengelola operasi eksekusi pengujian, atau membuat laboratorium atau lokakarya langsung khusus. Dukungan di tempat juga di luar cakupan.

Terapkan solusinya

AWS Launch Wizard adalah metode penerapan yang disarankan untuk solusi ini. Ini menyediakan:

- Pengalaman konfigurasi terpandu dengan panel bantuan terperinci di setiap langkah
- Halaman terpusat untuk memantau kesehatan semua penyebaran Anda
- Indikasi ketika ada versi solusi yang lebih baru yang tersedia untuk penerapan atau peningkatan

Atau, Anda dapat menerapkan solusi secara langsung menggunakan CloudFormation template AWS.

Ikhtisar proses penyebaran

Sebelum Anda menerapkan solusi, tinjau biaya, arsitektur, keamanan, dan pertimbangan lain yang dibahas sebelumnya dalam panduan ini.

Waktu untuk menyebarkan: Sekitar 15 menit untuk tumpukan utama, ditambah 5 menit untuk setiap Wilayah tambahan

> **Note**
>
> Solusi ini mencakup metrik pengumpulan data ke AWS. Kami menggunakan data ini untuk lebih memahami bagaimana pelanggan menggunakan solusi ini dan layanan serta produk terkait. AWS memiliki data yang dikumpulkan melalui survei ini. Pengumpulan data tunduk pada Pemberitahuan Privasi AWS.

> **Note**
>
> Anda bertanggung jawab atas biaya layanan AWS yang digunakan saat menjalankan solusi ini. Untuk detail selengkapnya, kunjungi bagian Biaya dalam panduan ini dan lihat halaman web harga untuk setiap layanan AWS yang digunakan dalam solusi ini.

Terapkan menggunakan AWS Launch Wizard

Solusi ini menampilkan proses penerapan terpandu menggunakan AWS Launch Wizard. Ikuti langkah-langkah ini untuk menerapkan Pengujian Beban Terdistribusi di AWS ke akun Anda.

1. Masuk ke AWS Management Console dan pilih tombol di bawah ini untuk memulai proses penerapan.

2. Jika ada lebih dari satu pola penerapan yang tersedia untuk solusi, pilih salah satu yang paling sesuai untuk kasus penggunaan Anda.
3. Pilih versi yang akan diterapkan. Versi terbaru direkomendasikan.
4. Klik pada tombol Launch deployment wizard.

Anda kemudian akan mengikuti serangkaian langkah untuk mengumpulkan informasi yang diperlukan untuk menyebarkan solusi. Diperlukan waktu sekitar 15 menit untuk menyediakan sumber daya yang dibutuhkan.

Pilih penyebaran Anda dari daftar Deployment untuk melihat statusnya.

Terapkan menggunakan AWS CloudFormation

Solusi ini menggunakan CloudFormation templat dan tumpukan AWS untuk mengotomatiskan penerapannya. CloudFormation Template menentukan sumber daya AWS yang disertakan dalam solusi ini dan propertinya. CloudFormation Tumpukan menyediakan sumber daya yang dijelaskan dalam template.

CloudFormation Templat AWS

Anda dapat mengunduh CloudFormation template untuk solusi ini sebelum menerapkannya. Solusi ini menggunakan AWS CloudFormation untuk mengotomatiskan penerapan Pengujian Beban Terdistribusi di AWS. Ini mencakup CloudFormation template AWS berikut, yang dapat Anda unduh sebelum penerapan:

View template

distribu

load-testing-on-aws.template - Gunakan template ini untuk meluncurkan solusi dan semua komponen terkait. Konfigurasi default menerapkan layanan inti dan pendukung yang ditemukan di layanan AWS di bagian solusi ini, tetapi Anda dapat menyesuaikan template untuk memenuhi kebutuhan spesifik Anda.

> **Note**
>
> CloudFormation Sumber daya AWS dibuat dari konstruksi AWS Cloud Development Kit (AWS CDK). Jika sebelumnya Anda telah menerapkan solusi ini, lihat Memperbarui solusi untuk petunjuk pemutakhiran.

Luncurkan tumpukan

Ikuti langkah-langkah ini untuk menerapkan Pengujian Beban Terdistribusi pada solusi AWS ke akun Anda. CloudFormation Template AWS otomatis ini menerapkan Pengujian Beban Terdistribusi di AWS.

1. Masuk ke AWS Management Console dan pilih tombol untuk meluncurkan CloudFormation template.

   Launch solution

   Atau, Anda dapat mengunduh template sebagai titik awal untuk implementasi Anda sendiri.
2. Template diluncurkan di Wilayah AS Timur (Virginia N.) secara default. Untuk meluncurkan solusi ini di Wilayah AWS yang berbeda, gunakan pemilih wilayah di bilah navigasi konsol.

   > **Note**
   >
   > Solusi ini menggunakan Amazon Cognito, yang saat ini hanya tersedia di Wilayah AWS tertentu. Oleh karena itu, Anda harus meluncurkan solusi ini di Wilayah AWS tempat Amazon Cognito tersedia. Untuk ketersediaan layanan terbaru menurut Wilayah, lihat Daftar Layanan Regional AWS.

3. Pada halaman Buat tumpukan, verifikasi bahwa URL templat yang benar ditampilkan di kotak teks URL Amazon S3 dan pilih Berikutnya.
4. Pada halaman Tentukan detail tumpukan, tetapkan nama ke tumpukan solusi Anda.
5. Di bawah Parameter, tinjau parameter untuk templat dan modifikasi seperlunya. Solusi ini menggunakan nilai default berikut.

Parameter	Default	Deskripsi
Nama Administrator	<Requires input>	Nama pengguna untuk administrator solusi awal.
Email Administrator	*<Requires input>*	Alamat email pengguna administrator. Setelah diluncurkan, email akan dikirim ke alamat ini dengan instruksi login konsol.
ID VPC yang ada	<Optional input>	Jika Anda memiliki VPC yang ingin Anda gunakan dan sudah dibuat, masukkan ID VPC yang ada di Wilayah yang sama tempat tumpukan digunakan. Misalnya, vpc-1a2b3c4d5e6f.
Subnet pertama yang ada	<Optional input>	ID subnet pertama dalam VPC Anda yang ada. Subnet ini membutuhkan rute ke internet untuk menarik gambar kontainer untuk menjalankan pengujian. Misalnya, subnet-7h8i9j0k.
Subnet kedua yang ada	<Optional input>	ID subnet kedua dalam VPC yang ada. Subnet ini membutuhkan rute ke internet untuk menarik gambar

Parameter	Default	Deskripsi
		kontainer untuk menjalankan pengujian. Misalnya, subnet-1x2y3z.
Berikan blok CIDR yang valid untuk solusi membuat VPC	192.168.0.0/16	Anda dapat membiarkan parameter ini kosong jika Anda menggunakan VPC yang ada
Berikan blok CIDR yang valid untuk subnet A untuk solusi membuat VPC	192.168.0.0/20	Blok CIDR untuk subnet A dari AWS Fargate VPC
Berikan blok CIDR yang valid untuk subnet B untuk solusi membuat VPC	192.168.16.0/20	Blok CIDR untuk subnet B dari AWS Fargate VPC
Menyediakan blok CIDR untuk memungkinkan lalu lintas keluar dari tugas Fargate	0.0.0.0/0	Blok CIDR yang membatasi akses keluar kontainer Amazon ECS.
Perbarui Gambar Kontainer Otomatis	No	Secara otomatis menggunakan gambar yang paling mutakhir dan aman hingga rilis minor berikutnya. Memilih No akan menarik gambar seperti yang dirilis semula, tanpa pembaruan keamanan apa pun.

Parameter	Default	Deskripsi
Menyebarkan Server MCP Opsional	No	Terapkan server MCP jarak jauh opsional, menggunakan AgentCore Gateway untuk menghubungkan aplikasi AI ke Pengujian Beban Terdistribusi di AWS.

6. Pilih Berikutnya.
7. Pada halaman Konfigurasikan opsi tumpukan, pilih Berikutnya.
8. Pada halaman Ulasan, tinjau dan konfirmasikan pengaturan. Centang kotak yang menyatakan bahwa template akan membuat sumber daya AWS Identity and Access Management (IAM).
9. Pilih Membuat tumpukan untuk menerapkannya.

Anda dapat melihat status tumpukan di CloudFormation konsol AWS di kolom Status. Anda akan menerima status CREATE_COMPLETE dalam waktu sekitar 15 menit.

> **Note**
>
> Selain fungsi AWS Lambda utama, solusi ini mencakup fungsi Lambda sumber daya khusus, yang hanya berjalan selama konfigurasi awal atau saat sumber daya diperbarui atau dihapus.

Saat menjalankan solusi ini, fungsi Lambda sumber daya khusus tidak aktif. Namun, jangan hapus fungsi ini karena perlu untuk mengelola sumber daya terkait.

Penyebaran Multi-Wilayah

Waktu untuk menyebarkan: Sekitar 5 menit per Wilayah

Anda dapat menjalankan pengujian di beberapa Wilayah.

Saat Anda menerapkan solusi Pengujian Beban Terdistribusi, solusi ini akan membuat CloudFormation templat regional di bucket skenario S3. URL untuk template ini tercantum dalam CloudFormation output tumpukan utama Anda di bawah kunci “RegionalCFTemplate”.

Untuk menjalankan pengujian Multi-wilayah, Anda harus menerapkan CloudFormation templat regional di setiap Wilayah tempat Anda ingin menjalankan pengujian.

> **Note**
>
> Setiap akun AWS hanya dapat menggunakan satu tumpukan regional per wilayah. Selain itu, tumpukan regional tidak dapat digunakan di wilayah yang sama dengan tumpukan utama.

Anda dapat menginstal template regional sebagai berikut:

1. Di konsol web solusi, navigasikan ke Dasbor di menu sebelah kiri.
2. Gunakan ikon clipboard untuk menyalin tautan CloudFormation templat di Amazon S3.
3. Masuk ke CloudFormation konsol AWS dan pilih Wilayah yang benar.
4. Pada halaman Buat tumpukan, verifikasi bahwa URL templat yang benar ditampilkan di kotak teks URL Amazon S3 dan pilih Berikutnya.
5. Pada halaman Tentukan detail tumpukan, tetapkan nama ke tumpukan solusi Anda.
6. Di bawah Parameter, tinjau parameter untuk templat dan modifikasi seperlunya. Solusi ini menggunakan nilai default berikut.

Parameter	Default	Deskripsi
ID VPC yang ada	<Optional input>	Jika Anda memiliki VPC yang ingin Anda gunakan dan sudah dibuat, masukkan ID VPC yang ada di Wilayah yang sama tempat tumpukan digunakan. Misalnya, vpc-1a2b3c4d5e6f.
Subnet pertama yang ada	<Optional input>	ID subnet pertama dalam VPC Anda yang ada. Subnet ini membutuhkan rute ke internet untuk menarik gambar kontainer untuk

Parameter	Default	Deskripsi
		menjalankan pengujian. Misalnya, subnet-7h8i9j0k.
Subnet kedua yang ada	<Optional input>	ID subnet kedua dalam VPC yang ada. Subnet ini membutuhkan rute ke internet untuk menarik gambar kontainer untuk menjalank an pengujian. Misalnya, subnet-1x2y3z.
Berikan blok CIDR yang valid untuk solusi membuat VPC	192.168.0.0/16	Jika Anda tidak memberika n nilai untuk VPC yang ada, blok CIDR untuk Amazon VPC yang dibuat solusi berisi alamat IP untuk AWS Fargate.
Menyediakan blok CIDR untuk memungkinkan lalu lintas keluar dari tugas Fargate	0.0.0.0/0	Blok CIDR yang membatasi akses keluar kontainer Amazon ECS.

7. Pilih Berikutnya.
8. Pada halaman Konfigurasikan opsi tumpukan, pilih Berikutnya.
9. Pada halaman Ulasan, tinjau dan konfirmasikan pengaturan. Pastikan untuk mencentang kotak yang mengakui bahwa template akan membuat sumber daya AWS Identity and Access Management (IAM).
10. Pilih Membuat tumpukan untuk menerapkannya.

    Anda dapat melihat status tumpukan di CloudFormation konsol AWS di kolom Status. Anda akan menerima status CREATE_COMPLETE dalam waktu kurang lebih lima menit.

Ketika Wilayah telah berhasil digunakan, mereka muncul di konsol web. Saat Anda membuat pengujian, semua Wilayah yang tersedia tercantum di Dasbor dan dalam Pembuatan Skenario. Anda dapat menambahkan Wilayah ke pengujian di langkah Bentuk Lalu Lintas Pembuatan Skenario.

Solusinya membuat item DynamoDB untuk setiap Wilayah yang digunakan dalam tabel skenario, yang berisi informasi yang diperlukan tentang sumber daya pengujian di Wilayah tersebut. Anda dapat mengurutkan hasil pengujian di konsol web berdasarkan Wilayah. Untuk melihat hasil agregat di semua Wilayah dalam pengujian Multi-wilayah, gunakan metrik Amazon CloudWatch . Anda dapat menemukan kode sumber untuk grafik dalam hasil tes setelah tes selesai.

> **Note**
>
> Anda dapat meluncurkan tumpukan regional tanpa konsol web. Dapatkan tautan ke templat regional di bucket skenario Amazon S3 dan berikan sebagai sumber saat meluncurkan tumpukan regional di Wilayah yang diperlukan. Atau, Anda dapat mengunduh templat dan mengunggahnya sebagai sumber untuk Wilayah yang Anda inginkan.

Perbarui solusinya

Memperbarui solusi menerapkan fitur terbaru, patch keamanan, dan perbaikan bug untuk penerapan Anda. Untuk memperbarui ke versi terbaru, lihat bagian yang sesuai berdasarkan metode penerapan asli Anda: AWS Launch Wizard atau AWS CloudFormation.

> **Important**
>
> Sebelum memperbarui, pastikan tidak ada tes beban yang sedang berjalan. Proses pembaruan dapat mengganggu ketersediaan solusi untuk sementara.

Perbarui menggunakan AWS Launch Wizard

Konsol secara otomatis menampilkan versi solusi terbaru yang tersedia di dropdown versi Deployment. Jika sebelumnya Anda telah menerapkan solusi, ikuti prosedur ini untuk memperbarui penerapan Anda ke versi terbaru.

1. Buka Launch Wizard Deployment.
2. Pilih penyebaran yang ingin Anda perbarui.
3. Pilih Tindakan, lalu Perbarui versi penerapan.
4. Pilih versi terbaru dari versi Deployment yang tersedia.
5. Tinjau konfigurasi.
6. Buat perubahan yang diperlukan pada setiap langkah.
7. Konfirmasikan pembaruan.

Perbarui menggunakan AWS CloudFormation

Jika sebelumnya Anda telah menerapkan solusi, ikuti prosedur ini untuk memperbarui CloudFormation tumpukan ke versi terbaru.

1. Masuk ke CloudFormation konsol, pilih CloudFormation tumpukan yang ada, dan pilih Perbarui tumpukan.
2. Pilih Buat pembaruan langsung.
3. Pilih Ganti template yang ada.

4. Di bawah Tentukan template:
   a. Pilih URL Amazon S3.
   b. Salin tautan templat terbaru.
   c. Tempel tautan di kotak URL Amazon S3.
   d. Verifikasi bahwa URL templat yang benar ditampilkan di kotak teks URL Amazon S3.
   e. Pilih Berikutnya.
   f. Pilih Selanjutnya sekali lagi.
5. Di bawah Parameter, tinjau parameter untuk templat dan modifikasi sesuai kebutuhan. Lihat Luncurkan tumpukan untuk detail tentang parameter.
6. Pilih Berikutnya.
7. Pada halaman Konfigurasikan opsi tumpukan, pilih Berikutnya.
8. Pada halaman Ulasan, tinjau dan konfirmasikan pengaturan.
9. Pilih kotak yang mengakui bahwa template mungkin membuat sumber daya IAM.
10. Pilih Lihat set perubahan dan verifikasi perubahan.
11. Pilih Perbarui tumpukan untuk menyebarkan tumpukan.

Anda dapat melihat status tumpukan di CloudFormation konsol AWS di kolom Status. Anda akan menerima `UPDATE_COMPLETE` status dalam waktu sekitar 15 menit.

> **Note**
>
> Jika Anda mengalami masalah autentikasi Amazon Cognito saat masuk dari browser setelah peningkatan tumpukan, segarkan browser Anda (Ctrl+Shift+R aktif Windows/Linux atau Cmd +Shift+R di Mac) untuk menghapus data yang di-cache dan coba lagi.

Pemecahan masalah pembaruan dari versi sebelum v3.3.0

> **Note**
>
> Bagian ini hanya berlaku untuk pembaruan dari versi sebelum v3.3.0. Jika Anda memperbarui dari v3.3.0 atau versi lebih baru, ikuti prosedur pembaruan standar melalui AWS Launch Wizard atau AWS. CloudFormation

1. Unduh distributed-load-testing-on-aws.template.
2. Buka template dan navigasikan ke `Conditions:` dan cari`DLTCommonResourcesAppRegistryCondition`.
3. Anda akan melihat sesuatu yang serupa dengan yang berikut:

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

4. Ubah `true` nilai kedua menjadi`false`:

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

5. Gunakan template yang disesuaikan untuk memperbarui tumpukan Anda dengan mengikuti langkah-langkah dalam Pembaruan menggunakan AWS CloudFormation.
6. Pembaruan ini menghapus sumber daya terkait registri aplikasi dari tumpukan, sehingga pembaruan berhasil diselesaikan.
7. Lakukan pembaruan tumpukan lain menggunakan URL template terbaru.

Memperbarui tumpukan regional

Jika Anda telah menerapkan solusi di beberapa Wilayah, Anda harus memperbarui setiap tumpukan regional secara terpisah. Ikuti prosedur pembaruan standar untuk setiap CloudFormation tumpukan regional di Wilayah tempat Anda menerapkan infrastruktur pengujian.

Manajer Aplikasi AWS Systems Manager

Setelah memperbarui solusi, AWS Systems Manager Application Manager menyediakan tampilan tingkat aplikasi ke dalam solusi dan sumber dayanya. Anda dapat menggunakan Manajer Aplikasi untuk:

- Pantau sumber daya, biaya untuk sumber daya yang digunakan di seluruh tumpukan dan akun AWS, dan log dari lokasi pusat.
- Melihat data operasi untuk sumber daya solusi dalam konteks aplikasi, seperti status penerapan, CloudWatch alarm, konfigurasi sumber daya, dan masalah operasional.

