Pendekatan bertahap untuk rekayasa kinerja di AWS Cloud

AWS Panduan Preskriptif

AWS Panduan Preskriptif: Pendekatan bertahap untuk rekayasa kinerja di AWS Cloud

Table of Contents

Pendekatan bertahap untuk rekayasa kinerja di AWS Cloud

Amazon Web Services (kontributor)

April 2024 (riwayat dokumen)

Panduan ini menguraikan praktik terbaik untuk merencanakan, membangun, dan mengaktifkan rekayasa kinerja untuk beban kerja aplikasi yang berjalan di Amazon Web Services ().AWS Ini menjabarkan empat pilar untuk rekayasa kinerja, dan menyarankan pendekatan yang berbeda untuk memenuhi persyaratan kinerja aplikasi. Untuk setiap pilar, panduan ini mencantumkan alat dan solusi untuk menyiapkan pengujian kinerja dan lingkungan pengujian.

Apa itu rekayasa kinerja?

Rekayasa Kinerja mencakup teknik yang diterapkan selama siklus hidup pengembangan sistem untuk memastikan persyaratan kinerja non-fungsional (seperti throughput, latensi, atau penggunaan memori) terpenuhi.

Sebelum pengujian kinerja dimulai, Anda perlu mengatur lingkungan kinerja. Lingkungan kinerja yang khas berdiri di atas pilar berikut:

- Pembuatan data uji
- Uji observabilitas
- Otomatisasi uji
- Pelaporan uji

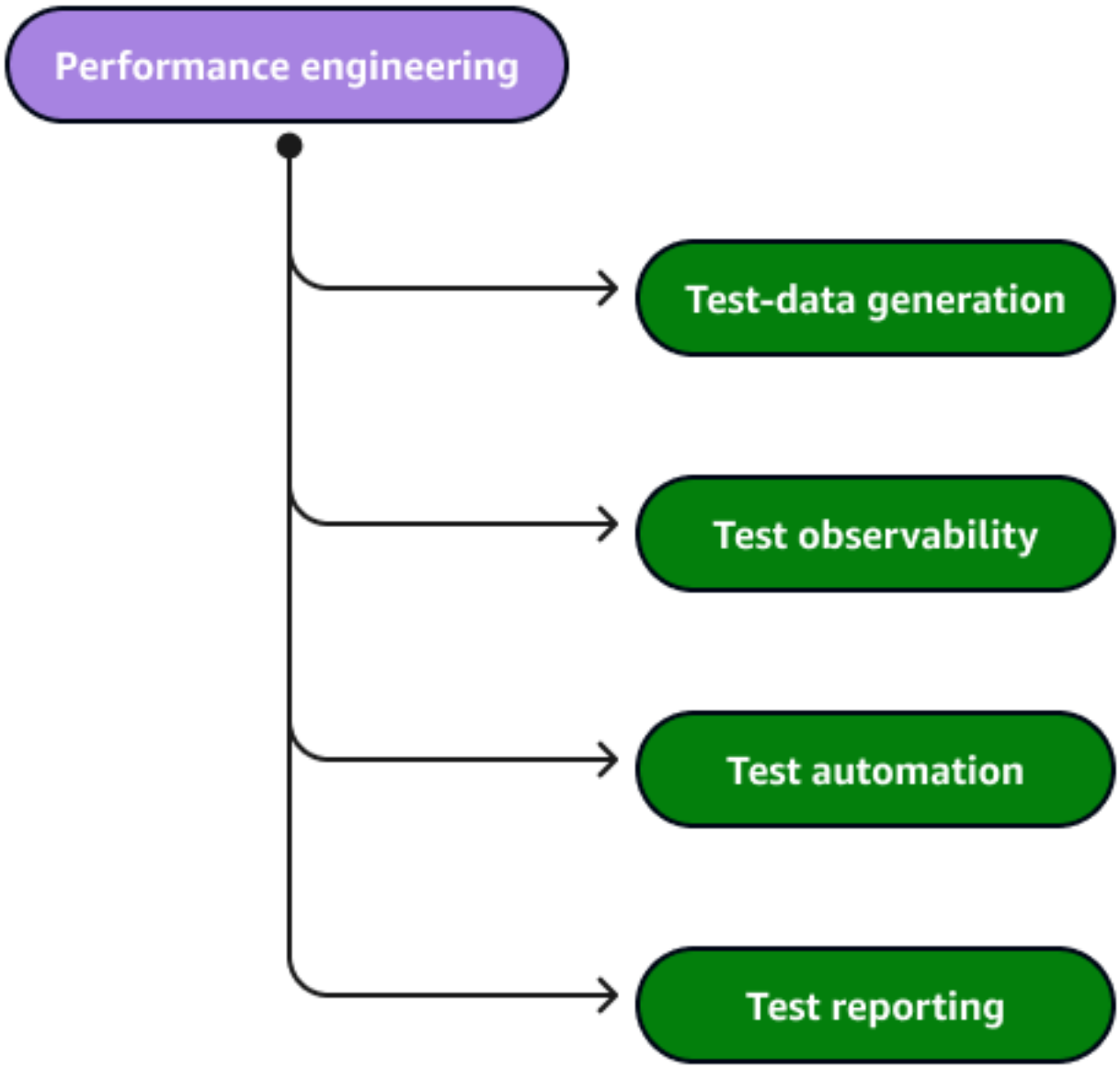
Mengapa menggunakan rekayasa kinerja?

Performance engineering adalah proses terus mengoptimalkan kinerja aplikasi sejak awal tahap desain. Ini membawa nilai besar bagi bisnis dengan menghindari pengerjaan ulang dan refactoring kode pada tahap selanjutnya dalam siklus pengembangan. Memulai rekayasa kinerja pada tahap desain menghasilkan aplikasi yang berkinerja lebih baik karena kinerja dapat diperhitungkan dalam desain. Rekayasa kinerja membutuhkan partisipasi aktif dari arsitek sistem, pengembang DevOps, dan Jaminan Kualitas.

Pilar rekayasa kinerja

Untuk mengaktifkan pola pikir rekayasa kinerja, penting untuk membangun fondasi yang kuat sambil menyiapkan rekayasa kinerja untuk aplikasi. Rekayasa kinerja membutuhkan pengaturan empat pilar utama:

- Pembuatan data uji — Insinyur kinerja menyiapkan alat untuk menghasilkan data uji.
- Pengamatan uji — Insinyur kinerja mengatur lingkungan observabilitas untuk memastikan bahwa kinerja berjalan dapat dicatat dan dilacak, dan bahwa sumber daya yang menangani beban dipantau.
- Otomatisasi uji — Insinyur kinerja mengembangkan tes otomatis yang mensimulasikan lalu lintas pengguna dan beban sistem menggunakan alat seperti Apache JMeter atau ghz.
- Pelaporan pengujian — Data dikumpulkan tentang konfigurasi setiap pengujian yang dijalankan bersama dengan hasil kinerja. Data memungkinkan mengkorelasikan perubahan konfigurasi dengan kinerja dan memberikan wawasan yang berharga.



Memasukkan pilar-pilar ini akan mendorong pola pikir kinerja mulai dari fase awal desain. Ini akan membantu menghindari perubahan pada aplikasi atau lingkungan dalam fase pengembangan dan pengujian selanjutnya.

Pembuatan data uji

Pembuatan test-data melibatkan pembuatan dan pemeliharaan sejumlah besar data untuk menjalankan kasus uji kinerja. Data yang dihasilkan ini bertindak sebagai masukan untuk kasus uji sehingga aplikasi dapat diuji pada kumpulan data yang beragam.

Seringkali, menghasilkan data uji adalah proses yang kompleks. Namun, menggunakan kumpulan data yang dibuat dengan buruk dapat menyebabkan perilaku aplikasi yang tidak dapat diprediksi di lingkungan produksi. Pembuatan test-data untuk pengujian kinerja berbeda dari pendekatan pembuatan test-data tradisional. Ini membutuhkan skenario dunia nyata, dan sebagian besar pelanggan ingin menguji beban kerja mereka dengan data yang mirip dengan data produksi aktual mereka. Data pengujian yang dihasilkan juga biasanya perlu disetel ulang atau disegarkan ke keadaan semula setelah setiap pengujian dijalankan, yang menambah waktu dan upaya.

Pembuatan data uji mencakup pertimbangan utama berikut:

- Akurasi — Akurasi data penting dalam semua aspek pengujian. Data yang tidak akurat menghasilkan hasil yang tidak akurat. Misalnya, ketika transaksi kartu kredit dihasilkan, seharusnya tidak untuk tanggal di masa depan.
- Validitas — Data harus valid untuk kasus penggunaan. Misalnya, saat menguji transaksi kartu kredit, tidak disarankan untuk menghasilkan 10.000 transaksi per pengguna per hari, karena ini menyimpang secara signifikan dari skenario kasus penggunaan yang valid.
- Otomatisasi — Otomatisasi pembuatan data uji dapat membawa manfaat upaya waktu. Ini juga mengarah pada otomatisasi pengujian yang efektif. Menghasilkan data uji secara manual dapat memiliki konsekuensi sehubungan dengan persyaratan kualitas dan upaya waktu.

  Ada mekanisme berbeda yang dapat diadopsi berdasarkan kasus penggunaan sebagai berikut:

  - Didorong API - Dalam hal ini, pengembang menyediakan API generasi data uji yang dapat digunakan penguji untuk menghasilkan data. Dengan menggunakan alat pengujian seperti JMeter, penguji dapat menskalakan pembuatan data menggunakan API bisnis. Misalnya, jika Anda memiliki API untuk menambahkan pengguna, Anda dapat menggunakan API yang sama untuk membuat ratusan pengguna dengan profil yang berbeda. Demikian pula, Anda dapat menghapus pengguna dengan memanggil operasi delete API. Untuk aplikasi alur kerja yang kompleks, pengembang dapat menyediakan API komposit yang dapat menghasilkan kumpulan data di berbagai komponen. Dengan menggunakan pendekatan ini, penguji dapat menulis otomatisasi untuk menghasilkan dan menghapus kumpulan data berdasarkan kebutuhan mereka.

    Namun, jika sistemnya kompleks atau waktu respons API per pemanggilan tinggi, mungkin perlu waktu lama untuk menyiapkan dan merobohkan data.
  - SQL statement driven — Pendekatan alternatif adalah dengan menggunakan pernyataan SQL backend untuk menghasilkan volume data yang besar. Pengembang dapat memberikan pernyataan SQL berbasis template untuk pembuatan data uji. Penguji dapat menggunakan

pernyataan untuk mengisi data, atau mereka dapat membuat skrip pembungkus di atas pernyataan ini untuk mengotomatiskan pembuatan data uji. Dengan menggunakan pendekatan ini, penguji dapat mengisi dan meruntuhkan data dengan sangat cepat jika data perlu diatur ulang setelah pengujian selesai. Namun, pendekatan ini memerlukan akses langsung ke database aplikasi, yang mungkin tidak mungkin dilakukan di lingkungan aman yang khas. Selain itu, kueri yang tidak valid dapat mengakibatkan populasi data yang salah, yang dapat menghasilkan hasil yang miring. Pengembang juga harus terus memperbarui pernyataan SQL dalam kode aplikasi untuk mencerminkan perubahan yang dibuat pada aplikasi dari waktu ke waktu.

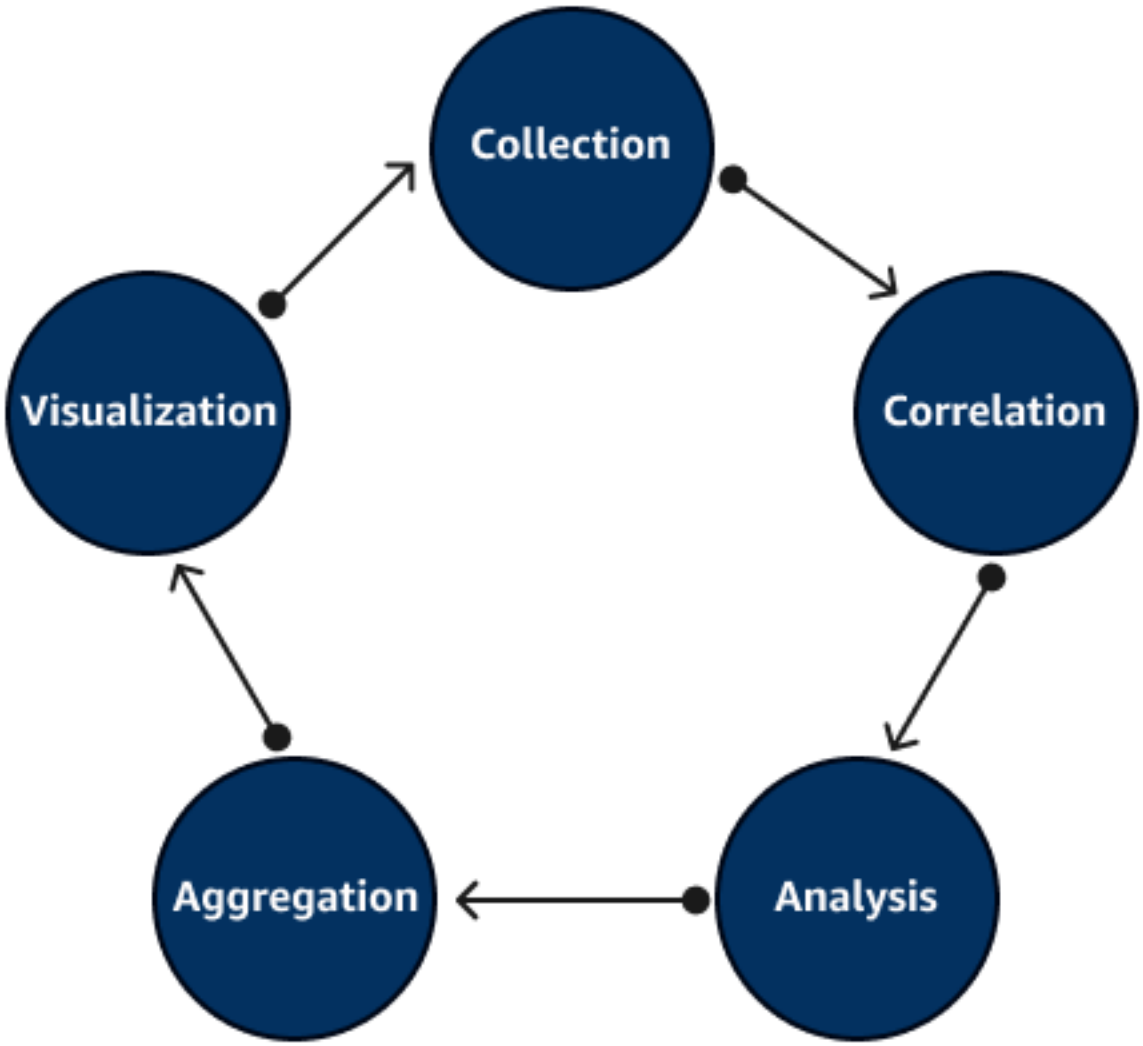
Alat pembuatan data uji

AWS menyediakan alat kustom asli yang dapat Anda gunakan untuk pembuatan data uji:

- Amazon Kinesis Data Generator - Amazon Kinesis Data Generator (KDG) menyederhanakan tugas menghasilkan data dan mengirimkannya ke Amazon Kinesis. Alat ini menyediakan UI yang ramah pengguna yang berjalan langsung di browser Anda. Untuk informasi selengkapnya dan implementasi referensi, lihat Test Your Streaming Data Solution dengan postingan blog Amazon Kinesis Data Generator Baru.
- AWS Glue Test Data Generator - AWS Glue Test Data Generator menyediakan kerangka kerja yang dapat dikonfigurasi untuk pembuatan data uji menggunakan AWS Glue PySpark pekerjaan tanpa server. Deskripsi data uji yang diperlukan sepenuhnya dapat dikonfigurasi melalui file konfigurasi YAMAL. Untuk informasi selengkapnya dan implementasi referensi, lihat GitHub repositori AWS Glue Test Data Generator.

Uji observabilitas

Pengamatan pengujian mendukung pengumpulan, korelasi, agregasi, dan analisis telemetri di jaringan, infrastruktur, dan aplikasi Anda selama pengujian kinerja berjalan. Anda mendapatkan wawasan penuh tentang perilaku, kinerja, dan kesehatan sistem Anda. Wawasan ini membantu Anda mendeteksi, menyelidiki, dan memperbaiki masalah lebih cepat. Dengan menambahkan kecerdasan buatan dan pembelajaran mesin, Anda dapat secara proaktif bereaksi, memprediksi, dan mencegah masalah.



Observabilitas bergantung pada penebangan, pemantauan, dan penelusuran. Tanggung jawab pelaksanaan kegiatan ini berhasil mencakup tim aplikasi dan infrastruktur.

Pada awal fase desain, tim aplikasi harus memahami keadaan tumpukan observabilitas mereka saat ini, termasuk pencatatan, pemantauan, dan penelusuran. Mereka kemudian dapat memilih alat yang terintegrasi lebih lancar ke dalam tumpukan observabilitas.

Demikian pula, tim infrastruktur bertanggung jawab untuk mengelola dan menskalakan infrastruktur observabilitas.

Pertimbangkan aspek-aspek berikut sehubungan dengan observabilitas pengujian:

- Ketersediaan log dan jejak aplikasi
- Korelasi log dan jejak

- Ketersediaan node, kontainer, dan metrik aplikasi
- Otomatisasi untuk mengatur dan memperbarui infrastruktur observabilitas sesuai permintaan
- Kemampuan untuk memvisualisasikan telemetri
- Penskalaan infrastruktur observabilitas

Pencatatan log

Logging adalah proses menyimpan data tentang peristiwa yang terjadi dalam suatu sistem. Log dapat mencakup masalah, kesalahan, atau informasi tentang operasi saat ini. Log dapat diklasifikasikan ke dalam berbagai jenis, seperti berikut ini:

- Log peristiwa
- Log server
- Log sistem
- Otorisasi dan akses log
- Log Audit

Pengembang dapat mencari log untuk kode atau pola kesalahan tertentu, memfilternya berdasarkan bidang tertentu, atau mengarsipkannya dengan aman untuk analisis masa depan. Log membantu pengembang untuk melakukan analisis akar penyebab untuk masalah kinerja dan juga untuk berkorelasi antara komponen sistem.

Membangun solusi logging yang efektif melibatkan koordinasi yang erat antara tim aplikasi dan infrastruktur. Log aplikasi tidak berguna kecuali ada infrastruktur logging yang dapat diskalakan yang mendukung kasus penggunaan seperti parsing, filtering, buffering, dan korelasi log. Kasus penggunaan umum, seperti menghasilkan ID korelasi, mencatat waktu berjalan untuk metode penting bisnis, dan mendefinisikan pola log, dapat disederhanakan.

Tim aplikasi

Pengembang aplikasi harus memastikan bahwa log yang dihasilkan mengikuti praktik terbaik pencatatan. Praktik terbaik meliputi:

- Menghasilkan korelasi IDs untuk melacak permintaan unik
- Mencatat waktu yang dibutuhkan oleh metode penting bisnis
- Logging pada tingkat log yang sesuai

- Berbagi pustaka logging umum

Saat Anda merancang aplikasi yang berinteraksi dengan layanan mikro yang berbeda, gunakan prinsip desain logging ini untuk menyederhanakan penyaringan dan ekstraksi log di backend.

Menghasilkan korelasi IDs untuk melacak permintaan unik

Ketika aplikasi menerima permintaan, itu dapat memeriksa apakah ID korelasi sudah ada di header. Jika ID tidak ada, aplikasi harus menghasilkan ID. Misalnya, Application Load Balancer menambahkan header yang disebut. `X-Amzn-Trace-Id` Aplikasi dapat menggunakan header untuk menghubungkan permintaan dari penyeimbang beban ke aplikasi. Demikian pula, aplikasi harus menyuntikkan `traceId` jika memanggil layanan mikro dependen sehingga log yang dihasilkan oleh komponen yang berbeda dalam aliran permintaan berkorelasi.

Mencatat waktu yang dibutuhkan oleh metode penting bisnis

Ketika aplikasi menerima permintaan, ia berinteraksi dengan komponen yang berbeda. Aplikasi harus mencatat waktu yang dibutuhkan untuk metode bisnis kritis dalam pola yang ditentukan. Ini dapat membuatnya lebih mudah untuk mengurai log di backend. Ini juga dapat membantu Anda menghasilkan wawasan yang berguna dari log. Anda dapat menggunakan pendekatan seperti pemrograman berorientasi aspek (AOP) untuk menghasilkan log tersebut sehingga Anda dapat memisahkan masalah logging dari logika bisnis Anda.

Logging pada tingkat log yang sesuai

Aplikasi harus menulis log yang memiliki jumlah informasi yang bermanfaat. Gunakan level log untuk mengkategorikan peristiwa berdasarkan tingkat keparahannya. Misalnya, gunakan `WARNING` dan `ERROR` tingkatkan untuk peristiwa penting yang perlu diselidiki. Gunakan `INFO` dan `DEBUG` untuk penelusuran mendetail dan peristiwa volume tinggi. Atur penangan log untuk menangkap hanya level yang diperlukan dalam produksi. Menghasilkan terlalu banyak logging di `INFO` level tersebut tidak membantu, dan itu menambah tekanan dalam infrastruktur backend. `DEBUG`logging bisa bermanfaat, tetapi harus digunakan dengan hati-hati. Menggunakan `DEBUG` log dapat menghasilkan volume data yang besar, sehingga tidak direkomendasikan dalam lingkungan pengujian kinerja.

Berbagi pustaka logging umum

Tim aplikasi harus menggunakan pustaka logging umum, seperti AWS SDK untuk Java, dengan pola logging umum yang telah ditentukan sebelumnya yang dapat digunakan pengembang sebagai dependensi dalam proyek mereka.