Pemecahan Masalah

Resolusi masalah yang diketahui memberikan instruksi untuk mengurangi kesalahan yang diketahui. Jika petunjuk ini tidak mengatasi masalah Anda, Hubungi AWS Support memberikan petunjuk untuk membuka kasus AWS Support untuk solusi ini.

Resolusi masalah yang diketahui

Masalah: Anda menggunakan VPC yang ada dan pengujian Anda gagal dengan status Gagal, menghasilkan pesan galat berikut:

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

- Penyelesaian:

Pastikan bahwa subnet ada di VPC yang ditentukan dan bahwa mereka memiliki rute ke internet dengan gateway internet atau gateway NAT. AWS Fargate memerlukan akses untuk menarik image container dari repositori publik agar berhasil menjalankan pengujian.

Masalah: Pengujian memakan waktu terlalu lama untuk dijalankan atau macet berjalan tanpa batas

- Penyelesaian:

Batalkan pengujian dan periksa AWS Fargate untuk memastikan bahwa semua tugas telah berhenti. Jika mereka belum berhenti, hentikan semua tugas Fargate secara manual. Periksa batas tugas Fargate sesuai permintaan di akun Anda untuk memastikan bahwa Anda dapat meluncurkan jumlah tugas yang diinginkan. Anda juga dapat memeriksa CloudWatch log untuk fungsi runner tugas Lambda untuk wawasan lebih lanjut tentang kegagalan saat meluncurkan tugas Fargate. Periksa log CloudWatch ECS untuk detail tentang apa yang terjadi di kontainer Fargate yang sedang berjalan.

Masalah: Tes dimulai tetapi gagal diselesaikan atau status tugas ECS tidak diketahui

- Penyelesaian:

Jika Anda memilih opsi untuk menyediakan VPC yang ada di akun tempat solusi telah digunakan, pastikan bahwa VPC yang digunakan oleh Tugas ECS memiliki alamat IP gratis yang cukup untuk

memulai jumlah tugas yang disediakan dalam input pengujian. Definisi tugas ECS menggunakan gambar ECR yang membutuhkan gateway internet atau rute ke internet sehingga layanan ECS dapat menyediakan tugas dengan mengunduh gambar solusi ECR dari aws-solutions/ -. distributed-load-testing-on aws-load-tester Jika Anda tidak dapat memberikan rute ke internet karena semua subnet di VPC bersifat pribadi, Anda dapat meng-host gambar ECR di akun Anda menggunakan ECR pull through cache. Perbarui definisi tugas dengan URI gambar ECR baru dan buat revisi baru. Setelah definisi tugas diperbarui, konfigurasi solusi dalam tabel DynamoDB perlu diperbarui untuk menggunakan revisi baru. Nama tabel DynamoDB dapat ditemukan di tab output tumpukan CloudFormation di bawah kunci. ScenariosTable Perbarui atribut TaskDefinition untuk item dengan kunci TestID dan nilai region- [SOLUTION-DEPLOYED-REGION].

Masalah: Pengujian perlu menggunakan titik akhir yang bersifat pribadi atau tidak tersedia melalui gateway internet

- Penyelesaian:

Saat menguji titik akhir API pribadi yang tidak dapat diakses melalui gateway internet, pertimbangkan pendekatan berikut:

1. Konfigurasi Jaringan: Pastikan tabel rute subnet yang digunakan oleh tugas ECS diperbarui dengan rute ke rentang alamat IP dari titik akhir pribadi yang sedang diuji. Ini memungkinkan lalu lintas pengujian mencapai titik akhir pribadi dalam VPC Anda.
2. Resolusi DNS: Untuk domain kustom, konfigurasikan pengaturan DNS di VPC Anda untuk menyelesaikan nama domain titik akhir pribadi. Lihat dokumentasi DNS VPC untuk petunjuk terperinci.
3. Titik Akhir VPC: Jika menguji layanan AWS, pertimbangkan untuk menggunakan titik akhir VPC (PrivateLinkAWS) untuk membuat konektivitas pribadi. Misalnya, untuk menguji API Gateway pribadi, Anda dapat membuat titik akhir VPC untuk API Gateway. Lihat dokumentasi API Gateway Pribadi.
4. Pengintip VPC: Jika titik akhir pribadi berada di VPC yang berbeda, buat peering VPC antara VPC tempat solusi diterapkan dan VPC yang berisi titik akhir pribadi. Konfigurasikan tabel rute yang sesuai di keduanya VPCs. Lihat dokumentasi VPC Peering.
5. Transit Gateway: Untuk skenario jaringan yang lebih kompleks yang melibatkan beberapa VPCs, pertimbangkan untuk menggunakan AWS Transit Gateway untuk merutekan lalu lintas antara VPC solusi dan VPC yang berisi titik akhir pribadi. Lihat dokumentasi Transit Gateway.

6. Grup Keamanan: Pastikan bahwa grup keamanan yang terkait dengan tugas ECS Anda memungkinkan lalu lintas keluar ke titik akhir pribadi, dan grup keamanan titik akhir pribadi mengizinkan lalu lintas masuk dari tugas ECS.

Untuk menguji Application Load Balancer internal atau instans EC2, pastikan rentang VPC CIDR tidak tumpang tindih dan rute yang diperlukan dikonfigurasi dalam tabel rute.

Masalah: Pengujian selesai tetapi hasilnya tidak tersedia di UI

- Penyelesaian:

Jika pengujian telah selesai tetapi hasilnya tidak tersedia di UI, file hasil harus tetap tersedia di Bucket S3 dari tugas ECS yang menjalankan pengujian. Ini adalah batasan yang diketahui dalam solusinya. Dalam arsitektur saat ini, solusinya menggunakan fungsi penguraian hasil Lambda untuk meringkas hasil dari beberapa tugas ECS, yang kemudian disimpan sebagai item dalam tabel DynamoDB. Tabel DynamoDB memiliki batas ukuran item maksimum 400 KB. Keterbatasan ini tercapai tergantung pada kompleksitas skrip pengujian, konkurensi, dan jumlah tugas yang digunakan. Kesalahan tidak berarti tes gagal; ini menunjukkan bahwa proses untuk meringkas hasil dan menyimpannya dalam tabel DynamoDB untuk operasi CRUD telah gagal. Hasilnya masih tersedia di bucket S3 untuk skenario pengujian.

Hubungi AWS Support

Jika Anda memiliki AWS Business Support+, AWS Enterprise Support, atau Unified Operations, Anda dapat menggunakan AWS Support Center untuk mendapatkan bantuan ahli terkait solusi ini. Bagian berikut memberikan petunjuk.

Buat kasus

1. Masuk ke Support Center.
2. Pilih Buat kasus.

Bagaimana kami bisa membantu?

1. Pilih Teknis
2. Untuk Layanan, pilih Solusi.

3. Untuk Kategori, pilih Pengujian Beban Terdistribusi di AWS.
4. Untuk Keparahan, pilih opsi yang paling cocok dengan kasus penggunaan Anda.
5. Saat Anda memasukkan Layanan, Kategori, dan Tingkat Keparahan, antarmuka akan mengisi tautan ke pertanyaan pemecahan masalah umum. Jika Anda tidak dapat menyelesaikan pertanyaan Anda dengan tautan ini, pilih Langkah selanjutnya: Informasi tambahan.

Informasi tambahan

1. Untuk Subjek, masukkan teks yang merangkum pertanyaan atau masalah Anda.
2. Untuk Deskripsi, jelaskan masalah secara rinci, termasuk nama produk ini dan versi yang Anda gunakan, seperti contoh ini: Pengujian Beban Terdistribusi di AWS VX.Y.z.
3. Pilih Lampirkan file.
4. Lampirkan informasi yang dibutuhkan AWS Support untuk memproses permintaan.

Bantu kami menyelesaikan kasus Anda lebih cepat

1. Masukkan informasi yang diminta.
2. Pilih Langkah selanjutnya: Selesaikan sekarang atau hubungi kami.

Selesaikan sekarang atau hubungi kami

1. Tinjau solusi Selesaikan sekarang.
2. Jika Anda tidak dapat menyelesaikan masalah Anda dengan solusi ini, pilih Hubungi kami, masukkan informasi yang diminta, dan pilih Kirim.

Copot pemasangan solusinya

Anda dapat menghapus instalasi Pengujian Beban Terdistribusi pada solusi AWS dari AWS Management Console atau dengan menggunakan AWS Command Line Interface. Anda harus secara manual menghapus konsol, skenario, dan mencatat bucket Amazon Simple Storage Service (Amazon S3) yang dibuat oleh solusi ini. Implementasi AWS Solutions tidak secara otomatis menghapusnya jika Anda memiliki data untuk disimpan.

> **Note**
>
> Jika Anda telah menerapkan tumpukan regional, Anda harus menghapus tumpukan di Wilayah tersebut sebelum menghapus tumpukan utama.

Menggunakan Konsol Manajemen AWS

AWS CloudFormation

1. Masuk ke CloudFormation konsol AWS.
2. Pada halaman Stacks, pilih tumpukan instalasi solusi ini.
3. Pilih Hapus.

AWS Launch Wizard

1. Masuk ke konsol AWS Launch Wizard.
2. Pada halaman Launch Wizard Deployment, pilih penerapan solusi ini.
3. Pilih Actions (Tindakan), lalu Delete (Hapus).
4. Konfirmasi penghapusan.

Menggunakan AWS Command Line Interface

Tentukan apakah AWS Command Line Interface (AWS CLI) tersedia di lingkungan Anda. Untuk petunjuk penginstalan, lihat Apa itu Antarmuka Baris Perintah AWS di Panduan Pengguna AWS CLI. Setelah mengonfirmasi bahwa AWS CLI tersedia, jalankan perintah berikut.

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

Menghapus bucket Amazon S3

Solusi ini dikonfigurasi untuk mempertahankan bucket Amazon S3 yang dibuat solusi (untuk diterapkan di Wilayah keikutsertaan) jika Anda memutuskan untuk menghapus tumpukan AWS untuk mencegah kehilangan data yang tidak disengaja. CloudFormation Setelah menghapus instalan solusi, Anda dapat menghapus bucket S3 ini secara manual jika Anda tidak perlu menyimpan data. Ikuti langkah-langkah ini untuk menghapus bucket Amazon S3.

1. Masuk ke konsol Amazon S3.
2. Pilih Bucket dari panel navigasi kiri.
3. Di bidang Temukan ember berdasarkan nama, masukkan nama tumpukan solusi ini.
4. Pilih salah satu bucket S3 solusi dan pilih Kosong.
5. Masukkan hapus secara permanen di bidang verifikasi dan pilih Kosong.
6. Pilih bucket S3 yang baru saja Anda kosongkan dan pilih Delete.
7. Masukkan nama bucket S3 di kolom verifikasi dan pilih Delete bucket.

Ulangi langkah 4 hingga 7 hingga Anda menghapus semua ember S3.

Untuk menghapus bucket S3 menggunakan AWS CLI, jalankan perintah berikut:

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

Gunakan solusinya

Bagian ini memberikan panduan komprehensif untuk menggunakan solusi Pengujian Beban Terdistribusi pada AWS, mulai dari membuat skenario pengujian pertama hingga menganalisis hasil terperinci. Alur kerja termasuk membuat skenario pengujian, menjalankan pengujian, dan menjelajahi hasil pengujian.

Buat skenario pengujian

Membuat skenario pengujian melibatkan empat langkah utama: mengonfigurasi pengaturan umum, mendefinisikan skenario, membentuk pola lalu lintas, dan meninjau konfigurasi Anda.

Langkah 1: Pengaturan umum

Konfigurasikan parameter dasar untuk uji beban Anda termasuk nama pengujian, deskripsi, dan opsi konfigurasi umum.

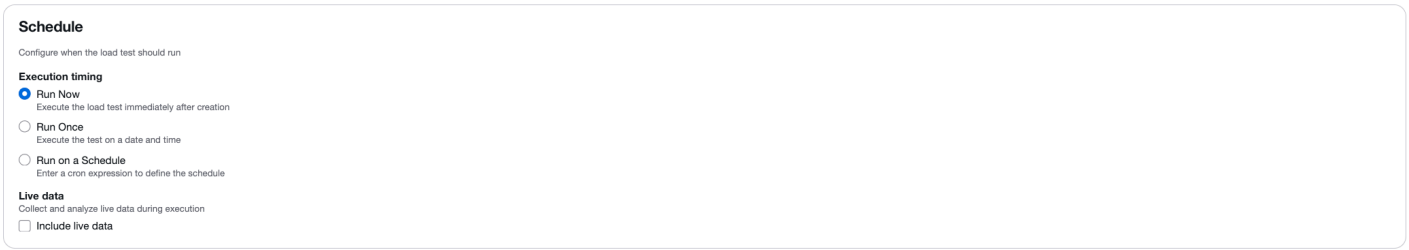
Identifikasi tes

- Nama uji (Wajib) - Nama deskriptif untuk skenario pengujian Anda
- Deskripsi tes (Wajib) - Rincian tambahan tentang tujuan pengujian dan konfigurasi
- Tag (Opsional) - Tambahkan hingga 5 tag untuk mengkategorikan dan mengatur skenario pengujian Anda

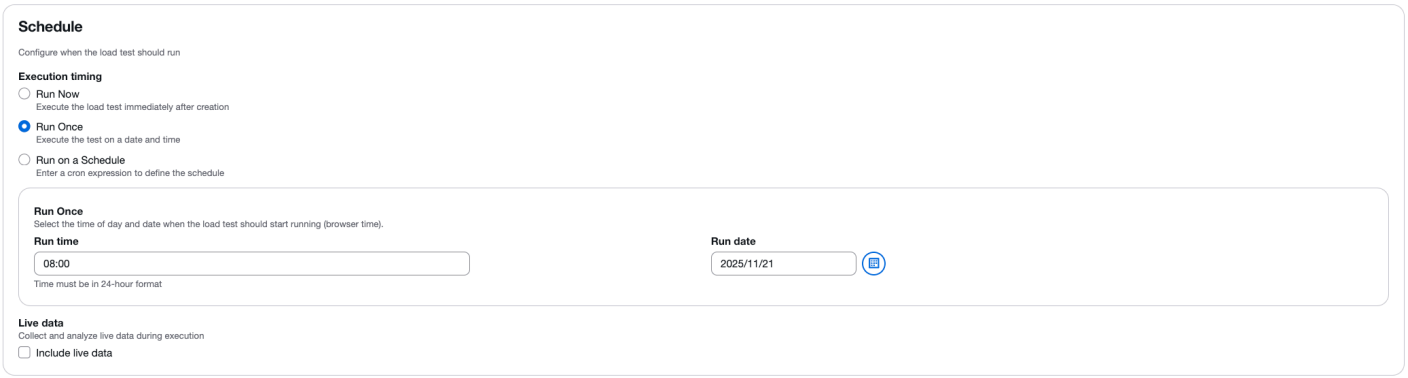
Opsi penjadwalan

Konfigurasikan kapan pengujian harus dijalankan:

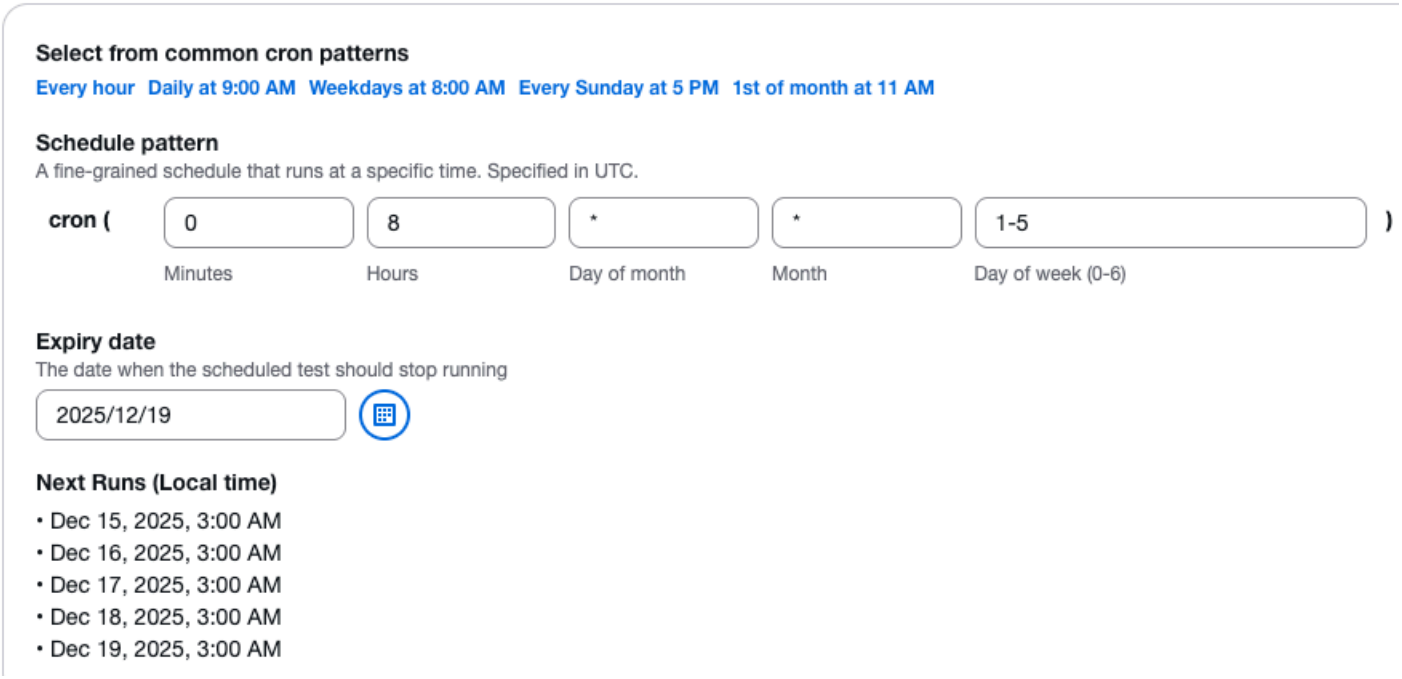
- Jalankan Sekarang - Jalankan tes segera setelah pembuatan.



- Jalankan Sekali - Jadwalkan tes untuk dijalankan pada tanggal dan waktu tertentu.