Tim infrastruktur

DevOps insinyur dapat mengurangi upaya dengan menggunakan prinsip desain logging berikut saat memfilter dan mengekstraksi log di backend. Tim infrastruktur harus mengatur dan mendukung sumber daya berikut.

Agen log

Agen log (pengirim log) adalah program yang membaca log dari satu lokasi dan mengirimkannya ke lokasi lain. Agen log digunakan untuk membaca file log yang disimpan di komputer dan mengunggah peristiwa log ke backend untuk sentralisasi.

Log adalah data tidak terstruktur yang harus terstruktur sebelum Anda dapat membuat wawasan yang berarti darinya. Agen log menggunakan parser untuk membaca pernyataan log dan mengekstrak bidang yang relevan seperti stempel waktu, tingkat log, dan nama layanan, dan mereka menyusun data tersebut ke dalam format JSON. Memiliki agen log ringan di tepi berguna karena mengarah pada pemanfaatan sumber daya yang lebih sedikit. Agen log dapat langsung mendorong ke backend, atau dapat menggunakan forwarder log perantara yang mendorong data ke backend. Menggunakan pengirim log membongkar pekerjaan dari agen log di sumbernya.

Pengurai log

Sebuah parser log mengubah log yang tidak terstruktur menjadi log terstruktur. Parser agen log juga memperkaya log dengan menambahkan metadata. Penguraian data data dapat dilakukan di sumber (akhir aplikasi) atau dapat dilakukan secara terpusat. Skema untuk menyimpan log harus dapat diperluas sehingga Anda dapat menambahkan bidang baru. Sebaiknya gunakan format log standar seperti JSON. Namun, dalam beberapa kasus, log harus diubah ke format JSON untuk pencarian yang lebih baik. Menulis ekspresi parser yang tepat memungkinkan transformasi yang efisien.

Backend log

Layanan backend log mengumpulkan, menyerap, dan memvisualisasikan data log dari berbagai sumber. Agen log dapat langsung menulis ke backend atau menggunakan forwarder log perantara. Saat pengujian kinerja, pastikan untuk menyimpan log sehingga dapat dicari di lain waktu. Simpan log di backend secara terpisah untuk setiap aplikasi. Misalnya, gunakan indeks khusus untuk aplikasi, dan gunakan pola indeks untuk mencari log yang tersebar di berbagai aplikasi terkait. Kami merekomendasikan menyimpan setidaknya 7 hari data untuk pencarian log. Namun, menyimpan data untuk durasi yang lebih lama dapat mengakibatkan biaya penyimpanan yang tidak perlu. Karena volume log yang besar dihasilkan selama pengujian kinerja, penting bagi infrastruktur logging untuk menskalakan dan mengukur backend logging dengan benar.

Visualisasi log

Untuk mendapatkan wawasan yang bermakna dan dapat ditindaklanjuti dari log aplikasi, gunakan alat visualisasi khusus untuk memproses dan mengubah data log mentah menjadi representasi grafis. Visualisasi seperti bagan, grafik, dan dasbor dapat membantu mengungkap tren, pola, dan anomali yang mungkin tidak mudah terlihat saat melihat log mentah.

Manfaat utama menggunakan alat visualisasi termasuk kemampuan untuk mengkorelasikan data di berbagai sistem dan aplikasi untuk mengidentifikasi dependensi dan kemacetan. Dasbor interaktif mendukung pengeboran data pada tingkat perincian yang berbeda untuk memecahkan masalah atau melihat tren penggunaan. Platform visualisasi data khusus menyediakan kemampuan seperti analitik, peringatan, dan berbagi data yang dapat meningkatkan pemantauan dan analisis.

Dengan menggunakan kekuatan visualisasi data pada log aplikasi, tim pengembangan dan operasi dapat memperoleh visibilitas ke dalam kinerja sistem dan aplikasi. Wawasan yang diperoleh dapat digunakan untuk berbagai tujuan, termasuk mengoptimalkan efisiensi, meningkatkan pengalaman pengguna, meningkatkan keamanan, dan perencanaan kapasitas. Hasil akhirnya adalah dasbor yang disesuaikan dengan berbagai pemangku kepentingan, memberikan at-a-glance pandangan yang merangkum data log menjadi informasi yang dapat ditindaklanjuti dan berwawasan luas.

Mengotomatiskan infrastruktur logging

Karena aplikasi yang berbeda memiliki persyaratan yang berbeda, penting untuk mengotomatiskan instalasi dan pengoperasian infrastruktur logging. Gunakan alat infrastruktur sebagai kode (IAc) untuk menyediakan backend infrastruktur logging. Kemudian Anda dapat menyediakan infrastruktur logging baik sebagai layanan bersama atau sebagai penyebaran pesan lebih dahulu independen untuk aplikasi tertentu.

Kami menyarankan agar pengembang menggunakan pipeline pengiriman berkelanjutan (CD) untuk mengotomatiskan hal-hal berikut:

- Terapkan infrastruktur logging sesuai permintaan dan hancurkan saat tidak diperlukan.
- Menyebarkan agen log di berbagai target.
- Terapkan konfigurasi parser log dan forwarder.
- Menyebarkan dasbor aplikasi.

Alat pencatatan

AWS menyediakan layanan pencatatan asli, mengkhawatirkan, dan dasbor. Berikut ini adalah populer Layanan AWS dan sumber daya untuk logging:

- Amazon OpenSearch Service membantu organisasi mengumpulkan, menyerap, dan memvisualisasikan data log dari berbagai sumber. Untuk informasi selengkapnya, lihat Pencatatan Terpusat dengan OpenSearch.
- CloudWatch Agen Amazon dan AWS Fluent Bit adalah agen log paling populer di AWS. Untuk informasi tentang penggunaan CloudWatch agen dengan Amazon CloudWatch Logs Insights, lihat posting blog Menyederhanakan log server Apache dengan Amazon CloudWatch Logs Insights. AWS Untuk implementasi referensi Fluent Bit, lihat posting blog Sentralized Container Logging with Fluent Bit.

Memantau

Pemantauan adalah proses mengumpulkan metrik yang berbeda, seperti CPU dan memori, dan menyimpannya dalam database deret waktu seperti Amazon Managed Service untuk Prometheus. Sistem pemantauan dapat berbasis push atau pull based. Dalam sistem berbasis push, sumber mendorong metrik secara berkala ke basis data deret waktu. Dalam sistem berbasis tarik, scraper mengikis metrik dari berbagai sumber dan menyimpannya dalam database deret waktu. Pengembang dapat menganalisis metrik, memfilter metrik, dan memplot mereka dari waktu ke waktu untuk memvisualisasikan kinerja. Melaksanakan pemantauan dengan sukses dapat dibagi menjadi dua bidang luas: aplikasi dan infrastruktur.

Untuk pengembang aplikasi, metrik berikut sangat penting:

- Latensi — Waktu yang dibutuhkan untuk menerima tanggapan
- Permintaan throughput — Jumlah total permintaan yang ditangani per detik
- Tingkat kesalahan permintaan - Jumlah total kesalahan

Tangkap pemanfaatan sumber daya, saturasi, dan jumlah kesalahan untuk setiap sumber daya (seperti wadah aplikasi, database) yang terlibat dalam transaksi bisnis. Misalnya, saat memantau penggunaan CPU, Anda dapat melacak pemanfaatan CPU rata-rata, beban rata-rata, dan beban puncak selama uji kinerja dijalankan. Ketika sumber daya mencapai saturasi selama pengujian stres, tetapi mungkin tidak mencapai saturasi selama kinerja berjalan untuk periode waktu yang lebih singkat.

Metrik-metrik

Aplikasi dapat menggunakan aktuator yang berbeda, seperti aktuator boot pegas, untuk memantau aplikasinya. Pustaka tingkat produksi ini umumnya mengekspos titik akhir REST untuk memantau informasi tentang aplikasi yang sedang berjalan. Pustaka dapat memantau infrastruktur yang mendasarinya, platform aplikasi, dan sumber daya lainnya. Jika salah satu metrik default tidak memenuhi persyaratan, pengembang harus menerapkan metrik khusus. Metrik khusus dapat membantu melacak indikator kinerja kunci bisnis (KPIs) yang tidak dapat dilacak melalui data dari implementasi default. Misalnya, Anda mungkin ingin melacak operasi bisnis seperti latensi integrasi API pihak ketiga atau jumlah total transaksi yang diselesaikan.

Kardinalitas

Kardinalitas mengacu pada jumlah deret waktu yang unik dari suatu metrik. Metrik diberi label untuk memberikan informasi tambahan. Misalnya, aplikasi berbasis REST yang melacak jumlah permintaan untuk API tertentu menunjukkan kardinalitas 1. Jika Anda menambahkan label pengguna untuk mengidentifikasi jumlah permintaan per pengguna, kardinalitas meningkat secara proporsional dengan jumlah pengguna. Dengan menambahkan label yang menciptakan kardinalitas, Anda dapat mengiris dan memotong metrik berdasarkan berbagai grup. Penting untuk menggunakan label yang tepat untuk kasus penggunaan yang tepat karena kardinalitas meningkatkan jumlah rangkaian metrik dalam database deret waktu pemantauan backend.

Resolusi

Dalam pengaturan pemantauan tipikal, aplikasi pemantauan dikonfigurasi untuk mengikis metrik dari aplikasi secara berkala. Periodisitas pengikisan mendefinisikan granularitas data pemantauan. Metrik yang dikumpulkan pada interval yang lebih pendek cenderung memberikan tampilan kinerja yang lebih akurat karena lebih banyak titik data yang tersedia. Namun, beban pada database deret waktu meningkat karena lebih banyak entri disimpan. Biasanya granularitas 60 detik adalah resolusi standar dan 1 detik adalah resolusi tinggi.

DevOps tim

Pengembang aplikasi sering meminta DevOps insinyur untuk mengatur lingkungan pemantauan untuk memvisualisasikan metrik infrastruktur dan aplikasi. DevOps Insinyur harus mengatur lingkungan yang dapat diskalakan dan mendukung alat visualisasi data yang digunakan oleh pengembang aplikasi. Ini melibatkan pengikisan data pemantauan dari berbagai sumber dan mengirim data ke database deret waktu pusat seperti Amazon Managed Service untuk Prometheus.

Pemantauan backend

Layanan backend pemantauan mendukung pengumpulan, penyimpanan, kueri, dan visualisasi data metrik. Ini biasanya database deret waktu seperti Amazon Managed Service untuk Prometheus atau InfluxDB. InfluxData Dengan menggunakan mekanisme penemuan layanan, kolektor pemantauan dapat mengumpulkan metrik dari berbagai sumber dan menyimpannya. Sementara pengujian kinerja, penting untuk menyimpan data metrik sehingga dapat dicari di lain waktu. Kami merekomendasikan untuk menyimpan setidaknya 15 hari data untuk metrik. Namun, menyimpan metrik untuk durasi yang lebih lama tidak menambah manfaat yang signifikan dan menyebabkan biaya penyimpanan yang tidak perlu. Karena pengujian kinerja dapat menghasilkan volume metrik yang besar, penting bagi infrastruktur metrik untuk menskalakan sambil memberikan kinerja kueri yang cepat. Layanan backend pemantauan menyediakan bahasa kueri yang dapat digunakan untuk melihat data metrik.