- Jalankan pada Jadwal - Gunakan penjadwalan berbasis cron untuk menjalankan tes secara otomatis secara berkala. Anda dapat memilih dari pola umum (setiap jam, harian, mingguan) atau menentukan ekspresi cron kustom.



Alur kerja penjadwalan

Saat Anda menjadwalkan tes, alur kerja berikut akan terjadi:

- Parameter jadwal dikirim ke API solusi melalui Amazon API Gateway.
- API meneruskan parameter ke fungsi Lambda yang membuat aturan CloudWatch Acara yang dijadwalkan berjalan pada tanggal yang ditentukan.
- Untuk pengujian satu kali (Jalankan Sekali), aturan CloudWatch Peristiwa berjalan pada tanggal yang ditentukan dan fungsi `api-services` Lambda menjalankan pengujian.

- Untuk pengujian berulang (Jalankan pada Jadwal), aturan CloudWatch Acara diaktifkan pada tanggal yang ditentukan, dan fungsi `api-services` Lambda membuat aturan baru yang berjalan segera dan berulang berdasarkan frekuensi yang ditentukan.

Data langsung

Pilih kotak centang Sertakan data langsung untuk melihat metrik waktu nyata saat pengujian Anda berjalan. Saat diaktifkan, Anda dapat memantau:

- Waktu respons rata-rata.
- Jumlah pengguna virtual.
- Permintaan yang berhasil dihitung.
- Jumlah permintaan gagal.

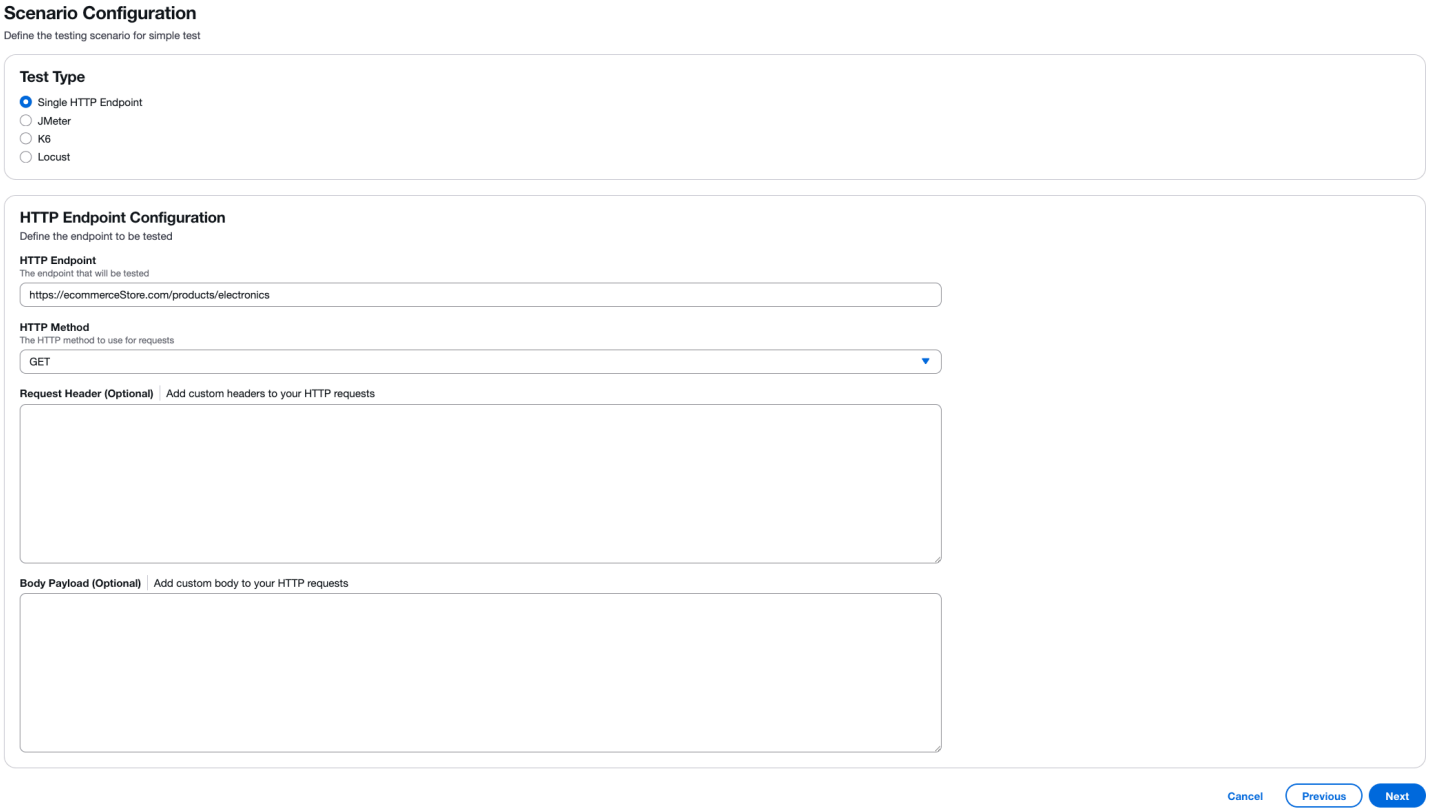
Fitur data langsung menyediakan bagan waktu nyata dengan data yang dikumpulkan pada interval satu detik. Untuk informasi lebih lanjut, lihat Pemantauan dengan data langsung.

Langkah 2: Konfigurasi skenario

Tentukan skenario pengujian spesifik dan pilih kerangka pengujian pilihan Anda.

Pemilihan jenis uji

Pilih jenis uji beban yang ingin Anda lakukan:



- Single HTTP Endpoint - Uji titik akhir API tunggal atau halaman web dengan konfigurasi sederhana.
- JMeter- Unggah skrip JMeter uji (file.jmx atau arsip.zip).
- K6 - Unggah skrip uji K6 (file.js atau arsip.zip).
- Locust - Unggah skrip pengujian Locust (file.py atau arsip.zip).

Gambar konfigurasi titik akhir HTTP: :images/test-types.png [Pilih jenis pengujian yang akan dijalankan] Saat “Titik Akhir HTTP Tunggal” dipilih, konfigurasikan pengaturan ini:

Titik Akhir HTTP (Diperlukan)

Masukkan URL lengkap dari titik akhir yang ingin Anda uji. Misalnya, `https://api.example.com/users`. Pastikan titik akhir dapat diakses dari infrastruktur AWS.

Metode HTTP (Diperlukan)

Pilih metode HTTP untuk permintaan Anda. Default-nya adalah `GET`. Pilihan lain termasuk`POST`,`PUT`,`DELETE`,`PATCH`,`HEAD`, dan`OPTIONS`.

Permintaan Header (Opsional)

Tambahkan header HTTP kustom ke permintaan Anda. Contoh umumnya meliputi:

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

Pilih Tambahkan Header untuk menyertakan beberapa header.

Muatan Tubuh (Opsional)

Tambahkan konten badan permintaan untuk permintaan POST atau PUT. Mendukung JSON, XHTML, atau format teks biasa. Sebagai contoh: `{"userId": 123, "action": "test"}`.

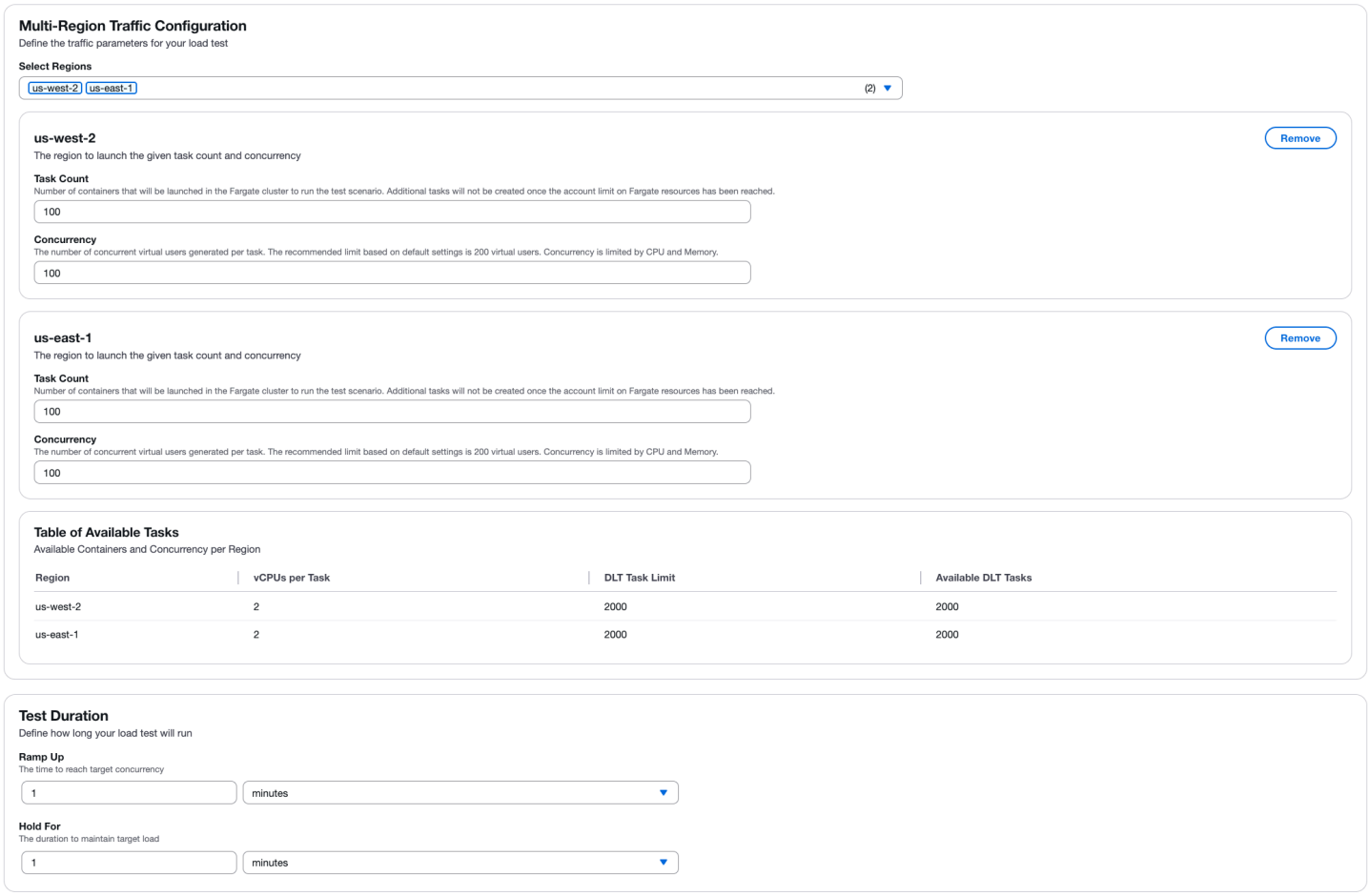
Skrip kerangka uji

Saat menggunakan JMeter, K6, atau Locust, unggah file skrip pengujian Anda atau arsip.zip yang berisi skrip pengujian dan file pendukung Anda. Untuk JMeter, Anda dapat menyertakan plugin khusus dalam `/plugins` folder dalam arsip.zip Anda.

> **Important**
>
> Meskipun skrip pengujian Anda (JMeter, K6, atau Locust) dapat menentukan konkurensi (pengguna virtual), tingkat transaksi (TPS), waktu peningkatan, dan parameter pemuatan lainnya, solusinya akan mengganti konfigurasi ini dengan nilai yang Anda tentukan di Layar Bentuk Lalu Lintas selama pembuatan pengujian. Konfigurasi Bentuk Lalu Lintas mengontrol jumlah tugas, konkurensi (pengguna virtual per tugas), durasi peningkatan, dan durasi penahanan untuk eksekusi pengujian.

Langkah 3: Bentuk lalu lintas

Konfigurasikan bagaimana lalu lintas akan didistribusikan selama pengujian Anda, termasuk dukungan multi-wilayah.



Konfigurasi lalu lintas multi-wilayah

Pilih satu atau beberapa wilayah AWS untuk mendistribusikan uji beban Anda secara geografis. Untuk setiap wilayah yang dipilih, konfigurasikan:

Hitungan Tugas

> Jumlah kontainer (tugas) yang akan diluncurkan di cluster Fargate untuk skenario pengujian. Tugas tambahan tidak akan dibuat setelah akun mencapai batas “Sumber daya Fargate telah tercapai”.

Konkurensi

> Jumlah pengguna virtual bersamaan yang dihasilkan per tugas. Batas yang disarankan didasarkan pada pengaturan default 2 v CPUs per tugas. Konkurensi dibatasi oleh sumber daya CPU dan Memori.

Tentukan jumlah pengguna

Jumlah pengguna yang dapat didukung kontainer untuk pengujian dapat ditentukan dengan secara bertahap meningkatkan jumlah pengguna dan memantau kinerja di Amazon CloudWatch. Setelah Anda mengamati bahwa kinerja CPU dan memori mendekati batasnya, Anda telah mencapai jumlah maksimum pengguna yang dapat didukung oleh wadah untuk pengujian tersebut dalam konfigurasi defaultnya (2 vCPU dan memori 4 GB).

Proses kalibrasi

Anda dapat mulai menentukan batas pengguna bersamaan untuk pengujian Anda dengan menggunakan contoh berikut:

1. Buat tes dengan tidak lebih dari 200 pengguna.
2. Saat pengujian berjalan, pantau CPU dan Memori menggunakan CloudWatch konsol:
   a. Dari panel navigasi kiri, di bawah Container Insights, pilih Performance Monitoring.
   b. Pada halaman pemantauan kinerja, dari menu drop-down kiri, pilih ECS Clusters.
   c. Dari menu tarik-turun kanan, pilih klaster Amazon Elastic Container Service (Amazon ECS) Anda.
3. Saat memantau, perhatikan CPU dan Memori. Jika CPU tidak melampaui 75% atau Memori tidak melampaui 85% (abaikan puncak satu kali), Anda dapat menjalankan tes lain dengan jumlah pengguna yang lebih tinggi.

Ulangi langkah 1-3 jika tes tidak melebihi batas sumber daya. Secara opsional, Anda dapat meningkatkan sumber daya kontainer untuk memungkinkan jumlah pengguna bersamaan yang lebih tinggi. Namun, ini menghasilkan biaya yang lebih tinggi. Untuk detailnya, lihat Panduan Pengembang.

> **Note**
>
> Untuk hasil yang akurat, jalankan hanya satu pengujian pada satu waktu saat menentukan batas pengguna bersamaan. Semua pengujian menggunakan cluster yang sama, dan wawasan CloudWatch kontainer mengumpulkan data kinerja berdasarkan cluster. Hal ini menyebabkan kedua pengujian dilaporkan ke CloudWatch Container Insights secara bersamaan, yang menghasilkan metrik pemanfaatan sumber daya yang tidak akurat untuk satu pengujian.

Untuk informasi lebih lanjut tentang kalibrasi pengguna per mesin, lihat Mengkalibrasi Tes Taurus dalam dokumentasi. BlazeMeter

> **Note**
>
> Solusi ini menampilkan informasi kapasitas yang tersedia untuk setiap wilayah, membantu Anda merencanakan konfigurasi pengujian dalam batas yang tersedia.

Tabel tugas yang tersedia

Tabel Tugas yang Tersedia menampilkan ketersediaan sumber daya untuk setiap wilayah yang dipilih:

- Wilayah - Nama wilayah AWS.
- v CPUs per Tugas - Jumlah virtual yang CPUs dialokasikan untuk setiap tugas (default: 2).
- Batas Tugas DLT - Jumlah maksimum tugas yang dapat dibuat berdasarkan batas Fargate akun Anda (default: 2000).
- Tugas DLT yang Tersedia - Jumlah tugas saat ini yang tersedia untuk digunakan di wilayah (default: 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

Untuk menambah jumlah tugas yang tersedia atau v CPUs per tugas, lihat Panduan Pengembang.

Durasi tes

Tentukan berapa lama uji beban Anda akan berjalan:

Naik

Waktu untuk mencapai target konkurensi. Beban secara bertahap meningkat dari 0 ke tingkat konkurensi yang dikonfigurasi selama periode ini.

Tahan Untuk

Durasi untuk mempertahankan beban target. Tes berlanjut pada konkurensi penuh untuk periode ini.

Langkah 4: Tinjau dan buat

Tinjau semua konfigurasi Anda sebelum membuat skenario pengujian. Verifikasi:

- Pengaturan umum (nama, deskripsi, jadwal).
- Konfigurasi skenario (jenis pengujian, titik akhir atau skrip).
- Bentuk lalu lintas (tugas, pengguna, durasi, wilayah).

Setelah meninjau, pilih Buat untuk menyimpan skenario pengujian Anda.

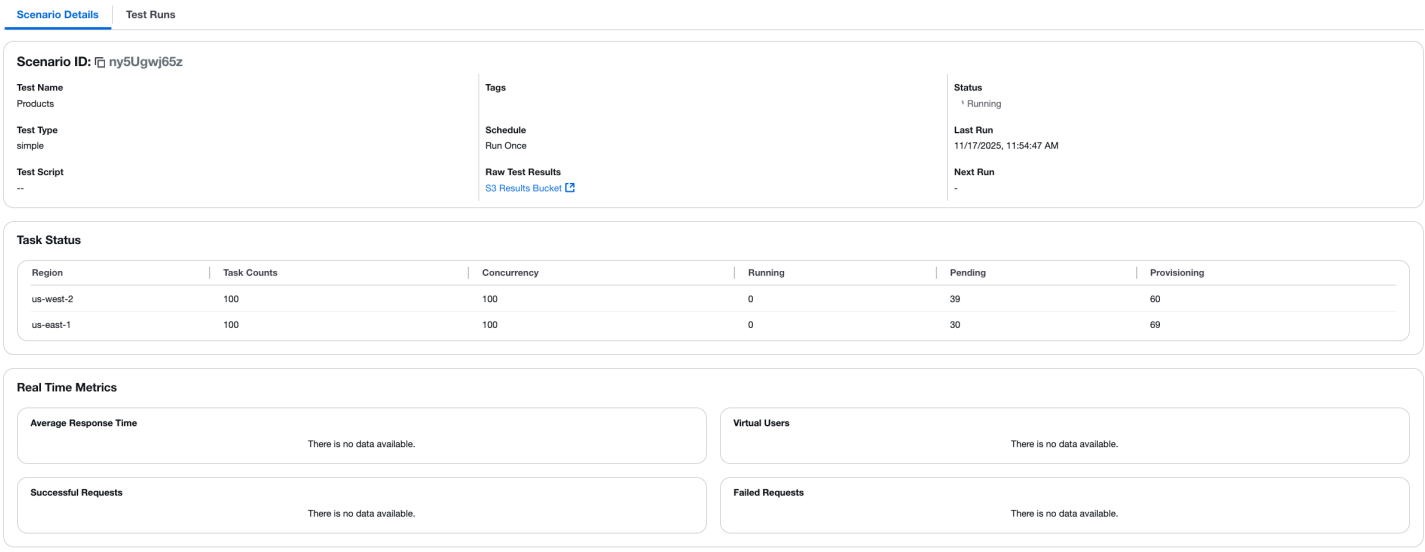
Mengelola skenario pengujian

Setelah membuat skenario pengujian, Anda dapat:

- Edit - Ubah konfigurasi pengujian. Kasus penggunaan umum meliputi:
  - Menyempurnakan bentuk lalu lintas untuk mencapai tingkat transaksi yang diinginkan.
- Salin - Gandakan skenario pengujian yang ada untuk membuat variasi. Kasus penggunaan umum meliputi:
  - Memperbarui titik akhir atau menambahkan headers/body parameter.
  - Menambahkan atau memodifikasi skrip pengujian.
- Hapus - Hapus skenario pengujian yang tidak lagi Anda butuhkan.