Visualisasi

Menyediakan alat visualisasi yang dapat menampilkan data aplikasi untuk memberikan wawasan yang berarti. DevOps Insinyur dan pengembang aplikasi harus mempelajari bahasa kueri untuk backend pemantauan dan bekerja sama untuk menghasilkan template dasbor yang dapat digunakan kembali. Di dasbor, sertakan latensi, dan kesalahan sambil juga menampilkan pemanfaatan dan saturasi sumber daya di seluruh infrastruktur dan sumber daya aplikasi.

Mengotomatiskan infrastruktur pemantauan

Mirip dengan logging, penting untuk mengotomatiskan instalasi dan pengoperasian infrastruktur pemantauan sehingga Anda dapat mengakomodasi berbagai persyaratan aplikasi yang berbeda. Gunakan alat IAc untuk menyediakan backend infrastruktur pemantauan. Kemudian Anda dapat menyediakan infrastruktur pemantauan baik sebagai layanan bersama atau sebagai penyebaran pesan lebih dahulu independen untuk aplikasi tertentu.

Gunakan saluran pipa CD untuk mengotomatiskan hal berikut:

- Menyebarkan infrastruktur pemantauan sesuai permintaan dan meruntuhkannya ketika tidak diperlukan.
- Perbarui konfigurasi pemantauan untuk memfilter atau menggabungkan metrik.
- Menyebarkan dasbor aplikasi.

Alat-alat pemantauan

Layanan Terkelola Amazon untuk Prometheus adalah layanan pemantauan yang kompatibel dengan Prometheus untuk infrastruktur kontainer dan metrik aplikasi untuk kontainer yang dapat Anda gunakan untuk memantau lingkungan kontainer dengan aman dalam skala besar. Untuk informasi selengkapnya, lihat posting blog Memulai dengan Amazon Managed Service untuk Prometheus.

Amazon CloudWatch menyediakan pemantauan full-stack pada AWS. CloudWatch mendukung solusi AWS native dan open source sehingga Anda dapat memahami apa yang terjadi di seluruh tumpukan teknologi Anda kapan saja.

AWS Alat asli meliputi yang berikut:

- CloudWatch Dasbor Amazon
- CloudWatchWawasan Kontainer
- CloudWatch metrik
- CloudWatch alarm

Amazon CloudWatch menawarkan fitur yang dibuat khusus yang menangani kasus penggunaan tertentu seperti pemantauan kontainer melalui CloudWatch Wawasan Kontainer. Fitur-fitur ini dibangun CloudWatch sehingga Anda dapat mengatur log, pengumpulan metrik, dan pemantauan.

Untuk aplikasi kontainer dan layanan mikro Anda, gunakan Wawasan Kontainer untuk mengumpulkan, menggabungkan, dan meringkas metrik dan log. Container Insights tersedia untuk Amazon Elastic Container Service (Amazon ECS), Amazon Elastic Kubernetes Service (Amazon EKS), dan platform Kubernetes di Amazon Elastic Compute Cloud (Amazon EC2). Wawasan Kontainer mengumpulkan data sebagai peristiwa log kinerja dalam format metrik yang disematkan. Entri peristiwa log kinerja ini menggunakan skema JSON terstruktur yang mendukung konsumsi dan penyimpanan data kardinalitas tinggi dalam skala besar.

Untuk informasi tentang penerapan Wawasan Kontainer dengan Amazon EKS, lihat posting blog Memperkenalkan Amazon CloudWatch Container Insights untuk Amazon EKS Fargate menggunakan Distro untuk. AWS OpenTelemetry

Pelacakan

Penelusuran melibatkan penggunaan khusus informasi pencatatan tentang proses program. Wawasan dari log dapat membantu insinyur men-debug transaksi individu dan mengidentifikasi

kemacetan. Penelusuran dapat diaktifkan secara otomatis atau dengan menggunakan instrumentasi manual.

Karena aplikasi terintegrasi dengan layanan yang berbeda, penting untuk mengidentifikasi kinerja aplikasi dan layanan yang mendasarinya. Penelusuran bekerja dengan jejak dan bentang. Jejak adalah proses permintaan lengkap, dan setiap jejak terdiri dari bentang. Rentang adalah interval waktu yang ditandai dan merupakan aktivitas dalam komponen atau layanan individu sistem. Jejak memberikan gambaran besar tentang apa yang terjadi ketika permintaan dibuat ke aplikasi.

Tim aplikasi

Pengembang aplikasi instrumen aplikasi mereka dengan mengirimkan data jejak untuk permintaan masuk dan keluar dan peristiwa lain dalam aplikasi, bersama dengan metadata tentang setiap permintaan. Untuk menghasilkan jejak, aplikasi harus diinstrumentasi untuk menghasilkan jejak. Instrumentasi bisa otomatis atau manual.

Instrumentasi otomatis

Anda dapat mengumpulkan telemetri dari aplikasi dengan menggunakan instrumentasi otomatis tanpa harus memodifikasi kode sumber. Agen instrumentasi otomatis dapat menghasilkan jejak aplikasi aplikasi atau layanan. Biasanya, Anda menggunakan perubahan konfigurasi untuk menambahkan agen atau mekanisme lain.

Instrumentasi pustaka melibatkan membuat perubahan kode aplikasi minimal untuk menambahkan instrumentasi bawaan. Instrumentasi menargetkan pustaka atau kerangka kerja tertentu, seperti AWS SDK, klien HTTP Apache, atau klien SQL.

Instrumentasi manual

Dalam pendekatan ini, pengembang aplikasi menambahkan kode instrumentasi ke aplikasi di setiap lokasi di mana mereka ingin mengumpulkan informasi jejak. Misalnya, gunakan pemrograman berorientasi aspek (AOP) untuk mengumpulkan data penelusuran. AWS X-Ray Pengembang dapat SDKs menggunakan instrumen aplikasi mereka.

Pengambilan sampel

Data jejak sering dihasilkan dalam volume besar. Penting untuk memiliki mekanisme untuk menentukan apakah data jejak harus diekspor atau tidak. Sampling adalah proses menentukan data apa yang harus diekspor. Hal ini umumnya dilakukan untuk menghemat biaya. Dengan menyesuaikan aturan pengambilan sampel, Anda dapat mengontrol jumlah data yang Anda rekam.

Anda juga dapat mengubah perilaku pengambilan sampel tanpa mengubah dan menerapkan ulang kode Anda. Penting untuk mengontrol laju pengambilan sampel untuk menghasilkan jumlah jejak yang tepat.

Pengembang aplikasi dapat membuat anotasi jejak dengan menambahkan metadata sebagai pasangan nilai kunci. Anotasi memperkaya jejak dan membantu menyempurnakan pemfilteran di backend.

DevOps tim

DevOps Insinyur sering diminta untuk mengatur lingkungan penelusuran bagi pengembang aplikasi untuk memvisualisasikan jejak untuk infrastruktur dan aplikasi. Menelusuri pengaturan lingkungan melibatkan pengumpulan data jejak dari sumber yang berbeda dan mengirimkannya ke toko pusat untuk divisualisasikan.

Menelusuri backend

Backend penelusuran adalah layanan seperti yang mengumpulkan data tentang permintaan AWS X-Ray yang dilayani aplikasi Anda. Ini menyediakan alat yang dapat Anda gunakan untuk melihat, memfilter, dan mendapatkan wawasan tentang data tersebut untuk mengidentifikasi masalah dan peluang untuk pengoptimalan. Untuk setiap permintaan yang dilacak ke aplikasi Anda, Anda dapat melihat informasi terperinci tentang permintaan dan respons, dan tentang panggilan lain yang dilakukan aplikasi Anda ke AWS sumber daya hilir, layanan mikro, database, dan web. APIs

Mengotomatiskan penelusuran

Karena aplikasi yang berbeda memiliki persyaratan penelusuran yang berbeda, penting untuk mengotomatiskan konfigurasi dan pengoperasian infrastruktur penelusuran. Gunakan alat IAc untuk menyediakan backend infrastruktur penelusuran.

Gunakan saluran pipa CD untuk mengotomatiskan hal berikut:

- Terapkan infrastruktur penelusuran sesuai permintaan dan hancurkan saat tidak diperlukan.
- Terapkan konfigurasi penelusuran di seluruh aplikasi.

Alat penelusuran

AWS menyediakan layanan berikut untuk penelusuran dan visualisasi terkait:

- AWS X-Ray menerima jejak dari aplikasi Anda, selain jejak dari AWS layanan yang digunakan aplikasi Anda yang sudah terintegrasi dengan X-Ray. Ada beberapa agen SDKs, dan alat yang dapat digunakan untuk instrumen aplikasi Anda untuk penelusuran X-Ray. Lihat informasi yang lebih lengkap dalam dokumentasi AWS X-Ray.

  Pengembang juga dapat menggunakan AWS X-Ray SDKs untuk mengirim jejak ke X-Ray. AWS X-Ray menyediakan SDKs untukGo,Java, Node.js,Python, .NET, danRuby. Setiap X-Ray SDK menyediakan yang berikut:

  - Interceptors untuk ditambahkan ke kode Anda untuk melacak permintaan HTTP yang masuk
  - Penangan klien untuk instrumen klien AWS SDK yang digunakan aplikasi Anda untuk memanggil layanan lain AWS
  - Klien HTTP untuk panggilan instrumen ke layanan web HTTP internal dan eksternal lainnya

  X-Ray SDKs juga mendukung panggilan instrumentasi ke database SQL, instrumentasi klien AWS SDK otomatis, dan fitur lainnya. Daripada mengirim data pelacakan langsung ke X-Ray, SDK mengirim dokumen segmen JSON ke proses daemon yang mendengarkan lalu lintas UDP. X-Ray Daemon menyangga segmen dalam antrean dan mengunggah segmen tersebut ke X-Ray dalam batch. Untuk informasi selengkapnya tentang menginstrumentasi aplikasi Anda dengan menggunakan X-Ray SDK, lihat dokumentasi X-Ray.
- Amazon OpenSearch Service adalah layanan AWS terkelola untuk menjalankan dan menskalakan OpenSearch cluster, yang dapat digunakan untuk menyimpan log, metrik, dan jejak secara terpusat. Plugin Observability memberikan pengalaman terpadu untuk mengumpulkan dan memantau metrik, log, dan jejak dari sumber data umum. Pengumpulan dan pemantauan data di satu tempat menyediakan tumpukan penuh, end-to-end pengamatan seluruh infrastruktur Anda. Untuk informasi implementasi, lihat dokumentasi OpenSearch Layanan.
- AWS Distro for OpenTelemetry (ADOT) adalah AWS distribusi berdasarkan proyek Cloud Native Computing Foundation (CNCF). OpenTelemetry ADOT saat ini mencakup dukungan instrumentasi otomatis untuk Java dan Python. Selain itu, ADOT mendukung instrumentasi otomatis AWS Lambda fungsi dan permintaan hilirnya menggunakanJava, Node.js, dan Python runtime, melalui Lapisan Lambda Terkelola ADOT. Pengembang dapat menggunakan kolektor ADOT untuk mengirim jejak ke backend yang berbeda, termasuk dan AWS X-Ray Amazon Service. OpenSearch

  Untuk contoh referensi tentang cara instrumen aplikasi Anda menggunakan ADOT SDK, lihat dokumentasinya. Untuk contoh referensi tentang cara menggunakan SDK ADOT untuk mengirim data ke OpenSearch Layanan Amazon, lihat dokumentasi OpenSearch Layanan.

Untuk contoh referensi tentang cara instrumen aplikasi Anda berjalan di Amazon EKS, lihat posting blog Metrik dan koleksi jejak menggunakan add-on Amazon EKS untuk AWS Distro untuk. OpenTelemetry

Otomatisasi uji

Pengujian otomatis dengan kerangka kerja dan alat khusus dapat mengurangi intervensi manusia dan memaksimalkan kualitas. Pengujian kinerja otomatis tidak berbeda dengan pengujian otomatisasi seperti pengujian unit dan pengujian integrasi.

Gunakan DevOps saluran pipa dalam tahapan yang berbeda untuk pengujian kinerja.

Lima tahap untuk pipa otomatisasi uji adalah:

1. Siapkan — Gunakan pendekatan test-data yang dijelaskan di bagian Test-data generation untuk tahap ini. Menghasilkan data uji realistis sangat penting untuk mendapatkan hasil tes yang valid. Anda harus hati-hati membuat beragam data pengujian yang mencakup berbagai kasus penggunaan dan sangat cocok dengan data produksi langsung. Sebelum menjalankan pengujian kinerja skala penuh, Anda mungkin perlu menjalankan uji coba awal untuk memvalidasi skrip pengujian, lingkungan, dan alat pemantauan.
2. Alat uji — Untuk melakukan pengujian kinerja, pilih alat pengujian beban yang sesuai, seperti JMeter atau ghz. Pertimbangkan yang paling sesuai untuk kebutuhan bisnis Anda dalam hal mensimulasikan beban pengguna dunia nyata.
3. Uji coba — Dengan alat uji dan lingkungan yang ditetapkan, jalankan tes end-to-end kinerja di berbagai beban dan durasi pengguna yang diharapkan. Sepanjang tes, pantau dengan cermat kesehatan sistem yang sedang diuji. Ini biasanya merupakan tahap yang berjalan lama. Pantau tingkat kesalahan untuk pembatalan pengujian otomatis, dan hentikan pengujian jika ada terlalu banyak kesalahan.

   Alat pengujian beban memberikan wawasan tentang pemanfaatan sumber daya, waktu respons, dan potensi kemacetan.
4. Pelaporan pengujian - Kumpulkan hasil tes bersama dengan aplikasi dan konfigurasi pengujian. Mengotomatiskan pengumpulan konfigurasi aplikasi, konfigurasi pengujian, dan hasil, yang

membantu merekam data terkait pengujian kinerja dan menyimpannya secara terpusat. Mempertahankan data kinerja secara terpusat membantu memberikan wawasan yang baik dan mendukung penentuan kriteria keberhasilan secara terprogram untuk bisnis Anda.

5. Bersihkan - Setelah Anda menyelesaikan uji kinerja, setel ulang lingkungan pengujian dan data untuk mempersiapkan proses berikutnya. Pertama, Anda mengembalikan setiap perubahan yang dibuat pada data pengujian selama dijalankan. Anda harus mengembalikan database dan penyimpanan data lainnya ke keadaan semula, mengembalikan catatan baru, diperbarui, atau dihapus yang dihasilkan selama pengujian.

Anda dapat menggunakan kembali pipeline untuk mengulangi pengujian beberapa kali hingga hasilnya mencerminkan kinerja yang Anda inginkan. Anda juga dapat menggunakan pipeline untuk memvalidasi bahwa perubahan kode tidak merusak kinerja. Anda dapat menjalankan pengujian validasi kode di luar jam kerja dan menggunakan data pengujian dan observabilitas yang tersedia untuk pemecahan masalah.

Praktik terbaik meliputi:

- Rekam waktu mulai dan berakhir, dan buat secara otomatis URLs untuk pencatatan, Ini membantu Anda memfilter data pengamatan di jendela waktu yang tepat. pemantauan, dan sistem penelusuran.
- Suntikkan pengidentifikasi pengujian di header saat menjalankan tes. Pengembang aplikasi dapat memperkaya pencatatan, pemantauan, dan penelusuran data mereka dengan menggunakan pengenal sebagai filter di backend.
- Batasi pipa hanya satu kali pada satu waktu. Menjalankan pengujian bersamaan menghasilkan noise yang dapat menyebabkan kebingungan selama pemecahan masalah. Penting juga untuk menjalankan pengujian di lingkungan kinerja khusus.

Alat otomatisasi uji

Alat pengujian memainkan peran penting dalam otomatisasi pengujian apa pun. Pilihan populer untuk alat pengujian open source meliputi yang berikut:

- Apache JMeter adalah kuda kekuatan berpengalaman. Selama bertahun-tahun, Apache JMeter telah menjadi lebih andal dan telah menambahkan fitur. Dengan antarmuka grafis, Anda dapat membuat tes kompleks tanpa mengetahui bahasa pemrograman. Perusahaan seperti BlazeMeter mendukung Apache JMeter.

- K6 adalah alat gratis yang menawarkan dukungan, hosting sumber beban, dan antarmuka web terintegrasi untuk mengatur, menjalankan, dan menganalisis tes beban.
- Tes beban Vegeta mengikuti konsep yang berbeda. Alih-alih mendefinisikan konkurensi atau melempar beban ke sistem Anda, Anda menentukan tingkat tertentu. Alat ini kemudian membuat beban itu terlepas dari waktu respons sistem Anda.
- Hei dan ab, alat penandaan bangku server HTTP Apache, adalah alat dasar yang dapat Anda gunakan dari baris perintah untuk menjalankan beban yang ditentukan pada satu titik akhir. Ini adalah cara tercepat untuk menghasilkan beban jika Anda memiliki server untuk menjalankan alat. Bahkan laptop lokal akan bekerja, meskipun mungkin tidak cukup kuat untuk menghasilkan beban tinggi.
- ghz adalah utilitas baris perintah dan paket Go untuk pengujian beban dan layanan gRPC penandaan bangku.

AWS menyediakan Pengujian Beban Terdistribusi pada AWS solusi. Solusi ini menciptakan dan mensimulasikan ribuan pengguna yang terhubung menghasilkan catatan transaksional dengan kecepatan konstan tanpa perlu menyediakan server. Untuk informasi selengkapnya, lihat Perpustakaan AWS Solusi.

Anda dapat menggunakan AWS CodePipeline untuk mengotomatiskan pipa pengujian kinerja. Untuk informasi selengkapnya tentang mengotomatiskan pengujian API Anda dengan menggunakan CodePipeline, lihat AWS DevOps Blog dan AWS dokumentasinya.

Pelaporan uji

Pelaporan pengujian mengacu pada pengumpulan, analisis, dan penyajian data yang terkait dengan kinerja sistem, aplikasi, layanan, atau proses. Ini melibatkan pengukuran berbagai metrik dan indikator untuk menilai efisiensi, daya tanggap, keandalan, dan efektivitas keseluruhan sistem atau komponen tertentu.

Pelaporan uji kinerja melibatkan pemilihan metrik yang relevan berdasarkan konteks dan tujuan analisis. Metrik kinerja umum termasuk waktu respons, throughput, tingkat kesalahan, pemanfaatan sumber daya (CPU, memori, disk), dan latensi jaringan.

Setelah data terkait kinerja dikumpulkan, perlu disimpan di repositori pusat. Hasil pengujian ini dapat berasal dari lingkungan, aplikasi, dan alat pengujian yang berbeda. Ketika Anda memiliki beberapa beban kerja yang berjalan di lingkungan yang berbeda, sulit untuk mengumpulkan data terkait kinerja dan berkorelasi antara titik-titik data ini untuk menarik kesimpulan yang tepat. Kami

merekomendasikan untuk mendefinisikan metode standar untuk mengumpulkan data metrik kinerja menggunakan repositori pusat untuk penyimpanan dan visualisasi data.

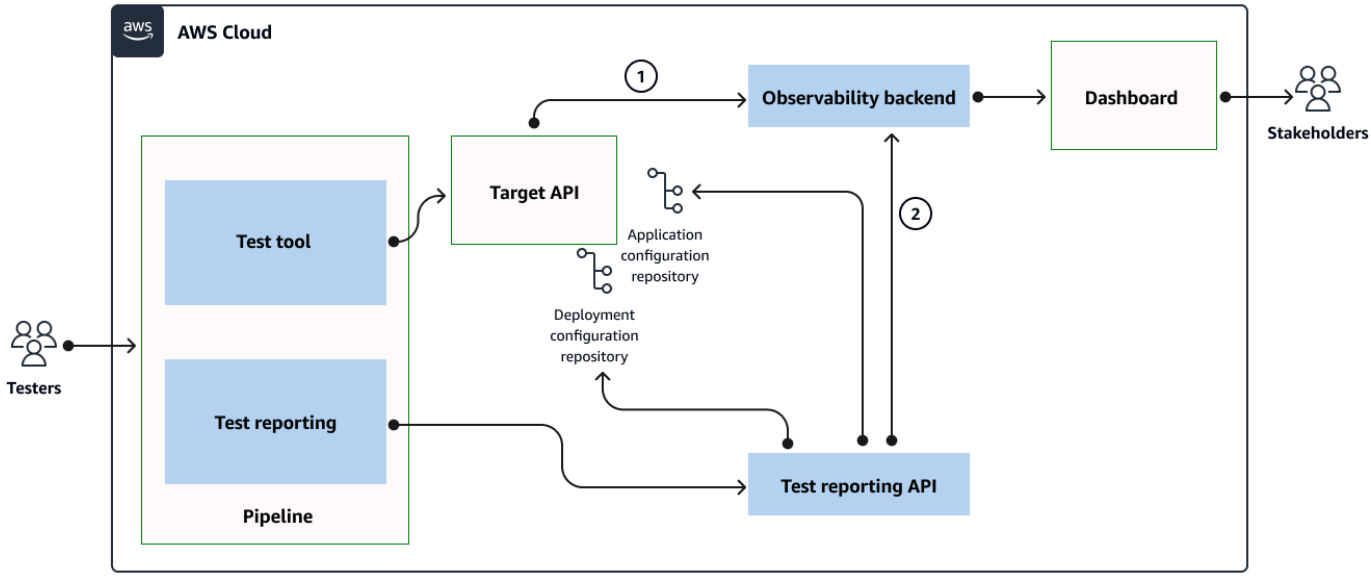
Rekaman standar

Kami merekomendasikan standarisasi cara pemangku kepentingan yang berbeda melakukan tes kinerja dan menulis data yang dihasilkan ke repositori pusat. Misalnya, ini bisa berbentuk API yang menerima hasil dan menyimpannya ke dalam solusi penyimpanan persisten. Dalam situasi di mana data perlu diambil dari sumber seperti GitOps atau Amazon Managed Service untuk Prometheus, API dapat langsung menarik detail tersebut dari sumber yang ditentukan berdasarkan file skema yang menjelaskan cara mengekstrak bidang dari spesifikasi penerapan dan spesifikasi Kubernetes. File skema dapat menggunakan `JSONPath` ekspresi atau Prometheus Query Language (PromQL). Seperti disebutkan sebelumnya, metrik yang dikumpulkan harus relevan dengan konteks dan tujuan analisis kinerja.