Jalankan skenario pengujian

Setelah membuat skenario pengujian, Anda dapat menjalankannya segera atau menjadwalkannya untuk dijalankan pada waktu tertentu di masa mendatang. Saat Anda menavigasi ke pengujian yang sedang berjalan, konsol akan menampilkan tab Detail Skenario dengan status tugas dan metrik waktu nyata.



Tampilan detail skenario

Tab Detail Skenario menampilkan informasi penting tentang pengujian Anda. Task staus tabel informasi real-time untuk setiap wilayah.

Tabel status tugas

Tabel Status Tugas menampilkan informasi real-time untuk setiap wilayah:

- Wilayah - Wilayah AWS tempat tugas berjalan
- Jumlah Tugas - Jumlah total tugas yang dikonfigurasi untuk wilayah tersebut
- Concurrency - Jumlah pengguna virtual per tugas
- Menjalankan - Jumlah tugas yang saat ini menjalankan tes
- Tertunda - Jumlah tugas yang menunggu untuk memulai
- Provisioning - Jumlah tugas yang sedang disediakan

Alur kerja eksekusi uji

Saat pengujian dimulai, alur kerja berikut terjadi:

1. Penyediaan tugas - Solusi menyediakan kontainer (tugas) di wilayah AWS yang ditentukan. Tugas muncul di kolom “Penyediaan”.
2. Startup tugas - Solusinya terus menyediakan tugas hingga jumlah tugas target tercapai di setiap wilayah. Tugas berpindah dari “Provisioning” ke “Pending” ke “Running”.

3. Pembuatan lalu lintas - Setelah solusi menyediakan semua tugas di suatu wilayah, mereka mulai mengirim lalu lintas ke titik akhir target Anda.
4. Eksekusi uji - Tes berjalan untuk durasi yang dikonfigurasi (ramp-up +hold time).
5. Penguraian hasil - Ketika pengujian berakhir, pekerjaan penguraian latar belakang mengumpulkan dan memproses hasil dari semua wilayah.

Status uji coba

Uji coba dapat memiliki status berikut:

- Dijadwalkan - Tes dijadwalkan untuk berjalan di masa depan.
- Menjalankan - Tes sedang berlangsung.
- Dibatalkan - Pengguna membatalkan uji coba yang sedang berlangsung.
- Errored - Uji coba mengalami kesalahan.
- Selesai - Uji coba selesai dengan sukses dan hasilnya siap.

Pemantauan dengan data langsung

Jika Anda mengaktifkan data langsung saat membuat skenario pengujian, Anda dapat melihat metrik waktu nyata saat pengujian sedang berjalan. Bagian Metrik Waktu Nyata menampilkan empat grafik yang diperbarui terus menerus saat pengujian berlangsung, dengan data dikumpulkan pada interval satu detik.

Deskripsi grafik

Waktu Respons Rata-rata

Menampilkan waktu respons rata-rata dalam hitungan detik untuk permintaan yang diproses oleh setiap wilayah. Sumbu Y menunjukkan waktu respons dalam hitungan detik, dan sumbu X menunjukkan waktu dalam sehari. Setiap daerah diwakili oleh warna yang berbeda dalam legenda.

Pengguna Virtual

Menunjukkan jumlah pengguna virtual bersamaan yang secara aktif menghasilkan beban di setiap wilayah. Grafik menampilkan bagaimana pengguna virtual meningkat selama pengujian dan mempertahankan tingkat konkurensi target.

Permintaan Berhasil

Menampilkan jumlah kumulatif permintaan yang berhasil dari waktu ke waktu untuk setiap wilayah. Grafik menunjukkan tingkat di mana permintaan yang berhasil sedang diproses.

Permintaan Gagal

Menampilkan jumlah kumulatif permintaan yang gagal dari waktu ke waktu untuk setiap wilayah. Hitungan rendah atau nol menunjukkan eksekusi tes yang sehat.

Visualisasi multi-wilayah

Saat menjalankan pengujian di beberapa wilayah, setiap grafik menampilkan data untuk semua wilayah secara bersamaan. Legenda di bagian bawah setiap grafik mengidentifikasi warna mana yang mewakili setiap wilayah (misalnya, us-west-2 dan us-east-1).

Implementasi teknis

Grup CloudWatch log untuk tugas Fargate berisi filter langganan yang menangkap hasil pengujian. Saat pola terdeteksi, fungsi Lambda menyusun data dan menerbitkannya ke topik AWS IoT Core. Konsol web berlangganan topik ini dan menampilkan metrik secara real-time.

> **Note**
>
> Data langsung bersifat sementara dan hanya tersedia saat pengujian sedang berjalan. Konsol web bertahan maksimal 5.000 titik data, setelah itu data tertua diganti dengan yang terbaru. Jika halaman diperbarui, grafik akan kosong dan mulai dari titik data berikutnya yang tersedia. Setelah tes selesai, solusinya menyimpan data hasil di DynamoDB dan Amazon S3. Jika belum ada data yang tersedia, grafik akan menampilkan “Tidak ada data yang tersedia.”

Membatalkan tes

Anda dapat membatalkan pengujian yang sedang berjalan dari konsol web. Saat Anda membatalkan pengujian, alur kerja berikut akan terjadi:

1. Permintaan pembatalan dikirim ke API `microservices`
2. `microservices`API memanggil fungsi `task-canceler` Lambda yang menghentikan semua tugas yang diluncurkan saat ini
3. Jika fungsi `task-runner` Lambda terus berjalan setelah panggilan pembatalan awal, tugas dapat terus diluncurkan sebentar
4. Setelah fungsi `task-runner` Lambda selesai, AWS Step Functions melanjutkan ke `Cancel Test` langkah, yang menjalankan fungsi `task-canceler` Lambda lagi untuk menghentikan tugas yang tersisa

> **Note**
>
> Pengujian yang dibatalkan membutuhkan waktu untuk menyelesaikan proses shutdown karena solusi menghentikan semua kontainer. Status pengujian akan berubah menjadi “Dibatalkan” setelah semua sumber daya dibersihkan.

Jelajahi hasil tes

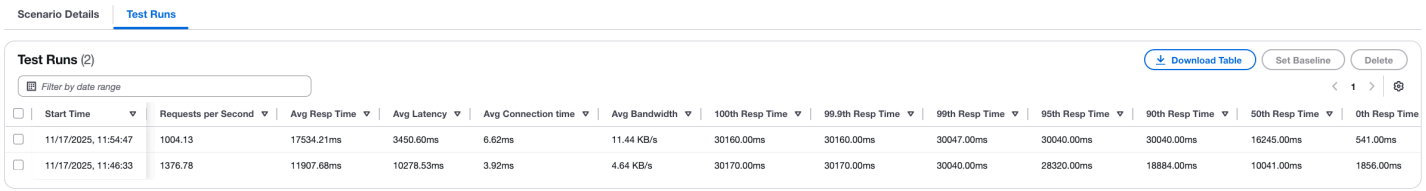
Setelah pekerjaan parsing selesai, hasil tes tersedia untuk analisis. Solusi ini menyediakan metrik dan alat yang komprehensif untuk membantu Anda memahami kinerja aplikasi yang sedang dimuat.

Metrik ringkasan uji coba

Saat pengujian selesai, solusi akan menghasilkan ringkasan yang menyertakan metrik berikut:

- Waktu respons rata-rata - Waktu respons rata-rata, dalam hitungan detik, untuk semua permintaan yang dihasilkan oleh tes.
- Latensi rata-rata - Latensi rata-rata, dalam hitungan detik, untuk semua permintaan yang dihasilkan oleh pengujian.
- Waktu koneksi rata-rata - Waktu rata-rata, dalam hitungan detik, diperlukan untuk terhubung ke host untuk semua permintaan.
- Bandwidth rata-rata - Bandwidth rata-rata untuk semua permintaan yang dihasilkan oleh tes.
- Jumlah total - Jumlah total permintaan.
- Hitungan keberhasilan - Jumlah total permintaan yang berhasil.
- Jumlah kesalahan - Jumlah total kesalahan.
- Permintaan per detik - Permintaan rata-rata per detik untuk semua permintaan yang dihasilkan oleh pengujian.
- Persentil - Persentil waktu respons termasuk p50 (median), p90, p95, dan p99, menunjukkan distribusi waktu respons di semua permintaan.

Tabel uji coba



Tabel uji coba menampilkan semua uji historis yang berjalan untuk sebuah skenario. Anda dapat:

- Lihat metrik ringkasan untuk setiap pengujian yang dijalankan.
- Tetapkan uji coba dasar untuk perbandingan kinerja.
- Unduh tabel sebagai file CSV.
- Alihkan kolom untuk menyesuaikan tampilan Anda.
- Pilih uji coba untuk melihat hasil terperinci.

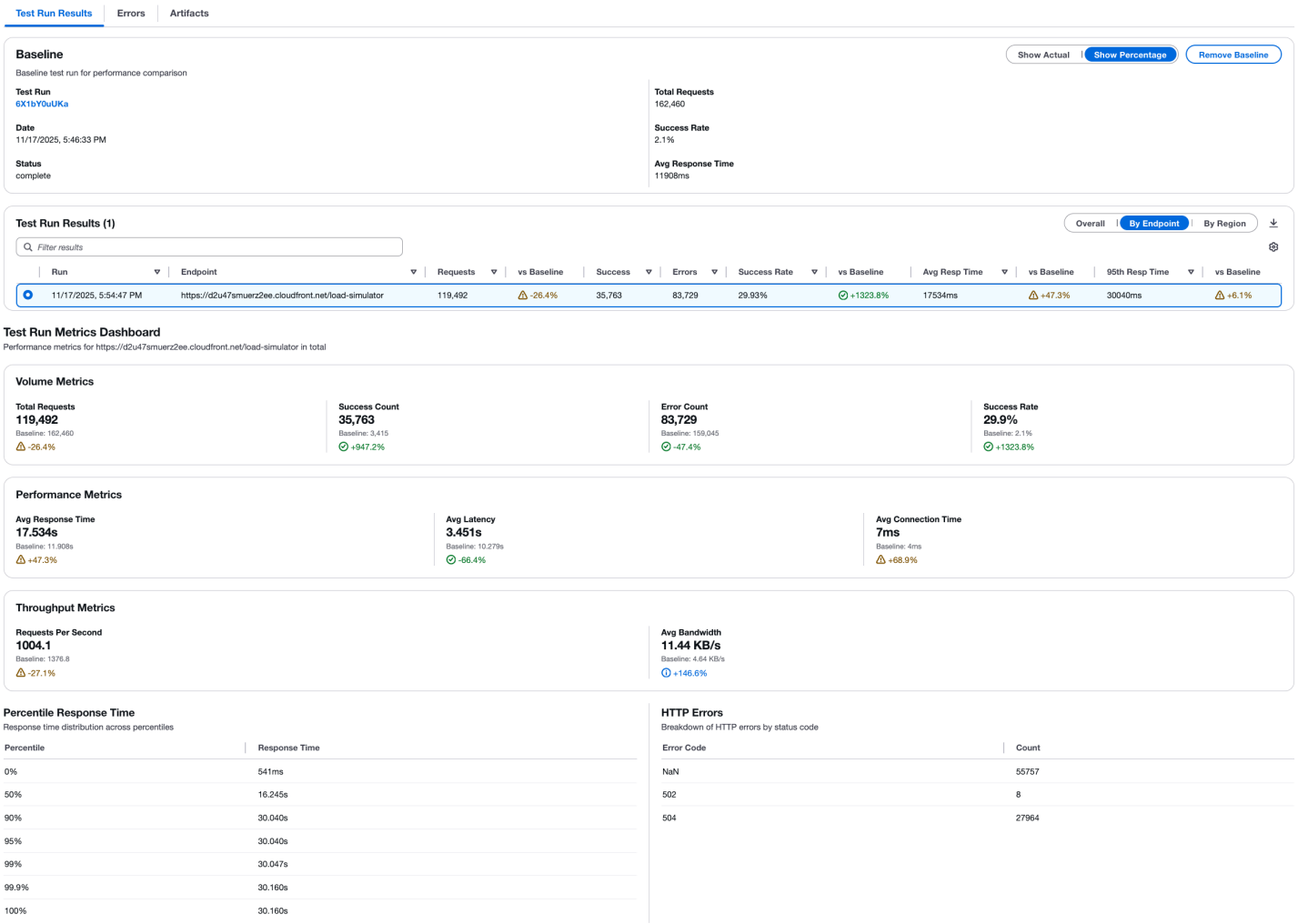
Perbandingan dasar

Anda dapat menetapkan uji coba sebagai baseline untuk membandingkan uji coba masa depan terhadapnya. Ketika baseline ditetapkan:

- Tabel uji coba menunjukkan perbedaan persentase (+/-%) dibandingkan dengan baseline untuk setiap metrik.
- Indikator dasar membantu Anda mengidentifikasi peningkatan atau regresi kinerja dengan cepat.
- Anda dapat mengubah atau menghapus baseline kapan saja.

Hasil tes terperinci

Memilih uji coba membuka tampilan hasil terperinci dengan tiga tab: Hasil Uji Jalankan, Kesalahan, dan Artefak.



Informasi dasar

Jika uji coba dasar diatur, itu akan ditampilkan di bagian atas halaman. Anda dapat memilih Tampilkan Aktual, Tampilkan Persentase, atau Hapus Garis Dasar untuk mengontrol bagaimana perbandingan dasar ditampilkan.

Tabel Hasil Uji Jalankan

Tabel hasil memberikan metrik terperinci dengan fitur-fitur berikut:

Tampilan dimensi

Beralih di antara tiga tampilan menggunakan tombol dimensi:

- Keseluruhan - Hasil agregat di semua titik akhir dan wilayah
- By Endpoint - Hasil dipecah berdasarkan titik akhir individu

- Berdasarkan Wilayah - Hasil dipecah menurut wilayah AWS

Tombol aksi

- Tampilkan Aktual - Menampilkan nilai metrik aktual
- Tampilkan Persentase - Tampilkan perbedaan persentase dari baseline
- Hapus Baseline - Hapus perbandingan dasar

Ekspor dan kustomisasi data

- Unduh tabel hasil sebagai file CSV
- Beralih kolom untuk menyesuaikan tampilan Anda
- Filter dan urutkan data untuk fokus pada metrik tertentu
- Filter dan urutkan data untuk fokus pada metrik tertentu.

Tab kesalahan

Tab kesalahan menyediakan analisis kesalahan terperinci:

- Lihat jumlah kesalahan berdasarkan jenis.
- Lihat kesalahan yang dikumpulkan berdasarkan pengujian keseluruhan atau titik akhir.
- Identifikasi pola dalam permintaan yang gagal.
- Memecahkan masalah dengan titik akhir atau wilayah tertentu.

Tab artefak

Tab artefak memungkinkan Anda mengakses semua file yang dihasilkan selama uji coba:

- Lihat artefak individu (log, file hasil).
- Unduh artefak khusus untuk analisis offline.
- Unduh semua artefak uji coba sebagai arsip tunggal.

Struktur hasil S3

Dalam versi 4.0, struktur hasil S3 telah berubah untuk meningkatkan organisasi:

- Struktur baru -`scenario-id/test-run-id/results-files`.
- Struktur lama - Pengujian dijalankan sebelum versi 4.0 menampilkan semua file hasil pada tingkat ID skenario.

> **Note**
>
> Hasil pengujian ditampilkan di konsol. Anda juga dapat mengakses hasil pengujian mentah langsung di bucket Amazon S3 di bawah folder. `Results` Untuk informasi selengkapnya tentang hasil tes Taurus, lihat Menghasilkan Laporan Uji di Panduan Pengguna Taurus.

Integrasi server MCP

Jika Anda menerapkan komponen server MCP opsional selama penerapan solusi, Anda dapat mengintegrasikan solusi Pengujian Beban Terdistribusi dengan alat pengembangan AI yang mendukung Protokol Konteks Model. Server MCP menyediakan akses terprogram untuk mengambil, mengelola, dan menganalisis tes beban melalui asisten AI.

Pelanggan dapat terhubung ke DLT MCP Server menggunakan klien pilihan mereka (Amazon Q, Claude, dll.), Yang masing-masing memiliki instruksi konfigurasi yang sedikit berbeda. Bagian ini menyediakan petunjuk penyiapan untuk MCP Inspector, Amazon Q CLI, Cline, dan Amazon Q Suite.

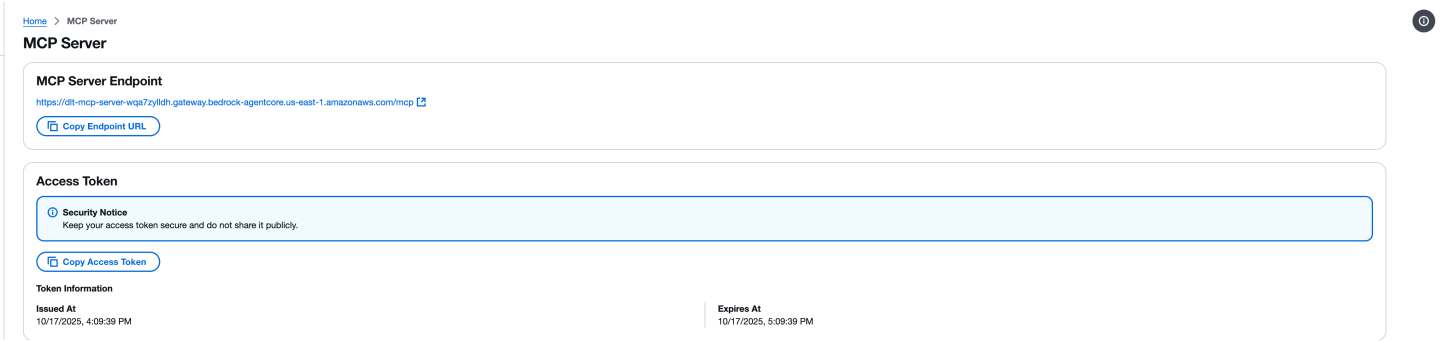
Langkah 1: Dapatkan titik akhir MCP dan token akses

Sebelum mengonfigurasi klien MCP apa pun, Anda perlu mengambil titik akhir server MCP dan token akses dari konsol web DLT.

1. Arahkan ke halaman MCP Server di konsol web Distributed Load Testing.
2. Temukan bagian Endpoint Server MCP.
3. Salin URL titik akhir menggunakan tombol Salin URL Titik Akhir. URL endpoint mengikuti format: `https://{gateway-id}.gateway.bedrock-agentcore.{region}.amazonaws.com/mcp`
4. Temukan bagian Token Akses.
5. Salin token akses menggunakan tombol Copy Access Token.

Important

Jaga token akses Anda tetap aman dan jangan membagikannya secara publik. Token menyediakan akses hanya-baca ke solusi Pengujian Beban Terdistribusi Anda melalui antarmuka MCP.



Langkah 2: Uji dengan MCP Inspector

Model Context Protocol menawarkan MCP Inspector, alat untuk langsung terhubung ke server MCP dan memanggil alat. Ini menyediakan UI yang nyaman dan permintaan jaringan sampel untuk menguji koneksi server MCP Anda sebelum mengonfigurasi klien AI.

Note

MCP Inspector membutuhkan versi 0.17 atau yang lebih baru. Semua permintaan juga dapat dibuat dengan JSON RPC secara langsung, tetapi MCP Inspector menyediakan antarmuka yang lebih ramah pengguna.

Instal dan luncurkan MCP Inspector

1. Instal npm jika perlu.
2. Jalankan perintah berikut untuk meluncurkan MCP Inspector:

```
npx @modelcontextprotocol/inspector
```

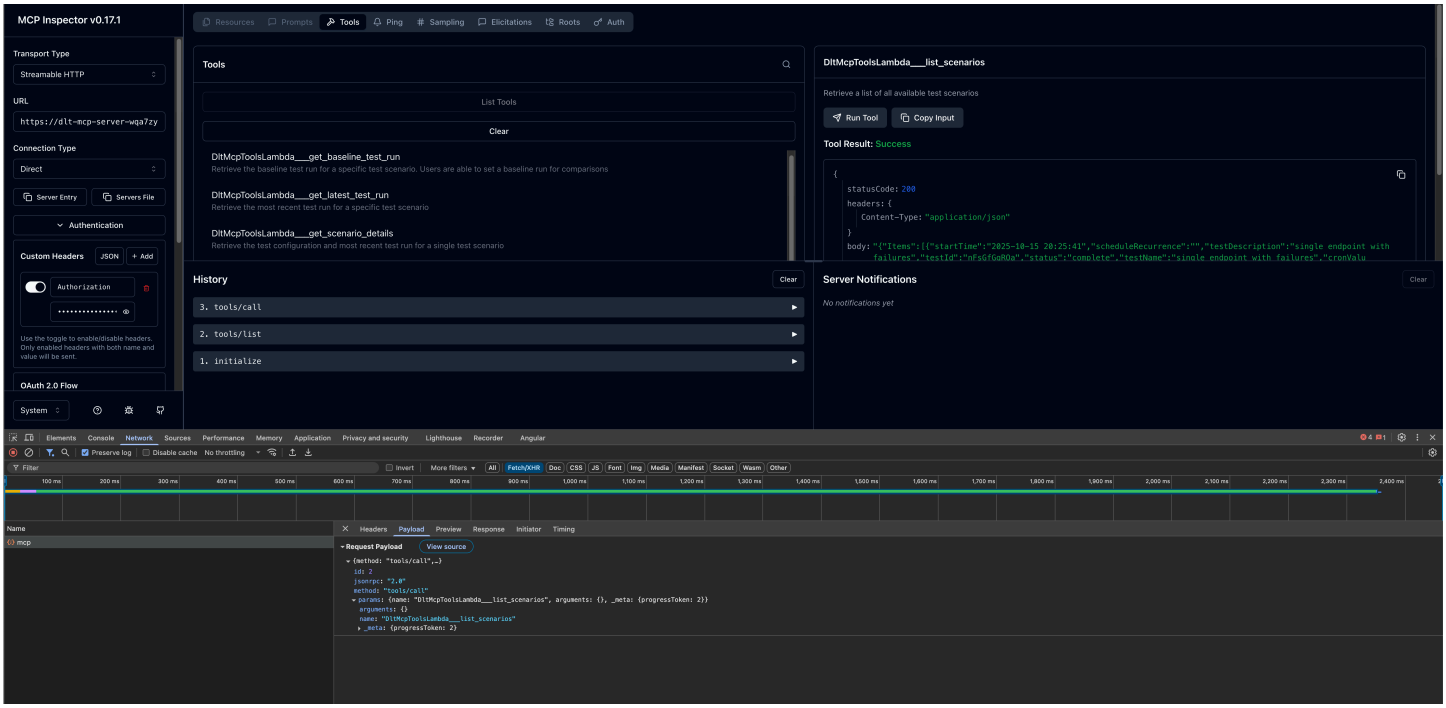
Konfigurasikan koneksi

1. Di antarmuka MCP Inspector, masukkan URL Endpoint Server MCP Anda.
2. Tambahkan header Otorisasi dengan token akses Anda.
3. Klik Connect untuk membuat koneksi.

Memanggil alat

Setelah terhubung, Anda dapat menguji alat MCP yang tersedia:

1. Jelajahi daftar alat yang tersedia di panel kiri.
2. Pilih alat (misalnya,`list_scenarios`).
3. Berikan parameter yang diperlukan.
4. Klik Memanggil untuk menjalankan alat dan melihat responsnya.



Langkah 3: Konfigurasikan klien pengembangan AI

Setelah memverifikasi koneksi server MCP Anda dengan MCP Inspector, Anda dapat mengonfigurasi klien pengembangan AI pilihan Anda.

Amazon Q CLI

Amazon Q CLI menyediakan akses baris perintah ke pengembangan berbantuan AI dengan integrasi server MCP.

Langkah-langkah konfigurasi

1. Edit file `mcp.json` konfigurasi. Untuk informasi selengkapnya tentang lokasi file konfigurasi, lihat Mengonfigurasi server MCP jarak jauh di Panduan Pengguna Pengembang Amazon Q.
2. Tambahkan konfigurasi server DLT MCP Anda:

```
{
  "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"
      }
    }
  }
}
```

Verifikasi konfigurasi

1. Di terminal, ketik q untuk meluncurkan Amazon Q CLI.
2. Ketik `/mcp` untuk melihat semua server MCP yang tersedia.
3. Ketik `/tools` untuk melihat alat yang tersedia yang disediakan oleh `dlt-mcp` dan server MCP yang dikonfigurasi lainnya.
4. Verifikasi bahwa `dlt-mcp` berhasil menginisialisasi.

Cline

Cline adalah asisten pengkodean AI yang mendukung integrasi server MCP.