Data yang diteruskan ke API dapat mencakup detail dan tag yang terkait dengan aplikasi dan lingkungan tempat pengujian telah dilakukan. Ini membantu dengan melakukan analitik pada data pengujian kinerja.

Pilar rekayasa kinerja beraksi

Arsitektur referensi berikut menunjukkan pilar rekayasa kinerja untuk menguji API tertentu.



1. Pencatatan, pemantauan, dan penelusuran data dikirim dari API target ke backend.
2. Saat dipanggil, API pelaporan pengujian mengirimkan hasil dan informasi konfigurasi ke backend.

Komponen inti adalah API target atau aplikasi yang sedang diuji. API target disinkronkan dengan repositori konfigurasi aplikasi dan repositori konfigurasi penerapan dalam GitOps mode untuk mendapatkan konfigurasi aplikasi dan infrastruktur terbaru. Sinkronisasi ini memungkinkan pengujian otomatis berjalan terhadap keadaan aplikasi yang diinginkan saat ini dan infrastruktur pendukungnya seperti yang didefinisikan dalam repositori Git.

Pipeline otomatisasi pengujian mengotomatiskan pembuatan data pengujian, menjalankan pengujian, dan melaporkan hasil pengujian untuk API target.

API target menghasilkan wawasan kinerja (metrik, log, dan jejak), menggunakan praktik terbaik observabilitas, dan mengalirkan data metrik ke backend observabilitas.

API pelaporan pengujian mengumpulkan semua data pelaporan terkait pengujian (konfigurasi dan hasil pengujian), dan menyimpannya di backend observabilitas.

Agregasi wawasan kinerja dan data pelaporan (konfigurasi, hasil pengujian) membantu Anda melakukan kueri data terkait kinerja untuk API target. Misalnya, Anda mungkin menanyakan hal berikut:

- Apa sepuluh transaksi paling lambat?
- Berapa P99, P90, jumlah rata-rata setiap tes?
- Bagaimana konfigurasi dari dua pengujian berjalan dibandingkan?

Mengkorelasikan kasus pengujian dengan hasil, konfigurasi, dan metrik selama periode waktu tertentu membantu mengidentifikasi konfigurasi terbaik dan hasil kinerja.

Dengan menggunakan hasil pengujian ini, Anda dapat membuat keputusan berdasarkan data yang lebih tepat untuk API dan memiliki kepercayaan diri saat membawa API ke produksi.

Sumber daya

Layanan AWS

- Amazon CloudWatch
- AWS CodePipeline
- AWS Distro untuk OpenTelemetry
- OpenSearch Layanan Amazon
- AWS X-Ray

Implementasi

- amazon-kinesis-data-generator
- AWS Glue Uji Generator Data
- Pengujian Beban Terdistribusi pada AWS

Postingan blog

- Pencatatan Kontainer Terpusat dengan Bit Lancar
- Uji Solusi Data Streaming Anda dengan Generator Data Amazon Kinesis Baru
- Memperkenalkan Amazon CloudWatch Container Insights untuk Amazon EKS Fargate menggunakan Distro AWS untuk OpenTelemetry
- Penelusuran Aplikasi di Kubernetes dengan AWS X-Ray
- Pengumpulan metrik dan jejak menggunakan add-on Amazon EKS untuk Distro for AWS OpenTelemetry
- Memulai dengan Amazon Managed Service untuk Prometheus

Lokakarya

- Pengantar AWS Observabilitas

AWS Bimbingan Preskriptif

- Aplikasi pengujian beban (panduan)

Aplikasi pihak ketiga

- Apache JMeter
- K6
- Vegeta
- Hei dan ab
- ghz

Kontributor

Kontributor dokumen ini meliputi:

- Varun Sharma, Sr. Konsultan Utama, AWS
- Akash Kumar, Sr. Konsultan Utama, AWS
- Archana Bhatnagar, Manajer Praktek, AWS
- Pratik Sharma, Pelayanan Profesional II, AWS

Riwayat dokumen

Tabel berikut menjelaskan perubahan signifikan pada panduan ini. Jika Anda ingin diberi tahu tentang pembaruan masa depan, Anda dapat berlangganan umpan RSS.

Perubahan	Deskripsi	Tanggal
Publikasi awal	—	April 24, 2024

AWS Glosarium Panduan Preskriptif

Berikut ini adalah istilah yang umum digunakan dalam strategi, panduan, dan pola yang disediakan oleh Panduan AWS Preskriptif. Untuk menyarankan entri, silakan gunakan tautan Berikan umpan balik di akhir glosarium.

Nomor

7 Rs

Tujuh strategi migrasi umum untuk memindahkan aplikasi ke cloud. Strategi ini dibangun di atas 5 Rs yang diidentifikasi Gartner pada tahun 2011 dan terdiri dari yang berikut:

- Refactor/re-architect — Pindahkan aplikasi dan modifikasi arsitekturnya dengan memanfaatkan sepenuhnya fitur cloud-native untuk meningkatkan kelincahan, kinerja, dan skalabilitas. Ini biasanya melibatkan porting sistem operasi dan database. Contoh: Migrasikan database Oracle lokal Anda ke Amazon Aurora Edition. PostgreSQL-Compatible
- Replatform (angkat dan bentuk ulang) — Pindahkan aplikasi ke cloud, dan perkenalkan beberapa tingkat pengoptimalan untuk memanfaatkan kemampuan cloud. Contoh: Memigrasikan database Oracle lokal Anda ke Amazon Relational Database Service (Amazon RDS) untuk Oracle di. AWS Cloud
- Pembelian kembali (drop and shop) - Beralih ke produk yang berbeda, biasanya dengan beralih dari lisensi tradisional ke model SaaS. Contoh: Migrasikan sistem manajemen hubungan pelanggan (CRM) Anda ke. Salesforce.com
- Rehost (lift dan shift) — Pindahkan aplikasi ke cloud tanpa membuat perubahan apa pun untuk memanfaatkan kemampuan cloud. Contoh: Migrasikan database Oracle lokal Anda ke Oracle pada instans EC2 di. AWS Cloud
- Relokasi (hypervisor-level lift and shift) — Pindahkan infrastruktur ke cloud tanpa membeli perangkat keras baru, menulis ulang aplikasi, atau memodifikasi operasi yang ada. Anda memigrasikan server dari platform lokal ke layanan cloud untuk platform yang sama. Contoh: Migrasikan Microsoft Hyper-V aplikasi ke AWS.
- Pertahankan (kunjungi kembali) - Simpan aplikasi di lingkungan sumber Anda. Ini mungkin termasuk aplikasi yang memerlukan refactoring besar, dan Anda ingin menunda pekerjaan itu sampai nanti, dan aplikasi lama yang ingin Anda pertahankan, karena tidak ada pembenaran bisnis untuk memigrasikannya.

- Pensiun — Menonaktifkan atau menghapus aplikasi yang tidak lagi diperlukan di lingkungan sumber Anda.

A

A2A () Agent-to-Agent

Protokol stateful untuk kolaborasi agen-ke-agen yang mendukung delegasi tugas dan transfer negara.

ABAC

Lihat kontrol akses berbasis atribut.

layanan abstrak

Lihat layanan terkelola.

ASAM

Lihat atomisitas, konsistensi, isolasi, daya tahan.

migrasi aktif-aktif

Metode migrasi database di mana basis data sumber dan target tetap sinkron (dengan menggunakan alat replikasi dua arah atau operasi penulisan ganda), dan kedua database menangani transaksi dari menghubungkan aplikasi selama migrasi. Metode ini mendukung migrasi dalam batch kecil yang terkontrol alih-alih memerlukan pemotongan satu kali. Ini lebih fleksibel tetapi membutuhkan lebih banyak pekerjaan daripada migrasi aktif-pasif.

migrasi aktif-pasif

Metode migrasi database di mana database sumber dan target disimpan dalam sinkron, tetapi hanya database sumber yang menangani transaksi dari menghubungkan aplikasi sementara data direplikasi ke database target. Basis data target tidak menerima transaksi apa pun selama migrasi.

Agen

Sistem AI yang dapat secara mandiri bernalar, merencanakan, dan mengambil tindakan menggunakan alat untuk mencapai tujuan.

Agen Ops

Praktik operasional untuk membangun, menguji, menyebarkan, dan menjalankan agen AI dalam produksi dalam skala besar.

fungsi agregat

Fungsi SQL yang beroperasi pada sekelompok baris dan menghitung nilai pengembalian tunggal untuk grup. Contoh fungsi agregat meliputi `SUM` dan`MAX`.

AI

Lihat kecerdasan buatan.

AIOps

Lihat operasi kecerdasan buatan.

anonimisasi

Proses menghapus informasi pribadi secara permanen dalam kumpulan data. Anonimisasi dapat membantu melindungi privasi pribadi. Data anonim tidak lagi dianggap sebagai data pribadi.

anti-pola

Solusi yang sering digunakan untuk masalah berulang di mana solusinya kontra-produktif, tidak efektif, atau kurang efektif daripada alternatif.

kontrol aplikasi

Pendekatan keamanan yang memungkinkan penggunaan hanya aplikasi yang disetujui untuk membantu melindungi sistem dari malware.

portofolio aplikasi

Kumpulan informasi rinci tentang setiap aplikasi yang digunakan oleh organisasi, termasuk biaya untuk membangun dan memelihara aplikasi, dan nilai bisnisnya. Informasi ini adalah kunci untuk penemuan portofolio dan proses analisis dan membantu mengidentifikasi dan memprioritaskan aplikasi yang akan dimigrasi, dimodernisasi, dan dioptimalkan.

kecerdasan buatan (AI)

Bidang ilmu komputer yang didedikasikan untuk menggunakan teknologi komputasi untuk melakukan fungsi kognitif yang biasanya terkait dengan manusia, seperti belajar, memecahkan masalah, dan mengenali pola. Untuk informasi lebih lanjut, lihat Apa itu Kecerdasan Buatan?

operasi kecerdasan buatan (AIOps)

Proses menggunakan teknik pembelajaran mesin untuk memecahkan masalah operasional, mengurangi insiden operasional dan intervensi manusia, dan meningkatkan kualitas layanan. Untuk informasi selengkapnya tentang cara AIOps digunakan dalam strategi AWS migrasi, lihat panduan integrasi operasi.

enkripsi asimetris

Algoritma enkripsi yang menggunakan sepasang kunci, kunci publik untuk enkripsi dan kunci pribadi untuk dekripsi. Anda dapat berbagi kunci publik karena tidak digunakan untuk dekripsi, tetapi akses ke kunci pribadi harus sangat dibatasi.

atomisitas, konsistensi, isolasi, daya tahan (ACID)

Satu set properti perangkat lunak yang menjamin validitas data dan keandalan operasional database, bahkan dalam kasus kesalahan, kegagalan daya, atau masalah lainnya.

kontrol akses berbasis atribut (ABAC)

Praktik membuat izin berbutir halus berdasarkan atribut pengguna, seperti departemen, peran pekerjaan, dan nama tim. Untuk informasi selengkapnya, lihat ABAC untuk AWS dokumentasi AWS Identity and Access Management (IAM).

sumber data otoritatif

Lokasi di mana Anda menyimpan versi utama data, yang dianggap sebagai sumber informasi yang paling dapat diandalkan. Anda dapat menyalin data dari sumber data otoritatif ke lokasi lain untuk tujuan memproses atau memodifikasi data, seperti menganonimkan, menyunting, atau membuat nama samaran.