Langkah-langkah konfigurasi

1. Di Cline, navigasikan ke Kelola Server MCP> Konfigurasi> Konfigurasikan Server MCP.
2. Perbarui `cline_mcp_settings.json` file:

```
{
  "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. Simpan file konfigurasi.
4. Mulai ulang Cline untuk menerapkan perubahan.

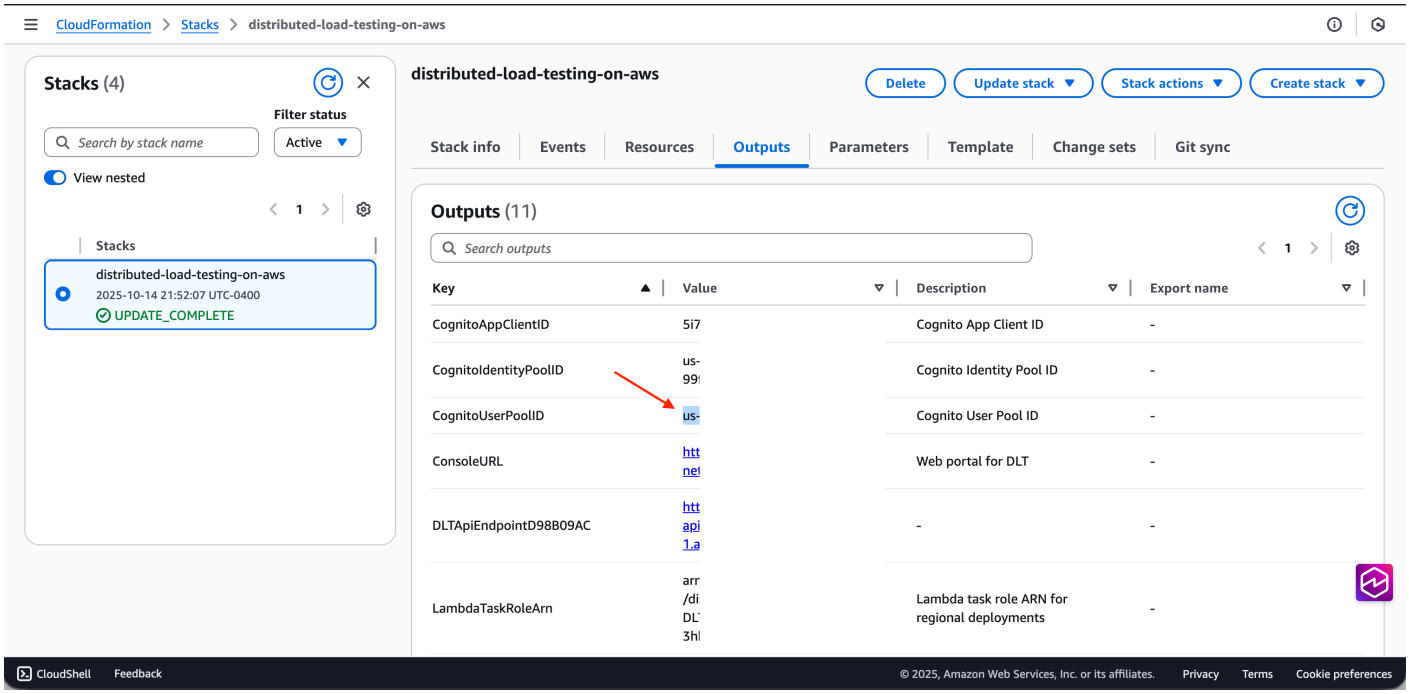
Suite Amazon Q

Amazon Q Suite menyediakan platform asisten AI yang komprehensif dengan dukungan untuk tindakan server MCP.

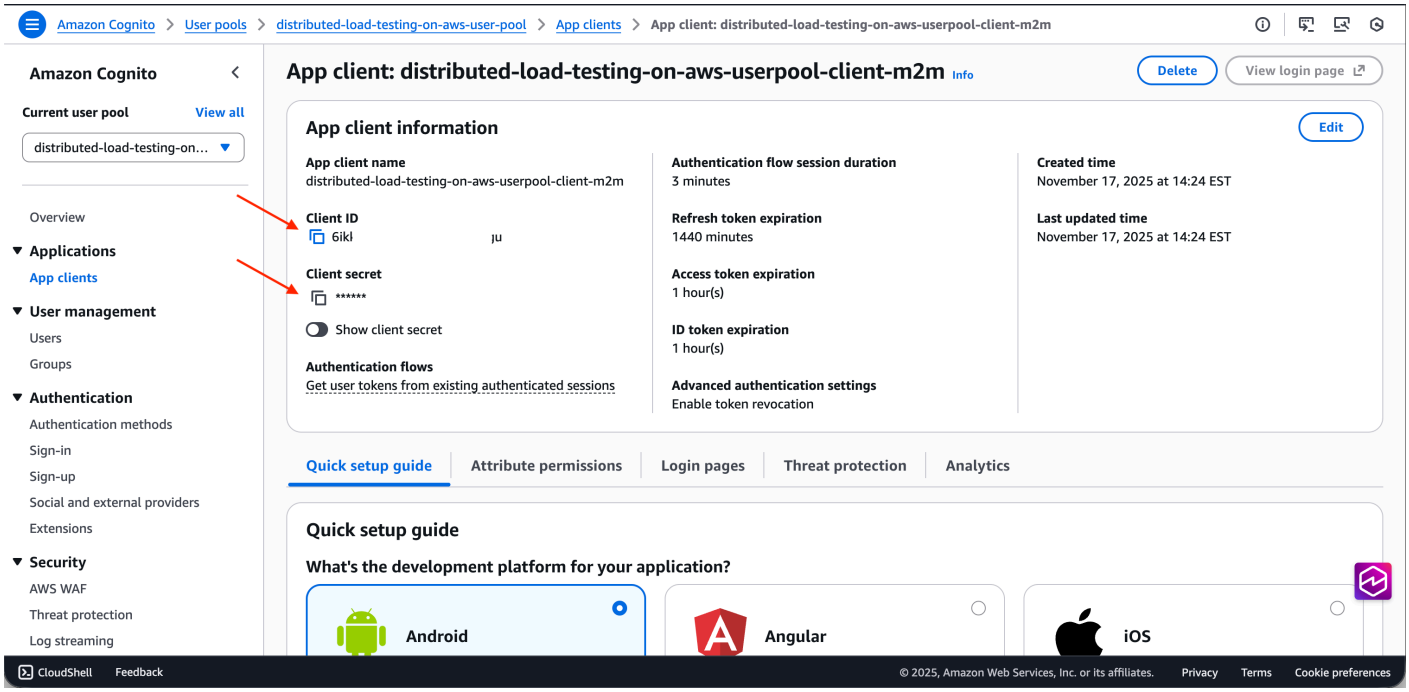
Prasyarat

Sebelum mengonfigurasi server MCP di Amazon Q Suite, Anda perlu mengambil OAuth kredensil dari kumpulan pengguna Cognito penerapan DLT:

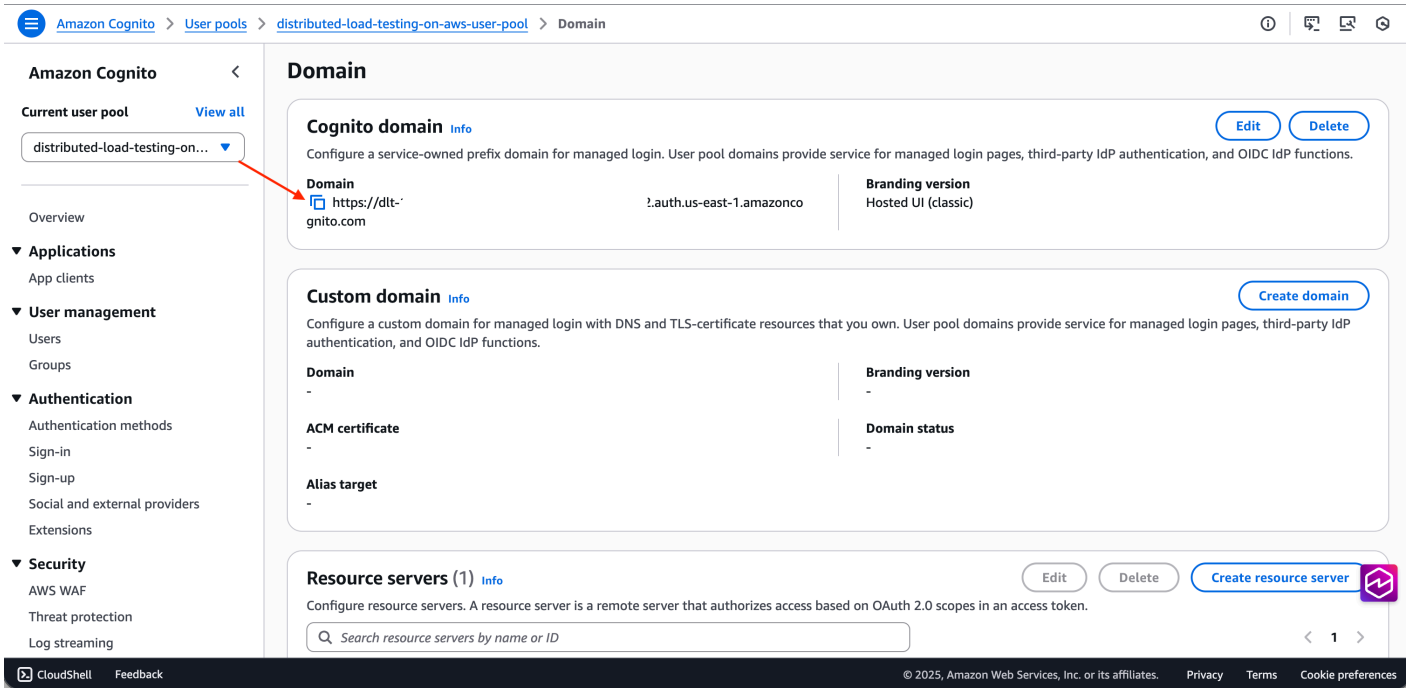
1. Arahkan ke CloudFormation konsol AWS.
2. Pilih tumpukan Pengujian Beban Terdistribusi.
3. Di tab Output, cari dan salin ID Kumpulan Pengguna Cognito yang terkait dengan penerapan DLT Anda.



4. Arahkan ke konsol Amazon Cognito.
5. Pilih kumpulan pengguna menggunakan User Pool ID dari CloudFormation output.
6. Di navigasi kiri, pilih Integrasi aplikasi > Klien aplikasi.



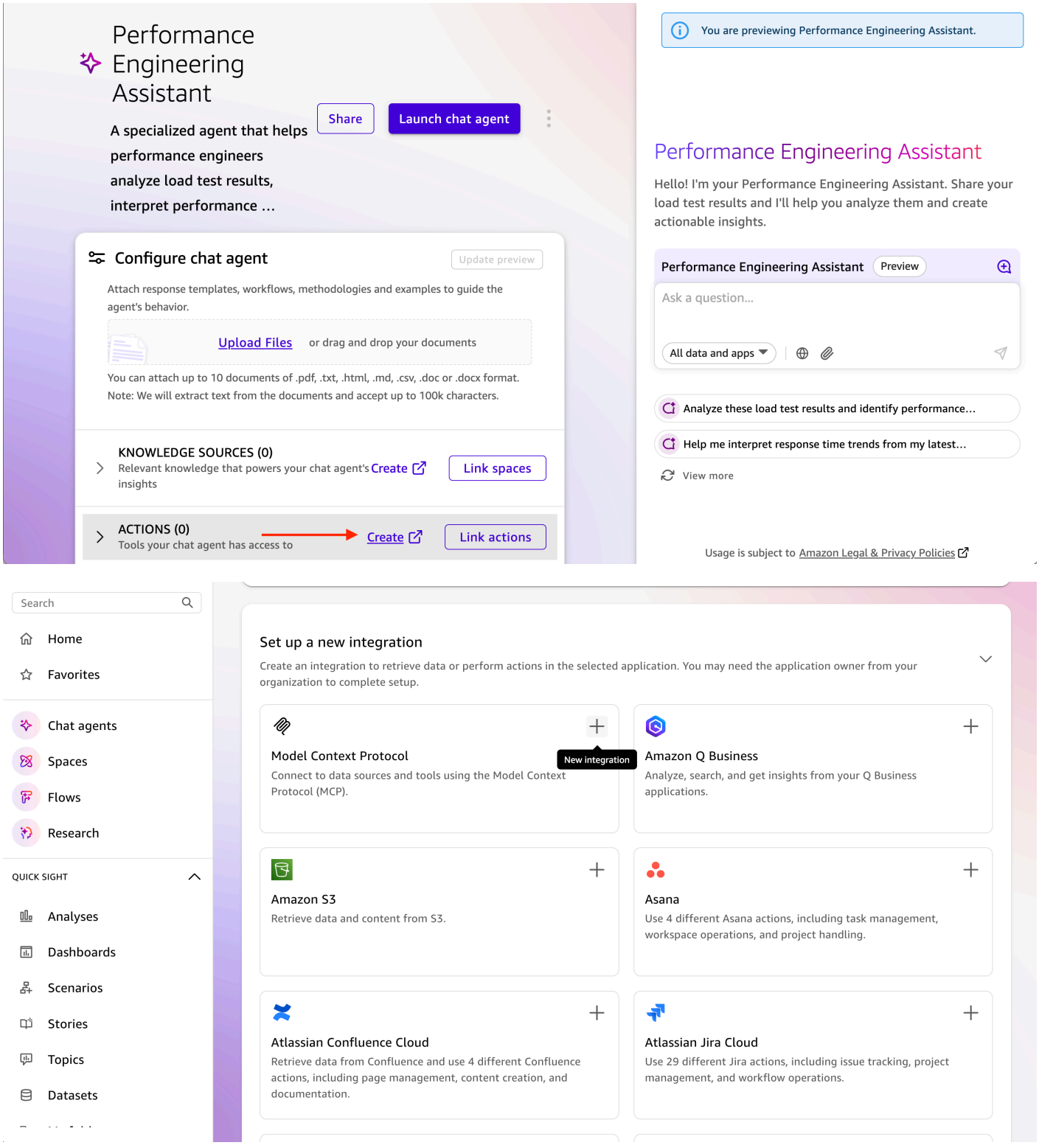
7. Temukan klien aplikasi dengan nama yang diakhiri dengan `m2m` (machine-to-machine).
8. Salin ID Klien dan rahasia Klien.
9. Dapatkan domain kumpulan pengguna dari tab Domain.



10. Buat URL titik akhir token dengan menambahkan `/oauth2/token` ke akhir domain.

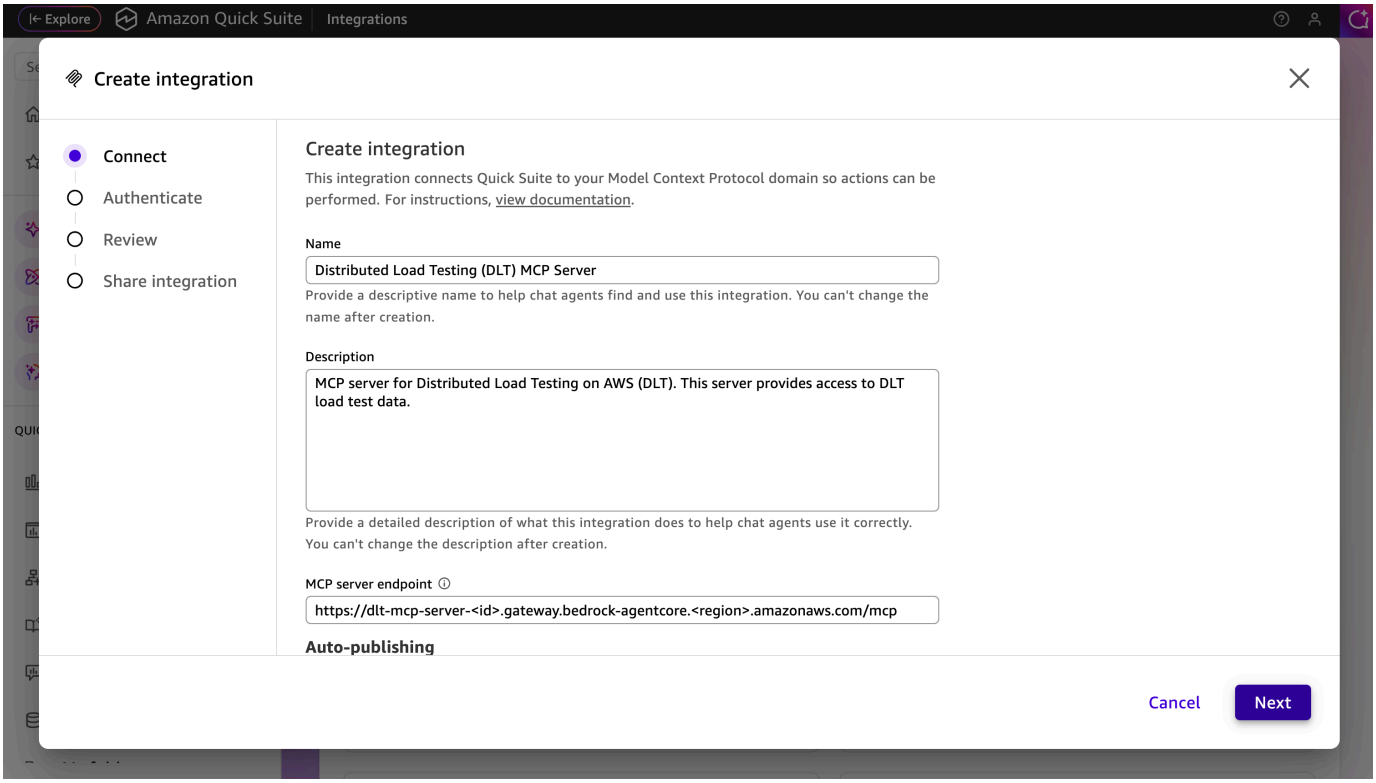
Langkah-langkah konfigurasi

1. Di Amazon Q Suite, buat agen baru atau pilih agen yang sudah ada.
2. Tambahkan prompt agen yang menjelaskan cara berinteraksi dengan server DLT MCP.
3. Tambahkan tindakan baru dan pilih tindakan server MCP.

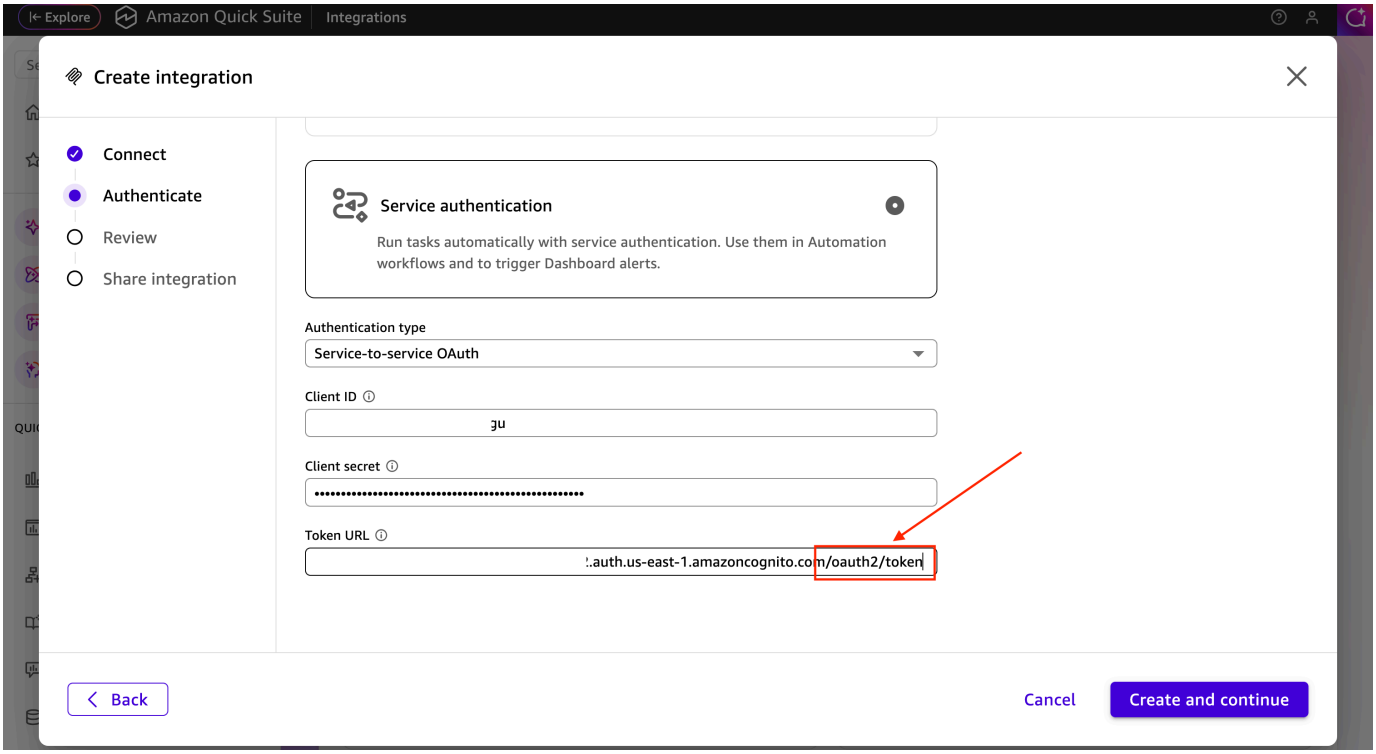


4. Konfigurasikan detail server MCP:

- URL Server MCP: Titik akhir MCP DLT Anda



- Jenis Otentikasi: Otentikasi berbasis layanan
- Titik Akhir Token: URL titik akhir token Cognito Anda
- ID Klien: ID klien dari klien aplikasi m2m
- Rahasia Klien: Rahasia klien dari klien aplikasi m2m



5. Simpan konfigurasi tindakan server MCP.
6. Tambahkan tindakan server MCP baru ke agen Anda.

Luncurkan dan uji agen

1. Luncurkan agen di Amazon Q Suite.
2. Mulailah percakapan dengan agen menggunakan petunjuk bahasa alami.
3. Agen akan menggunakan alat MCP untuk mengambil dan menganalisis data pengujian beban Anda.

Contoh petunjuk

Contoh berikut menunjukkan cara berinteraksi dengan asisten AI Anda untuk menganalisis data pengujian beban melalui antarmuka MCP. Sesuaikan pengujian IDs, rentang tanggal, dan kriteria agar sesuai dengan kebutuhan pengujian spesifik Anda.

Untuk informasi rinci tentang alat MCP yang tersedia dan parameternya, lihat spesifikasi alat MCP di Panduan Pengembang.

Kueri hasil tes sederhana

Interaksi bahasa alami dengan MCP Server bisa sesederhana `Show me the load tests that have completed in the last 24 hours with their associated completion status` atau bisa lebih deskriptif seperti

```
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.
```

Analisis kinerja interaktif dengan pengungkapan progresif

```
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.
```

Validasi kesiapan produksi

```
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]%.
```

Analisis tren kinerja

```
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.
```

Memecahkan masalah tes yang gagal

```
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].
```

Panduan developer

Bagian ini menyediakan kode sumber untuk solusi dan penyesuaian tambahan.

Kode sumber

Kunjungi GitHub repositori kami untuk mengunduh templat dan skrip untuk solusi ini, dan untuk berbagi penyesuaian Anda dengan orang lain.

Maintenance

Solusi ini menggunakan gambar Docker dengan versi tetap yang sesuai dengan setiap rilis solusi. Secara default, pembaruan otomatis dinonaktifkan, memberi Anda kendali penuh atas kapan dan pembaruan versi mana yang diterapkan pada penerapan Anda. Tim AWS Solutions menggunakan Amazon ECR Enhanced Scanning untuk mendeteksi Kerentanan Umum dan Eksposur (CVEs) dalam gambar dasar dan paket yang diinstal. Jika memungkinkan, tim menerbitkan gambar yang ditambal dengan tag versi yang sama untuk diselesaikan CVEs tanpa merusak kompatibilitas dengan versi solusi yang dirilis.

Ketika gambar ditambal pada versi minor yang sama, tag stabil diperbarui secara otomatis, dan tag gambar tambahan dibuat dalam format`<solution-version>_<date-of-fix>`. Jika versi mayor atau minor dirilis, Anda harus melakukan pembaruan tumpukan penuh untuk mendapatkan versi gambar terbaru, karena tag stabil ditambahkan agar sesuai dengan versi solusi.

Jika Anda memilih pembaruan otomatis selama penerapan, perubahan pada gambar, termasuk tambalan CVE dan perbaikan bug minor, secara otomatis diterapkan hingga rilis minor terbaru yang cocok.

Versi

Secara default, solusi ini digunakan dengan pembaruan otomatis dinonaktifkan. Ini berarti versi gambar kontainer dikunci ke versi spesifik yang cocok dengan versi solusi yang Anda gunakan, memberi Anda kontrol penuh atas pembaruan versi.

Jika Anda memilih untuk mengaktifkan pembaruan otomatis dengan memilih Ya selama CloudFormation penerapan, solusi akan secara otomatis menerima patch keamanan dan perbaikan

bug minor yang tidak rusak hingga versi minor terbaru yang cocok. Misalnya, jika Anda menerapkan versi 4.0.0 dengan pembaruan otomatis diaktifkan, Anda akan menerima pembaruan hingga 4.0.x, tetapi tidak 4.1.0 atau lebih tinggi.

Untuk mengontrol versi gambar kontainer secara manual, Anda dapat mengedit definisi tugas untuk menentukan versi gambar tertentu menggunakan format versi yang ditandai. Ini memungkinkan Anda untuk menyematkan ke versi gambar tertentu terlepas dari pengaturan pembaruan otomatis.

Kustomisasi gambar kontainer

Solusi ini menggunakan repositori image Amazon Elastic Container Registry (Amazon ECR) publik yang dikelola oleh AWS untuk menyimpan gambar yang digunakan untuk menjalankan pengujian yang dikonfigurasi. Jika Anda ingin menyesuaikan gambar kontainer, Anda dapat membangun kembali dan mendorong gambar ke dalam repositori gambar ECR di akun AWS Anda sendiri.

Jika Anda ingin menyesuaikan solusi ini, Anda dapat menggunakan gambar kontainer default atau, edit wadah ini agar sesuai dengan kebutuhan Anda. Jika Anda menyesuaikan solusinya, gunakan contoh kode berikut untuk mendeklarasikan variabel lingkungan sebelum membuat solusi khusus Anda.

```
#!/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
```

Jika Anda memilih untuk menyesuaikan gambar kontainer, Anda dapat menghostingnya di repositori gambar pribadi, atau repositori gambar publik di akun AWS Anda. Sumber daya gambar ada di `deployment/ecr/distributed-load-testing-on-aws-load-tester` direktori, yang terletak di basis kode.

Anda dapat membangun dan mendorong gambar ke tujuan host.

- Untuk repositori dan gambar Amazon ECR pribadi, lihat repositori pribadi Amazon ECR dan gambar pribadi di Panduan Pengguna Amazon ECR.

- Untuk repositori dan gambar ECR Amazon publik, lihat repositori publik Amazon ECR dan gambar publik di Panduan Pengguna Publik Amazon ECR.

Setelah Anda membuat gambar Anda sendiri, Anda dapat mendeklarasikan variabel lingkungan berikut sebelum membangun solusi khusus Anda.

```
#!/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
```

Contoh berikut menunjukkan file kontainer.

```
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"]
```

Selain file kontainer, direktori berisi skrip bash berikut yang mengunduh konfigurasi pengujian dari Amazon S3 sebelum menjalankan Taurus/Blazemeter program.

```
#!/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
```

Selain Dockerfile dan skrip bash, dua skrip Python juga disertakan dalam direktori. Setiap tugas menjalankan skrip Python dari dalam skrip bash. Tugas pekerja menjalankan `ecslistener.py` skrip, sedangkan tugas pemimpin akan menjalankan `ecscontroller.py` skrip. `ecslistener.py`Skrip membuat soket pada port 50000 dan menunggu pesan.

`ecscontroller.py`Skrip terhubung ke soket dan mengirimkan pesan uji awal ke tugas pekerja, yang memungkinkan mereka untuk memulai secara bersamaan.

API pengujian beban terdistribusi

Solusi pengujian beban ini membantu Anda mengekspos data hasil pengujian dengan cara yang aman. API bertindak sebagai “pintu depan” untuk akses ke data pengujian yang disimpan di Amazon DynamoDB. Anda juga dapat menggunakan APIs untuk mengakses fungsionalitas tambahan apa pun yang Anda buat ke dalam solusi.

Solusi ini menggunakan kumpulan pengguna Amazon Cognito yang terintegrasi dengan Amazon API Gateway untuk identifikasi dan otorisasi. Ketika kumpulan pengguna digunakan dengan API, klien hanya diperbolehkan memanggil metode yang diaktifkan kumpulan pengguna setelah mereka memberikan token identitas yang valid.

Untuk informasi selengkapnya tentang menjalankan pengujian secara langsung melalui API, lihat Permintaan Penandatanganan di dokumentasi Referensi API REST API Amazon API Gateway.

Operasi berikut tersedia di API solusi.

> **Note**
>
> Untuk informasi lebih lanjut tentang `testScenario` dan parameter lainnya, lihat skenario dan contoh payload di GitHub repositori.

Info Tumpukan

- DAPATKAN /tumpukan-info

Skenario

- DAPATKAN /skenario
- POST/skenario
- PILIHAN/skenario
- DAPATKAN /scenarios/ {teSid}
- POSTING /scenarios/ {TESid}

- HAPUS /scenarios/ {teSid}
- PILIHAN /scenarios/ {TESid}

Uji Berjalan

- DAPATKAN /scenarios/ {teSTid} /testruns
- DAPATKAN /scenarios/ {teSid} /testruns/ {} testRunId
- HAPUS /scenarios/ {teSTid} /testruns/ {} testRunId

Baseline

- DAPATKAN /scenarios/ {teSid} /baseline
- PUT /scenarios/ {teSid} /baseline
- HAPUS /scenarios/ {teSid} /baseline

Tugas

- DAPATKAN /tugas
- PILIHAN/tugas

Daerah

- DAPATKAN /wilayah
- PILIHAN/wilayah

DAPATKAN /tumpukan-info

Deskripsi

`GET /stack-info`Operasi mengambil informasi tentang tumpukan yang digunakan termasuk waktu pembuatan, wilayah, dan versi. Endpoint ini digunakan oleh front-end.

Respons

200 - Sukses

Nama	Deskripsi
`created_time`	Stempel waktu ISO 8601 saat tumpukan dibuat (misalnya,) `2025-09-09T19:40:22Z`
`region`	Wilayah AWS tempat tumpukan digunakan (misalnya,`us-east-1`)
`version`	Versi solusi yang diterapkan (misalnya ,`v4.0.0`)

Tanggapan Kesalahan

- `403`- Terlarang: Izin tidak cukup untuk mengakses informasi tumpukan
- `404`- Tidak ditemukan: Informasi tumpukan tidak tersedia
- `500`- Kesalahan server internal

DAPATKAN /skenario

Deskripsi

`GET /scenarios`Operasi ini memungkinkan Anda untuk mengambil daftar skenario pengujian.

Respons

Nama	Deskripsi
`data`	Daftar skenario termasuk ID, nama, deskripsi , status, waktu berjalan, tag, total run, dan run terakhir untuk setiap pengujian

POST/skenario

Deskripsi

`POST /scenarios`Operasi ini memungkinkan Anda untuk membuat atau menjadwalkan skenario pengujian.

Isi permintaan

Nama	Deskripsi
`testName`	Nama tes
`testDescription`	Deskripsi tes
`testTaskConfigs`	Objek yang menentukan `concurrency` (jumlah paralel berjalan), `taskCount` (jumlah tugas yang diperlukan untuk menjalankan tes), dan `region` untuk skenario
`testScenario`	Definisi tes termasuk konkurensi, waktu pengujian, host, dan metode untuk pengujian
`testType`	Jenis tes (misalnya,`simple`,`jmeter`)
`fileType`	Jenis file upload (misalnya,`none`,`script`,`zip`)
`tags`	Sebuah array string untuk mengkategorikan tes. Bidang opsional dengan panjang maksimum 5 (misalnya,`["blue", "3.0", "critical"]`)
`scheduleDate`	Tanggal untuk menjalankan tes. Hanya disediakan jika menjadwalkan tes (misalnya ,`2021-02-28`)
`scheduleTime`	Waktu untuk menjalankan tes. Hanya disediakan jika menjadwalkan tes (misalnya,`21:07`)

Nama	Deskripsi
`scheduleStep`	Langkah dalam proses jadwal. Hanya disediakan jika menjadwalkan tes berulang. (Langkah-langkah yang tersedia termasuk `create` dan`start`)
`cronvalue`	Nilai cron untuk menyesuaikan penjadwalan berulang. Jika digunakan, hilangkan scheduleDate dan ScheduleTime.
`cronExpiryDate`	Tanggal yang diperlukan sehingga cron kedaluwarsa dan tidak berjalan tanpa batas waktu.
`recurrence`	Terulangnya tes terjadwal. Hanya disediakan jika menjadwalkan tes berulang (misalnya,,, `daily` `weekly`biweekly, atau) `monthly`

Respons

Nama	Deskripsi
`testId`	ID unik dari tes
`testName`	Nama tes
`status`	Status tes

PILIHAN/skenario

Deskripsi

`OPTIONS /scenarios`Operasi ini memberikan respons untuk permintaan dengan header respons CORS yang benar.

Respons

Nama	Deskripsi
`testId`	ID unik dari tes
`testName`	Nama tes
`status`	Status tes

DAPATKAN /scenarios/ {teSid}

Deskripsi

`GET /scenarios/{testId}`Operasi ini memungkinkan Anda untuk mengambil rincian skenario pengujian tertentu.

Permintaan parameter

`testId`

- ID unik dari tes

  Tipe: String

  Diperlukan: Ya

`latest`

- Parameter kueri untuk mengembalikan hanya uji coba terbaru. Default adalah `true`

  Tipe: Boolean

  Wajib: Tidak

`history`

- Parameter kueri untuk menyertakan riwayat uji coba dalam respons. Default-nya adalah `true`. Setel `false` untuk mengecualikan riwayat

  Tipe: Boolean

  Wajib: Tidak

Respons

Nama	Deskripsi
`testId`	ID unik dari tes
`testName`	Nama tes
`testDescription`	Deskripsi tes
`testType`	Jenis pengujian yang dijalankan (misalnya ,`simple`,`jmeter`)
`fileType`	Jenis file yang diunggah (misalnya,, `nonescript`,`zip`)
`tags`	Array string untuk mengkategorikan tes
`status`	Status tes
`startTime`	Waktu dan tanggal ketika tes terakhir dimulai
`endTime`	Waktu dan tanggal ketika tes terakhir berakhir
`testScenario`	Definisi tes termasuk konkurensi, waktu pengujian, host, dan metode untuk pengujian
`taskCount`	Jumlah tugas yang dibutuhkan untuk menjalankan tes
`taskIds`	Daftar tugas IDs untuk menjalankan tes
`results`	Hasil akhir dari tes
`history`	Daftar hasil akhir dari tes sebelumnya (dikecualikan saat`history=false`)
`totalRuns`	Jumlah total uji coba untuk skenario ini
`lastRun`	Stempel waktu dari uji coba terakhir

Nama	Deskripsi
`errorReason`	Pesan kesalahan yang dihasilkan saat terjadi kesalahan
`nextRun`	Jalankan terjadwal berikutnya (misalnya ,`2017-04-22 17:18:00`)
`scheduleRecurrence`	Pengulangan tes (misalnya,,`daily`, `weeklybiweekly`,`monthly`)

POSTING /scenarios/ {TESid}

Deskripsi

`POST /scenarios/{testId}`Operasi ini memungkinkan Anda untuk membatalkan skenario pengujian tertentu.

Parameter permintaan

`testId`

- ID unik dari tes

  Tipe: String

  Diperlukan: Ya

Respons

Nama	Deskripsi
`status`	Status tes

HAPUS /scenarios/ {teSid}

Deskripsi

`DELETE /scenarios/{testId}`Operasi ini memungkinkan Anda untuk menghapus semua data yang terkait dengan skenario pengujian tertentu.

Parameter permintaan

`testId`

- ID unik dari tes

  Tipe: String

  Diperlukan: Ya

Respons

Nama	Deskripsi
`status`	Status tes

PILIHAN /scenarios/ {TESid}

Deskripsi

`OPTIONS /scenarios/{testId}`Operasi ini memberikan respons untuk permintaan dengan header respons CORS yang benar.

Respons

Nama	Deskripsi
`testId`	ID unik dari tes
`testName`	Nama tes

Nama	Deskripsi
`testDescription`	Deskripsi tes
`testType`	Jenis pengujian yang dijalankan (misalnya ,`simple`,`jmeter`)
`fileType`	Jenis file yang diunggah (misalnya,, `nonescript`,`zip`)
`status`	Status tes
`startTime`	Waktu dan tanggal ketika tes terakhir dimulai
`endTime`	Waktu dan tanggal ketika tes terakhir berakhir
`testScenario`	Definisi tes termasuk konkurensi, waktu pengujian, host, dan metode untuk pengujian
`taskCount`	Jumlah tugas yang dibutuhkan untuk menjalankan tes
`taskIds`	Daftar tugas IDs untuk menjalankan tes
`results`	Hasil akhir dari tes
`history`	Daftar hasil akhir dari tes sebelumnya
`errorReason`	Pesan kesalahan yang dihasilkan saat terjadi kesalahan

DAPATKAN /scenarios/ {teSTid} /testruns

Deskripsi

`GET /scenarios/{testId}/testruns`Operasi mengambil uji coba IDs untuk skenario pengujian tertentu, secara opsional difilter berdasarkan rentang waktu. Kapan`latest=true`, hanya mengembalikan satu uji coba terbaru.

Permintaan parameter

`testId`

- ID skenario pengujian

  Tipe: String

  Diperlukan: Ya

`latest`

- Kembalikan hanya ID uji coba terbaru

  Jenis: Boolean

  Default: `false`

  Wajib: Tidak

`start_timestamp`

- Stempel waktu ISO 8601 untuk memfilter pengujian berjalan dari (inklusif). Sebagai contoh, `2024-01-01T00:00:00Z`.

  Jenis: String (format tanggal-waktu)

  Wajib: Tidak

`end_timestamp`

- Stempel waktu ISO 8601 untuk memfilter pengujian berjalan hingga (inklusif). Sebagai contoh, `2024-12-31T23:59:59Z`.

  Jenis: String (format tanggal-waktu)

  Wajib: Tidak

`limit`

- Jumlah maksimum uji coba yang akan dikembalikan (diabaikan saat`latest=true`)

  Jenis: Integer (minimum: 1, maksimum: 100)

  Default: `20`

  Wajib: Tidak

`next_token`

- Token pagination dari respon sebelumnya untuk mendapatkan halaman berikutnya

  Tipe: String

  Wajib: Tidak

Respons

200 - Sukses

Nama	Deskripsi
`testRuns`	Array objek uji coba, masing-masing berisi `testRunId` (string) dan `startTime` (tanggal-waktu ISO 8601)
`pagination`	Objek yang mengandung `limit` (integer) dan `next_token` (string atau null). Token adalah nol jika tidak ada hasil lagi

Tanggapan Kesalahan

- `400`- Format atau parameter stempel waktu tidak valid
- `404`- Skenario uji tidak ditemukan
- `500`- Kesalahan server internal

Contoh penggunaan

- Uji coba terbaru saja: `GET /scenarios/test123/testruns?latest=true`
- Terbaru dalam rentang waktu: `GET /scenarios/test123/testruns?latest=true&start_timestamp=2024-01-01T00:00:00Z`
- Permintaan halaman berikutnya: `GET /scenarios/test123/testruns?limit=20&next_token=eyJ0ZXN0SWQiOiJzZVFVeTEyTEtMIiwic3RhcnRUaW1lIjoiMjAyNC0wMS`

DAPATKAN /scenarios/ {teSid} /testruns/ {} testRunId

Deskripsi

`GET /scenarios/{testId}/testruns/{testRunId}`Operasi mengambil hasil dan metrik lengkap untuk uji coba tertentu. Secara opsional hilangkan hasil riwayat dengan `history=false` respons yang lebih cepat.

Permintaan parameter

`testId`

- ID skenario pengujian

  Tipe: String

  Diperlukan: Ya

`testRunId`

- ID uji coba khusus

  Tipe: String

  Diperlukan: Ya

`history`

- Sertakan array riwayat sebagai tanggapan. Setel `false` untuk menghilangkan riwayat untuk respons yang lebih cepat

  Jenis: Boolean

  Default: `true`

  Wajib: Tidak

Respons

200 - Sukses

Nama	Deskripsi
`testId`	ID unik pengujian (misalnya,`seQUy12LKL`)

Nama	Deskripsi
`testRunId`	ID uji coba khusus (misalnya,`2DEwHItEne`)
`testDescription`	Deskripsi uji beban
`testType`	Jenis tes (misalnya,`simple`,`jmeter`)
`status`	Status uji coba:`complete`,, `runningfailed`, atau `cancelled`
`startTime`	Waktu dan tanggal ketika tes dimulai (misalnya ,`2025-09-09 21:01:00`)
`endTime`	Waktu dan tanggal ketika tes berakhir (misalnya,`2025-09-09 21:18:29`)
`succPercent`	Persentase keberhasilan (misalnya,`100.00`)
`testTaskConfigs`	Array objek konfigurasi tugas yang berisi`region`,`taskCount` , dan `concurrency`
`completeTasks`	Wilayah pemetaan objek untuk jumlah tugas yang diselesaikan
`results`	Objek yang berisi metrik terperinci termasuk `avg_lt` (latensi rata-rata), persentil (`p0_0`,`p50_0`,`p90_0`,`p95_0`,`p99_0`,`p99_9`,`p100_0`), `avg_rt` (waktu respons rata-rata), `avg_ct` (waktu koneksi rata-rata), `stdev_rt` (waktu respons deviasi standar),,`concurrency` , (jumlah keberhasilan)`throughput` , `succ` (jumlah kegagalan),,,`bytes`, `fail` (array kode respons) `testDuration` `metricS3Location` , `rc` dan array `labels`

Nama	Deskripsi
`testScenario`	Objek yang berisi konfigurasi uji dengan`execution` ,`reporting` , dan `scenarios` properti
`history`	Array hasil tes historis (dikecualikan saat`history=false`)

Tanggapan Kesalahan

- `400`- TeStid tidak valid atau testRunId
- `404`- Uji coba tidak ditemukan
- `500`- Kesalahan server internal

HAPUS /scenarios/ {teSTid} /testruns/ {} testRunId

Deskripsi

`DELETE /scenarios/{testId}/testruns/{testRunId}`Operasi menghapus semua data dan artefak yang terkait dengan uji coba tertentu. Data uji coba dihapus dari DynamoDB, sedangkan data pengujian aktual di S3 tetap tidak berubah.

Permintaan parameter

`testId`

- ID skenario pengujian

  Tipe: String

  Diperlukan: Ya

`testRunId`

- ID uji coba khusus yang akan dihapus

  Tipe: String

  Diperlukan: Ya

Respons

204 - Sukses

Uji coba berhasil dihapus (tidak ada konten yang dikembalikan)

Tanggapan Kesalahan

- `400`- TeStid tidak valid atau testRunId
- `403`- Terlarang: Izin tidak cukup untuk menghapus uji coba
- `404`- Uji coba tidak ditemukan
- `409`- Konflik: Uji coba sedang berjalan dan tidak dapat dihapus
- `500`- Kesalahan server internal

DAPATKAN /scenarios/ {teSid} /baseline

Deskripsi

`GET /scenarios/{testId}/baseline`Operasi mengambil hasil tes dasar yang ditunjuk untuk sebuah skenario. Mengembalikan baik tes dasar menjalankan ID atau hasil dasar penuh tergantung pada parameter. `data`

Permintaan parameter

`testId`

- ID skenario pengujian

  Tipe: String

  Diperlukan: Ya

`data`

- Kembalikan data uji dasar penuh jika`true`, jika tidak, hanya testRunId

  Jenis: Boolean

  Default: `false`

  Wajib: Tidak

Respons

200 - Sukses

Kapan `data=false` (default):

Nama	Deskripsi
`testId`	ID skenario pengujian (misalnya,`seQUy12LKL`)
`baselineTestRunId`	ID uji coba dasar (misalnya,) `2DEwHItEne`

Kapan`data=true`:

Nama	Deskripsi
`testId`	ID skenario pengujian (misalnya,`seQUy12LKL`)
`baselineTestRunId`	ID uji coba dasar (misalnya,) `2DEwHItEne`
`baselineData`	Objek hasil uji coba lengkap (struktur yang sama dengan`GET /scenarios/{testId}/testruns/{testRunId}`)

Tanggapan Kesalahan

- `400`- Parameter TeSid tidak valid
- `404`- Skenario uji tidak ditemukan atau tidak ada set dasar
- `500`- Kesalahan server internal

PUT /scenarios/ {teSid} /baseline

Deskripsi

`PUT /scenarios/{testId}/baseline`Operasi ini menetapkan uji coba tertentu sebagai dasar untuk perbandingan kinerja. Hanya satu baseline yang dapat ditetapkan per skenario.

Permintaan parameter

`testId`

- ID skenario pengujian

  Tipe: String

  Diperlukan: Ya

Isi permintaan

Nama	Deskripsi
`testRunId`	ID test run untuk ditetapkan sebagai baseline (misalnya,) `2DEwHItEne`

Respons

200 - Sukses

Nama	Deskripsi
`message`	Pesan konfirmasi (misalnya,`Baseline set successfully`)
`testId`	ID skenario pengujian (misalnya,`seQUy12LKL`)
`baselineTestRunId`	ID uji coba dasar yang ditetapkan (misalnya,) `2DEwHItEne`

Tanggapan Kesalahan

- `400`- TeStid tidak valid atau testRunId
- `404`- Skenario uji atau uji coba tidak ditemukan
- `409`- Konflik: Uji coba tidak dapat ditetapkan sebagai baseline (misalnya, pengujian gagal)
- `500`- Kesalahan server internal

HAPUS /scenarios/ {teSid} /baseline

Deskripsi

`DELETE /scenarios/{testId}/baseline`Operasi menghapus nilai dasar untuk skenario dengan menyetelnya ke string kosong.

Permintaan parameter

`testId`

- ID skenario pengujian

  Tipe: String

  Diperlukan: Ya

Respons

204 - Sukses

Baseline berhasil dihapus (tidak ada konten yang dikembalikan)

Tanggapan Kesalahan

- `400`- TeSid Tidak Valid
- `500`- Kesalahan server internal

DAPATKAN /tugas

Deskripsi

`GET /tasks`Operasi ini memungkinkan Anda mengambil daftar tugas Amazon Elastic Container Service (Amazon ECS) yang sedang berjalan.

Respons

Nama	Deskripsi
`tasks`	Daftar tugas IDs untuk menjalankan tes

PILIHAN/tugas

Deskripsi