Zona Ketersediaan

Lokasi berbeda di dalam Wilayah AWS yang terisolasi dari kegagalan di Availability Zone lainnya dan menyediakan konektivitas jaringan latensi rendah yang murah ke Availability Zone lainnya di Wilayah yang sama.

AWS Kerangka Adopsi Cloud (AWS CAF)

Kerangka pedoman dan praktik terbaik AWS untuk membantu organisasi mengembangkan rencana yang efisien dan efektif untuk bergerak dengan sukses ke cloud. AWS CAF mengatur panduan ke dalam enam area fokus yang disebut perspektif: bisnis, orang, tata kelola, platform, keamanan, dan operasi. Perspektif bisnis, orang, dan tata kelola fokus pada keterampilan dan proses bisnis; perspektif platform, keamanan, dan operasi fokus pada keterampilan dan proses teknis. Misalnya, perspektif masyarakat menargetkan pemangku kepentingan yang menangani

sumber daya manusia (SDM), fungsi kepegawaian, dan manajemen orang. Untuk perspektif ini, AWS CAF memberikan panduan untuk pengembangan, pelatihan, dan komunikasi orang untuk membantu mempersiapkan organisasi untuk adopsi cloud yang sukses. Untuk informasi lebih lanjut, lihat situs web AWS CAF dan whitepaper AWS CAF.

AWS Kerangka Kualifikasi Beban Kerja (AWS WQF)

Alat yang mengevaluasi beban kerja migrasi database, merekomendasikan strategi migrasi, dan memberikan perkiraan kerja. AWS WQF disertakan dengan AWS Schema Conversion Tool ()AWS SCT. Ini menganalisis skema database dan objek kode, kode aplikasi, dependensi, dan karakteristik kinerja, dan memberikan laporan penilaian.

B

bot buruk

Bot yang dimaksudkan untuk mengganggu atau menyebabkan kerugian bagi individu atau organisasi.

BCP

Lihat perencanaan kontinuitas bisnis.

grafik perilaku

Pandangan interaktif yang terpadu tentang perilaku dan interaksi sumber daya dari waktu ke waktu. Anda dapat menggunakan grafik perilaku dengan Amazon Detective untuk memeriksa upaya logon yang gagal, panggilan API yang mencurigakan, dan tindakan serupa. Untuk informasi selengkapnya, lihat Data dalam grafik perilaku di dokumentasi Detektif.

sistem big-endian

Sistem yang menyimpan byte paling signifikan terlebih dahulu. Lihat juga endianness.

klasifikasi biner

Sebuah proses yang memprediksi hasil biner (salah satu dari dua kelas yang mungkin). Misalnya, model ML Anda mungkin perlu memprediksi masalah seperti “Apakah email ini spam atau bukan spam?” atau “Apakah produk ini buku atau mobil?”

filter mekar

Struktur data probabilistik dan efisien memori yang digunakan untuk menguji apakah suatu elemen adalah anggota dari suatu himpunan.

blue/green penyebaran

Strategi penyebaran tempat Anda membuat dua lingkungan yang terpisah namun identik. Anda menjalankan versi aplikasi saat ini di satu lingkungan (biru) dan versi aplikasi baru di lingkungan lain (hijau). Strategi ini membantu Anda dengan cepat memutar kembali dengan dampak minimal.

bot

Aplikasi perangkat lunak yang menjalankan tugas otomatis melalui internet dan mensimulasikan aktivitas atau interaksi manusia. Beberapa bot berguna atau bermanfaat, seperti perayap web yang mengindeks informasi di internet. Beberapa bot lain, yang dikenal sebagai bot buruk, dimaksudkan untuk mengganggu atau membahayakan individu atau organisasi.

botnet

Jaringan bot yang terinfeksi oleh malware dan berada di bawah kendali satu pihak, yang dikenal sebagai bot herder atau operator bot. Botnet adalah mekanisme paling terkenal untuk skala bot dan dampaknya.

cabang

Area berisi repositori kode. Cabang pertama yang dibuat dalam repositori adalah cabang utama. Anda dapat membuat cabang baru dari cabang yang ada, dan Anda kemudian dapat mengembangkan fitur atau memperbaiki bug di cabang baru. Cabang yang Anda buat untuk membangun fitur biasanya disebut sebagai cabang fitur. Saat fitur siap dirilis, Anda menggabungkan cabang fitur kembali ke cabang utama. Untuk informasi selengkapnya, lihat Tentang cabang (GitHub dokumentasi).

akses break-glass

Dalam keadaan luar biasa dan melalui proses yang disetujui, cara cepat bagi pengguna untuk mendapatkan akses ke Akun AWS yang biasanya tidak memiliki izin untuk mengaksesnya. Untuk informasi lebih lanjut, lihat indikator Implementasikan prosedur break-glass dalam panduan. AWS Well-Architected

strategi brownfield

Infrastruktur yang ada di lingkungan Anda. Saat mengadopsi strategi brownfield untuk arsitektur sistem, Anda merancang arsitektur di sekitar kendala sistem dan infrastruktur saat ini. Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan greenfield.

cache penyangga

Area memori tempat data yang paling sering diakses disimpan.

kemampuan bisnis

Apa yang dilakukan bisnis untuk menghasilkan nilai (misalnya, penjualan, layanan pelanggan, atau pemasaran). Arsitektur layanan mikro dan keputusan pengembangan dapat didorong oleh kemampuan bisnis. Untuk informasi selengkapnya, lihat bagian Terorganisir di sekitar kemampuan bisnis dari Menjalankan layanan mikro kontainer di whitepaper. AWS

perencanaan kelangsungan bisnis (BCP)

Rencana yang membahas dampak potensial dari peristiwa yang mengganggu, seperti migrasi skala besar, pada operasi dan memungkinkan bisnis untuk melanjutkan operasi dengan cepat.

C

KAFE

Lihat Kerangka Adopsi AWS Cloud.

penyebaran kenari

Rilis versi yang lambat dan bertahap untuk pengguna akhir. Ketika Anda yakin, Anda menyebarkan versi baru dan mengganti versi saat ini secara keseluruhan.

CCoE

Lihat Cloud Center of Excellence.

CDC

Lihat mengubah pengambilan data.

ubah pengambilan data (CDC)

Proses melacak perubahan ke sumber data, seperti tabel database, dan merekam metadata tentang perubahan tersebut. Anda dapat menggunakan CDC untuk berbagai tujuan, seperti mengaudit atau mereplikasi perubahan dalam sistem target untuk mempertahankan sinkronisasi.

rekayasa kekacauan

Sengaja memperkenalkan kegagalan atau peristiwa yang mengganggu untuk menguji ketahanan sistem. Anda dapat menggunakan AWS Fault Injection Service (AWS FIS) untuk melakukan eksperimen yang menekankan AWS beban kerja Anda dan mengevaluasi responsnya.

CI/CD

Lihat integrasi berkelanjutan dan pengiriman berkelanjutan.

klasifikasi

Proses kategorisasi yang membantu menghasilkan prediksi. Model ML untuk masalah klasifikasi memprediksi nilai diskrit. Nilai diskrit selalu berbeda satu sama lain. Misalnya, model mungkin perlu mengevaluasi apakah ada mobil dalam gambar atau tidak.

Pengembang Warga

Pengguna bisnis yang membuat aplikasi AI menggunakan platform tanpa code/low kode tanpa keterampilan teknis khusus.

Enkripsi sisi klien

Enkripsi data secara lokal, sebelum target Layanan AWS menerimanya.

Cloud Center of Excellence (CCoE)

Tim multi-disiplin yang mendorong upaya adopsi cloud di seluruh organisasi, termasuk mengembangkan praktik terbaik cloud, memobilisasi sumber daya, menetapkan jadwal migrasi, dan memimpin organisasi melalui transformasi skala besar. Untuk informasi selengkapnya, lihat posting CCoE di Blog Strategi AWS Cloud Perusahaan.

komputasi cloud

Teknologi cloud yang biasanya digunakan untuk penyimpanan data jarak jauh dan manajemen perangkat IoT. Cloud computing umumnya terhubung ke teknologi edge computing.

model operasi cloud

Dalam organisasi TI, model operasi yang digunakan untuk membangun, mematangkan, dan mengoptimalkan satu atau lebih lingkungan cloud. Untuk informasi selengkapnya, lihat Membangun Model Operasi Cloud Anda.

tahap adopsi cloud

Empat fase yang biasanya dilalui organisasi ketika mereka bermigrasi ke AWS Cloud:

- Proyek — Menjalankan beberapa proyek terkait cloud untuk bukti konsep dan tujuan pembelajaran
- Foundation — Melakukan investasi dasar untuk meningkatkan adopsi cloud Anda (misalnya, membuat landing zone, mendefinisikan CCoE, membuat model operasi)
- Migrasi — Migrasi aplikasi individual
- Re-invention — Mengoptimalkan produk dan layanan, dan berinovasi di cloud

Tahapan ini didefinisikan oleh Stephen Orban dalam posting blog The Journey Toward Cloud-First & the Stages of Adoption di blog Strategi AWS Cloud Perusahaan. Untuk informasi tentang bagaimana kaitannya dengan strategi AWS migrasi, lihat panduan kesiapan migrasi.