Operasi `OPTIONS /tasks` tugas memberikan respons untuk permintaan dengan header respons CORS yang benar.

Respons

Nama	Deskripsi
`taskIds`	Daftar tugas IDs untuk menjalankan tes

DAPATKAN /wilayah

Deskripsi

`GET /regions`Operasi ini memungkinkan Anda untuk mengambil informasi sumber daya regional yang diperlukan untuk menjalankan tes di Wilayah tersebut.

Respons

Nama	Deskripsi
`testId`	ID Wilayah
`ecsCloudWatchLogGroup`	Nama grup CloudWatch log Amazon untuk tugas Amazon Fargate di Wilayah
`region`	Wilayah di mana sumber daya dalam tabel ada
`subnetA`	ID salah satu subnet di Wilayah
`subnetB`	ID salah satu subnet di Wilayah
`taskCluster`	Nama klaster AWS Fargate di Wilayah
`taskDefinition`	ARN definisi tugas di Wilayah
`taskImage`	Nama gambar tugas di Wilayah
`taskSecurityGroup`	ID grup keamanan di Wilayah

PILIHAN/wilayah

Deskripsi

`OPTIONS /regions`Operasi ini memberikan respons untuk permintaan dengan header respons CORS yang benar.

Respons

Nama	Deskripsi
`testId`	ID Wilayah
`ecsCloudWatchLogGroup`	Nama grup CloudWatch log Amazon untuk tugas Amazon Fargate di Wilayah

Nama	Deskripsi
`region`	Wilayah di mana sumber daya dalam tabel ada
`subnetA`	ID salah satu subnet di Wilayah
`subnetB`	ID salah satu subnet di Wilayah
`taskCluster`	Nama klaster AWS Fargate di Wilayah
`taskDefinition`	ARN definisi tugas di Wilayah
`taskImage`	Nama gambar tugas di Wilayah
`taskSecurityGroup`	ID grup keamanan di Wilayah

Meningkatkan sumber daya kontainer

Untuk meningkatkan jumlah pengguna virtual bersamaan (konkurensi) pengujian beban Anda dapat mensimulasikan, Anda perlu meningkatkan CPU dan sumber daya memori yang dialokasikan untuk setiap tugas Amazon ECS. Ini melibatkan pembuatan revisi definisi tugas baru dengan batas sumber daya yang lebih tinggi, kemudian memperbarui konfigurasi DynamoDB solusi untuk menggunakan definisi tugas baru untuk uji coba di masa mendatang.

Buat revisi definisi tugas baru

Ikuti langkah-langkah ini untuk membuat definisi tugas baru dengan peningkatan sumber daya CPU dan memori:

1. Masuk ke konsol Amazon Elastic Container Service.
2. Di menu navigasi kiri, pilih Definisi Tugas.
3. Pilih kotak centang di sebelah definisi tugas yang sesuai dengan solusi ini. Misalnya, `[replaceable]` <stackName>`- EcsTaskDefinition -*<system-generated-random-Hash>*.
4. Pilih Buat revisi baru.
5. Pada halaman Buat revisi baru, lakukan tindakan berikut:
   a. Di bawah Ukuran tugas, ubah memori Tugas dan CPU Tugas ke nilai yang Anda inginkan. Nilai yang lebih tinggi memungkinkan lebih banyak pengguna virtual bersamaan per tugas.

b. Di bawah Container Definitions, tinjau batas memori Hard/Soft. Jika batas ini lebih rendah dari memori yang Anda inginkan, pilih wadahnya.

c. Di kotak dialog Edit container, buka Memory Limits dan perbarui Hard Limit agar sesuai atau kurang dari alokasi memori tugas Anda.

d. Pilih Perbarui.

6. Pada halaman Buat revisi baru, pilih Buat.

7. Setelah definisi tugas berhasil dibuat, catat ARN definisi tugas lengkap termasuk nomor versi. Misalnya: `[replaceable]` <stackName>`- EcsTaskDefinition -*<system-generated-random-Hash>*: [dapat diganti]`<system-generated-versionNumber>`.

Perbarui tabel DynamoDB

Setelah membuat revisi definisi tugas baru, Anda harus memperbarui tabel DynamoDB solusi sehingga pengujian masa depan berjalan menggunakan definisi tugas baru. Ulangi langkah-langkah ini untuk setiap Wilayah AWS tempat Anda ingin menggunakan definisi tugas yang diperbarui:

1. Arahkan ke konsol DynamoDB.

2. Dari panel navigasi kiri, pilih Jelajahi Item di bawah Tabel.

3. Pilih tabel `scenarios-table` DynamoDB yang terkait dengan solusi ini. Misalnya, `[replaceable]` <stackName>`- DLTTest RunnerStorage DLTScenarios Tabel-*<system-generated-random-Hash>*.

4. Pilih item yang sesuai dengan Wilayah tempat Anda membuat revisi definisi tugas baru. Misalnya, `region-[replaceable]` <region-name>`.

5. Di editor item, cari atribut TaskDefinition dan perbarui nilainya dengan ARN definisi tugas lengkap yang Anda rekam di bagian sebelumnya (termasuk nomor versi).

6. Pilih Simpan perubahan.

> **Note**
>
> Definisi tugas yang diperbarui hanya akan digunakan untuk uji coba baru. Setiap tes yang sedang berjalan atau dijadwalkan akan terus menggunakan definisi tugas sebelumnya.

Spesifikasi alat MCP

Solusi Pengujian Beban Terdistribusi memperlihatkan seperangkat alat MCP yang memungkinkan agen AI berinteraksi dengan skenario dan hasil pengujian. Alat-alat ini memberikan kemampuan abstrak tingkat tinggi yang selaras dengan cara agen AI memproses informasi, memungkinkan mereka untuk fokus pada analisis dan wawasan daripada kontrak API terperinci.

> **Note**
>
> Semua alat MCP menyediakan akses hanya-baca ke data solusi. Tidak ada modifikasi untuk menguji skenario atau konfigurasi yang didukung melalui antarmuka MCP.

list_scenario

Deskripsi

`list_scenarios`Alat ini mengambil daftar semua skenario pengujian yang tersedia dengan metadata dasar.

Titik akhir

`GET /scenarios`

Parameter

Tidak ada

Respons

Nama	Deskripsi
`testId`	Pengidentifikasi unik untuk skenario pengujian
`testName`	Nama skenario pengujian
`status`	Status skenario pengujian saat ini
`startTime`	Saat pengujian dibuat atau terakhir dijalankan

Nama	Deskripsi
`testDescription`	Deskripsi skenario pengujian

get_scenario_details

Deskripsi

`get_scenario_details`Alat ini mengambil konfigurasi pengujian dan uji coba terbaru untuk satu skenario pengujian.

Titik akhir

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

Parameter permintaan

`test_id`

- Pengidentifikasi unik untuk skenario pengujian

  Tipe: String

  Diperlukan: Ya

Respons

Nama	Deskripsi
`testTaskConfigs`	Konfigurasi tugas untuk setiap wilayah
`testScenario`	Definisi dan parameter uji
`status`	Status tes saat ini
`startTime`	Uji stempel waktu mulai
`endTime`	Stempel waktu akhir uji (jika selesai)

list_test_runs

Deskripsi

`list_test_runs`Alat ini mengambil daftar uji coba untuk skenario pengujian tertentu, diurutkan terbaru ke yang terlama. Mengembalikan maksimal 30 hasil.

Titik akhir

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

atau

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

Permintaan parameter

`test_id`

- Pengidentifikasi unik untuk skenario pengujian

  Tipe: String

  Diperlukan: Ya

`limit`

- Jumlah maksimum uji coba yang akan dikembalikan

  Jenis: Integer

  Default: 20

  Maksimum: 30

  Wajib: Tidak

`start_date`

- Stempel waktu ISO 8601 untuk memfilter berjalan dari tanggal tertentu

  Jenis: String (format tanggal-waktu)

  Wajib: Tidak

`end_date`

- Stempel waktu ISO 8601 untuk memfilter berjalan hingga tanggal tertentu

  Jenis: String (format tanggal-waktu)

  Wajib: Tidak

Respons

Nama	Deskripsi
`testRuns`	Array ringkasan uji coba dengan metrik kinerja dan persentil untuk setiap proses

get_test_run

Deskripsi

`get_test_run`Alat ini mengambil hasil terperinci untuk satu uji coba dengan kerusakan regional dan titik akhir.

Titik akhir

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

Permintaan parameter

`test_id`

- Pengidentifikasi unik untuk skenario pengujian

  Tipe: String

  Diperlukan: Ya

`test_run_id`

- Pengidentifikasi unik untuk uji coba tertentu

  Tipe: String

Diperlukan: Ya

Respons

Nama	Deskripsi
`results`	Data uji coba lengkap termasuk rincian hasil regional, metrik spesifik titik akhir, persentil kinerja (p50, p90, p95, p99), jumlah keberhasi lan dan kegagalan, waktu respons dan latensi, dan konfigurasi pengujian yang digunakan untuk menjalankan

get_latest_test_run

Deskripsi

`get_latest_test_run`Alat ini mengambil uji coba terbaru untuk skenario pengujian tertentu.

Titik akhir

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

> **Note**
>
> Hasil diurutkan berdasarkan waktu menggunakan Indeks Sekunder Global (GSI), memastikan uji coba terbaru dikembalikan.

Parameter permintaan

`test_id`

- Pengidentifikasi unik untuk skenario pengujian

  Tipe: String

  Diperlukan: Ya

Respons

Nama	Deskripsi
`results`	Data uji coba terbaru dengan format yang sama dengan `get_test_run`

get_baseline_test_run

Deskripsi

`get_baseline_test_run`Alat ini mengambil uji dasar yang dijalankan untuk skenario pengujian tertentu. Baseline digunakan untuk tujuan perbandingan kinerja.

Titik akhir

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

Parameter permintaan

`test_id`

- Pengidentifikasi unik untuk skenario pengujian

  Tipe: String

  Diperlukan: Ya

Respons

Nama	Deskripsi
`baselineData`	Data uji coba dasar untuk tujuan perbandingan, termasuk semua metrik dan konfigurasi dari baseline run yang ditentukan

get_test_run_artefak

Deskripsi

`get_test_run_artifacts`Alat ini mengambil informasi bucket Amazon S3 untuk mengakses artefak pengujian termasuk log, file kesalahan, dan hasil.

Titik akhir

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

Permintaan parameter

`test_id`

- Pengidentifikasi unik untuk skenario pengujian

  Tipe: String

  Diperlukan: Ya

`test_run_id`

- Pengidentifikasi unik untuk uji coba tertentu

  Tipe: String

  Diperlukan: Ya

Respons

Nama	Deskripsi
`bucketName`	Nama bucket S3 tempat artefak disimpan
`testRunPath`	Awalan jalur untuk penyimpanan artefak saat ini (versi 4.0+)
`testScenarioPath`	Awalan jalur untuk penyimpanan artefak lama (pra-versi 4.0)

> **Note**
>
> Semua alat MCP memanfaatkan titik akhir API yang ada. Tidak ada modifikasi pada yang mendasari APIs yang diperlukan untuk mendukung fungsionalitas MCP.

Referensi

Bagian ini mencakup informasi tentang pengumpulan data, petunjuk ke sumber daya terkait, dan daftar pembangun yang berkontribusi pada solusi ini.

Pengumpulan data

Solusi ini mengirimkan metrik operasional ke AWS (“Data”) tentang penggunaan solusi ini. Kami menggunakan Data ini untuk lebih memahami bagaimana pelanggan menggunakan solusi ini serta layanan serta produk terkait. Pengumpulan AWS atas Data ini tunduk pada Pemberitahuan Privasi AWS.

Kontributor

- Tom Burung Bulbul
- Fernando Dingler
- Beomseok Lee
- George Lenz
- Erin McGill
- Dimitri Lopez
- Kamyar Ziabari
- Bassem Wani
- Garvit Singh
- Nikhil Reddy
- Simon Kroll
- Ahern Knox
- Ian Downard
- Owen Brady
- Jim Tharo
- Tiag Ramachandran
- Yang Qin
- James Wang

Glosarium

Glosarium ini mendefinisikan akronim dan singkatan yang digunakan di seluruh Panduan Implementasi Pengujian Beban Terdistribusi pada AWS.

Protokol dan Format Teknis

AGPL

Lisensi Publik Umum Affero. Lisensi perangkat lunak sumber terbuka yang digunakan oleh K6.

API

Antarmuka Pemrograman Aplikasi. Satu set protokol dan alat untuk membangun aplikasi perangkat lunak dan memungkinkan komunikasi antara sistem yang berbeda.

CLI

Antarmuka Baris Perintah. Antarmuka berbasis teks untuk berinteraksi dengan perangkat lunak dan sistem operasi.

CORS

Berbagi Sumber Daya Lintas Asal. Fitur keamanan yang memungkinkan atau membatasi aplikasi web yang berjalan pada satu asal untuk mengakses sumber daya dari asal yang berbeda.

CSV

Nilai yang Dipisahkan Koma. Format file yang digunakan untuk menyimpan data tabular dalam teks biasa, yang biasa digunakan untuk ekspor data.

gRPC

Panggilan Prosedur Jarak Jauh gRPC. Kerangka kerja sumber terbuka berkinerja tinggi untuk panggilan prosedur jarak jauh.

HTTP

Protokol Transfer Hiperteks. Protokol dasar yang digunakan untuk mentransmisikan data di World Wide Web.

HTTPS

HTTP Aman. Perpanjangan HTTP yang menggunakan enkripsi untuk komunikasi aman melalui jaringan.

JSON

JavaScript Notasi Objek. Format pertukaran data ringan yang mudah dibaca dan ditulis oleh manusia dan memudahkan mesin untuk mengurai dan menghasilkan.

JWT

Token Web JSON. Sarana ringkas dan aman URL untuk mewakili klaim yang akan ditransfer antara dua pihak untuk otentikasi dan otorisasi.

OAuth

Otorisasi terbuka. Standar terbuka untuk delegasi akses yang biasa digunakan untuk otentikasi dan otorisasi berbasis token.

REST

Transfer Negara Representasional. Gaya arsitektur untuk merancang aplikasi jaringan menggunakan komunikasi stateless dan metode HTTP standar.

SSE

Acara yang Dikirim Server. Sebuah teknologi push server memungkinkan klien untuk menerima update otomatis dari server melalui koneksi HTTP.

UI

Antarmuka Pengguna. Elemen visual dan kontrol melalui mana pengguna berinteraksi dengan aplikasi perangkat lunak.

URL

Pencari Sumber Daya Seragam. Alamat yang digunakan untuk mengakses sumber daya di internet.

XML

Bahasa Markup yang Dapat Diperluas. Bahasa markup yang mendefinisikan aturan untuk pengkodean dokumen dalam format yang dapat dibaca manusia dan dapat dibaca mesin.

Ketentuan Pengujian dan Database

FTP

Protokol Transfer File. Protokol jaringan standar yang digunakan untuk mentransfer file antara klien dan server.

GSI

Indeks Sekunder Global. Fitur DynamoDB yang memungkinkan Anda untuk query data menggunakan kunci alternatif.

JDBC

Konektivitas Database Java. Java API untuk menghubungkan dan mengeksekusi query dengan database.

JMS

Layanan Pesan Java. Java API untuk mengirim pesan antara dua atau lebih klien.

TPS

Transaksi Per Detik. Ukuran jumlah transaksi yang dapat diproses sistem dalam satu detik.

AWS dan Ketentuan Sistem

ARN

Nama Sumber Daya Amazon. Pengidentifikasi unik untuk sumber daya AWS yang digunakan untuk menentukan sumber daya di seluruh layanan AWS.

ISO

Organisasi Internasional untuk Standardisasi. Organisasi non-pemerintah independen yang mengembangkan standar internasional. Direferensikan dalam panduan ini untuk format stempel waktu ISO 8601.

SLA

Perjanjian Tingkat Layanan. Komitmen antara penyedia layanan dan pelanggan yang menentukan tingkat layanan yang diharapkan.

UUID

Pengidentifikasi Unik Universal. Angka 128-bit yang digunakan untuk mengidentifikasi informasi secara unik dalam sistem komputer.

vCPU

Unit Pemrosesan Pusat Virtual. Prosesor virtual yang ditugaskan ke mesin virtual atau wadah, mewakili sebagian dari kekuatan pemrosesan CPU fisik.

Ketentuan Pengujian Beban

konkurensi

Jumlah pengguna virtual bersamaan per tugas. Parameter ini mengontrol berapa banyak pengguna simulasi yang dihasilkan setiap tugas Fargate selama uji beban.

tumpukan regional

CloudFormation Tumpukan yang digunakan di Wilayah AWS untuk menyediakan infrastruktur pengujian untuk pengujian beban multi-wilayah.

jumlah tugas

Jumlah kontainer Fargate (tugas) diluncurkan untuk menjalankan skenario pengujian. Total beban yang dihasilkan sama dengan jumlah tugas dikalikan dengan konkurensi.

skenario uji

Uji beban yang dikonfigurasi termasuk jenis pengujian, titik akhir target, jumlah tugas, konkurensi, durasi, dan parameter lainnya.

Revisi

Kunjungi ChangelOG.md di GitHub repositori kami untuk melacak peningkatan dan perbaikan khusus versi.

Pemberitahuan

Pelanggan bertanggung jawab untuk membuat penilaian independen mereka sendiri atas informasi dalam dokumen ini. Dokumen ini: (a) hanya untuk tujuan informasi, (b) mewakili penawaran dan praktik produk AWS saat ini, yang dapat berubah tanpa pemberitahuan, dan (c) tidak membuat komitmen atau jaminan apa pun dari AWS dan afiliasinya, pemasok, atau pemberi lisensinya. Produk atau layanan AWS disediakan “sebagaimana adanya” tanpa jaminan, pernyataan, atau ketentuan dalam bentuk apa pun, baik tersurat maupun tersirat. Tanggung jawab dan kewajiban AWS kepada pelanggannya dikendalikan oleh perjanjian AWS, dan dokumen ini bukan bagian dari, juga tidak mengubah, perjanjian apa pun antara AWS dan pelanggannya.

Pengujian Beban Terdistribusi di AWS dilisensikan berdasarkan ketentuan Lisensi Apache Versi 2.0 yang tersedia di The Apache Software Foundation.

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.