CMDB

Lihat database manajemen konfigurasi.

repositori kode

Lokasi di mana kode sumber dan aset lainnya, seperti dokumentasi, sampel, dan skrip, disimpan dan diperbarui melalui proses kontrol versi. Repositori cloud umum termasuk GitHub atau. Bitbucket Cloud Setiap versi kode disebut cabang. Dalam struktur layanan mikro, setiap repositori dikhususkan untuk satu bagian fungsionalitas. Satu CI/CD pipa dapat menggunakan beberapa repositori.

cache dingin

Cache buffer yang kosong, tidak terisi dengan baik, atau berisi data basi atau tidak relevan. Ini mempengaruhi kinerja karena instance database harus membaca dari memori utama atau disk, yang lebih lambat daripada membaca dari cache buffer.

data dingin

Data yang jarang diakses dan biasanya historis. Saat menanyakan jenis data ini, kueri lambat biasanya dapat diterima. Memindahkan data ini ke tingkat atau kelas penyimpanan yang berkinerja lebih rendah dan lebih murah dapat mengurangi biaya.

visi komputer (CV)

Bidang AI yang menggunakan pembelajaran mesin untuk menganalisis dan mengekstrak informasi dari format visual seperti gambar dan video digital. Misalnya, Amazon SageMaker AI menyediakan algoritma pemrosesan gambar untuk CV.

konfigurasi drift

Untuk beban kerja, konfigurasi berubah dari status yang diharapkan. Ini dapat menyebabkan beban kerja menjadi tidak patuh, dan biasanya bertahap dan tidak disengaja.

database manajemen konfigurasi (CMDB)

Repositori yang menyimpan dan mengelola informasi tentang database dan lingkungan TI, termasuk komponen perangkat keras dan perangkat lunak dan konfigurasinya. Anda biasanya menggunakan data dari CMDB dalam penemuan portofolio dan tahap analisis migrasi.

paket kesesuaian

Kumpulan AWS Config aturan dan tindakan remediasi yang dapat Anda kumpulkan untuk menyesuaikan kepatuhan dan pemeriksaan keamanan Anda. Anda dapat menerapkan paket kesesuaian sebagai entitas tunggal di Akun AWS dan Region, atau di seluruh organisasi, dengan menggunakan templat YAMM. Untuk informasi selengkapnya, lihat Paket kesesuaian dalam dokumentasi. AWS Config

integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD)

Proses mengotomatiskan sumber, membangun, menguji, pementasan, dan tahap produksi dari proses rilis perangkat lunak. CI/CD biasanya digambarkan sebagai pipa. CI/CD dapat membantu Anda mengotomatiskan proses, meningkatkan produktivitas, meningkatkan kualitas kode, dan memberikan lebih cepat. Untuk informasi lebih lanjut, lihat Manfaat pengiriman berkelanjutan. CD juga dapat berarti penerapan berkelanjutan. Untuk informasi selengkapnya, lihat Continuous Delivery vs Continuous Deployment.

CV

Lihat visi komputer.

D

data saat istirahat

Data yang stasioner di jaringan Anda, seperti data yang ada di penyimpanan.

klasifikasi data

Proses untuk mengidentifikasi dan mengkategorikan data dalam jaringan Anda berdasarkan kekritisan dan sensitivitasnya. Ini adalah komponen penting dari setiap strategi manajemen risiko keamanan siber karena membantu Anda menentukan perlindungan dan kontrol retensi yang tepat untuk data. Klasifikasi data adalah komponen pilar keamanan dalam AWS Well-Architected Framework. Untuk informasi selengkapnya, lihat Klasifikasi data.

penyimpangan data

Variasi yang berarti antara data produksi dan data yang digunakan untuk melatih model ML, atau perubahan yang berarti dalam data input dari waktu ke waktu. Penyimpangan data dapat mengurangi kualitas, akurasi, dan keadilan keseluruhan dalam prediksi model ML.

data dalam transit

Data yang aktif bergerak melalui jaringan Anda, seperti antara sumber daya jaringan.

jala data

Kerangka arsitektur yang menyediakan kepemilikan data terdistribusi dan terdesentralisasi dengan manajemen dan tata kelola terpusat.

minimalisasi data

Prinsip pengumpulan dan pemrosesan hanya data yang sangat diperlukan. Mempraktikkan minimalisasi data di dalamnya AWS Cloud dapat mengurangi risiko privasi, biaya, dan jejak karbon analitik Anda.

perimeter data

Satu set pagar pembatas pencegahan di AWS lingkungan Anda yang membantu memastikan bahwa hanya identitas tepercaya yang mengakses sumber daya tepercaya dari jaringan yang diharapkan. Untuk informasi selengkapnya, lihat Membangun perimeter data pada AWS.

prapemrosesan data

Untuk mengubah data mentah menjadi format yang mudah diuraikan oleh model ML Anda. Preprocessing data dapat berarti menghapus kolom atau baris tertentu dan menangani nilai yang hilang, tidak konsisten, atau duplikat.

asal data

Proses melacak asal dan riwayat data sepanjang siklus hidupnya, seperti bagaimana data dihasilkan, ditransmisikan, dan disimpan.

subjek data

Individu yang datanya dikumpulkan dan diproses.

gudang data

Sistem manajemen data yang mendukung intelijen bisnis, seperti analitik. Gudang data biasanya berisi sejumlah besar data historis, dan biasanya digunakan untuk kueri dan analisis.

bahasa definisi database (DDL)

Pernyataan atau perintah untuk membuat atau memodifikasi struktur tabel dan objek dalam database.

bahasa manipulasi basis data (DHTML)

Pernyataan atau perintah untuk memodifikasi (memasukkan, memperbarui, dan menghapus) informasi dalam database.

DDL

Lihat bahasa definisi database.

ansambel yang dalam

Untuk menggabungkan beberapa model pembelajaran mendalam untuk prediksi. Anda dapat menggunakan ansambel dalam untuk mendapatkan prediksi yang lebih akurat atau untuk memperkirakan ketidakpastian dalam prediksi.

pembelajaran mendalam

Subbidang ML yang menggunakan beberapa lapisan jaringan saraf tiruan untuk mengidentifikasi pemetaan antara data input dan variabel target yang diinginkan.

pertahanan-mendalam

Pendekatan keamanan informasi di mana serangkaian mekanisme dan kontrol keamanan dilapisi dengan cermat di seluruh jaringan komputer untuk melindungi kerahasiaan, integritas, dan ketersediaan jaringan dan data di dalamnya. Saat Anda mengadopsi strategi ini AWS, Anda menambahkan beberapa kontrol pada lapisan AWS Organizations struktur yang berbeda untuk membantu mengamankan sumber daya. Misalnya, pendekatan defense-in-depth mungkin menggabungkan otentikasi multi-faktor, segmentasi jaringan, dan enkripsi.

administrator yang didelegasikan

Di AWS Organizations, layanan yang kompatibel dapat mendaftarkan akun AWS anggota untuk mengelola akun organisasi dan mengelola izin untuk layanan tersebut. Akun ini disebut administrator yang didelegasikan untuk layanan itu. Untuk informasi selengkapnya dan daftar layanan yang kompatibel, lihat Layanan yang berfungsi dengan AWS Organizations AWS Organizations dokumentasi.

deployment

Proses pembuatan aplikasi, fitur baru, atau perbaikan kode tersedia di lingkungan target. Deployment melibatkan penerapan perubahan dalam basis kode dan kemudian membangun dan menjalankan basis kode itu di lingkungan aplikasi.

lingkungan pengembangan

Lihat lingkungan.

kontrol detektif

Kontrol keamanan yang dirancang untuk mendeteksi, mencatat, dan memperingatkan setelah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan kedua, memperingatkan Anda tentang peristiwa keamanan yang melewati kontrol pencegahan yang ada. Untuk informasi selengkapnya, lihat Kontrol Detektif dalam Menerapkan kontrol keamanan pada. AWS

pemetaan aliran nilai pengembangan (DVSM)

Sebuah proses yang digunakan untuk mengidentifikasi dan memprioritaskan kendala yang mempengaruhi kecepatan dan kualitas dalam siklus hidup pengembangan perangkat lunak. DVSM memperluas proses pemetaan aliran nilai yang awalnya dirancang untuk praktik manufaktur ramping. Ini berfokus pada langkah-langkah dan tim yang diperlukan untuk menciptakan dan memindahkan nilai melalui proses pengembangan perangkat lunak.

kembar digital

Representasi virtual dari sistem dunia nyata, seperti bangunan, pabrik, peralatan industri, atau jalur produksi. Kembar digital mendukung pemeliharaan prediktif, pemantauan jarak jauh, dan optimalisasi produksi.

tabel dimensi

Dalam skema bintang, tabel yang lebih kecil yang berisi atribut data tentang data kuantitatif dalam tabel fakta. Atribut tabel dimensi biasanya bidang teks atau angka diskrit yang berperilaku seperti teks. Atribut ini biasanya digunakan untuk pembatasan kueri, pemfilteran, dan pelabelan set hasil.

musibah

Peristiwa yang mencegah beban kerja atau sistem memenuhi tujuan bisnisnya di lokasi utama yang digunakan. Peristiwa ini dapat berupa bencana alam, kegagalan teknis, atau akibat dari tindakan manusia, seperti kesalahan konfigurasi yang tidak disengaja atau serangan malware.

pemulihan bencana (DR)

Strategi dan proses yang Anda gunakan untuk meminimalkan downtime dan kehilangan data yang disebabkan oleh bencana. Untuk informasi selengkapnya, lihat Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML~

Lihat bahasa manipulasi database.

desain berbasis domain

Pendekatan untuk mengembangkan sistem perangkat lunak yang kompleks dengan menghubungkan komponennya ke domain yang berkembang, atau tujuan bisnis inti, yang dilayani setiap komponen. Konsep ini diperkenalkan oleh Eric Evans dalam bukunya, Domain-Driven Design: Tackling Complexity in the Heart of Software (Boston: Addison-Wesley Professional, 2003). Untuk informasi tentang cara menggunakan desain berbasis domain dengan pola gambar pencekik, lihat Memodernisasi layanan web Microsoft ASP.NET (ASMX) lama secara bertahap menggunakan container dan Amazon API Gateway.

DR

Lihat pemulihan bencana.

deteksi drift

Melacak penyimpangan dari konfigurasi dasar. Misalnya, Anda dapat menggunakan AWS CloudFormation untuk mendeteksi penyimpangan dalam sumber daya sistem, atau Anda dapat menggunakannya AWS Control Tower untuk mendeteksi perubahan di landing zone yang mungkin memengaruhi kepatuhan terhadap persyaratan tata kelola.

DVSM

Lihat pemetaan aliran nilai pengembangan.

E

EDA

Lihat analisis data eksplorasi.

EDI

Lihat pertukaran data elektronik.

komputasi tepi

Teknologi yang meningkatkan daya komputasi untuk perangkat pintar di tepi jaringan IoT. Jika dibandingkan dengan komputasi awan, komputasi tepi dapat mengurangi latensi komunikasi dan meningkatkan waktu respons.

pertukaran data elektronik (EDI)

Pertukaran otomatis dokumen bisnis antar organisasi. Untuk informasi selengkapnya, lihat Apa itu Pertukaran Data Elektronik.

enkripsi

Proses komputasi yang mengubah data plaintext, yang dapat dibaca manusia, menjadi ciphertext.

kunci enkripsi

String kriptografi dari bit acak yang dihasilkan oleh algoritma enkripsi. Panjang kunci dapat bervariasi, dan setiap kunci dirancang agar tidak dapat diprediksi dan unik.

endianness

Urutan byte disimpan dalam memori komputer. Big-endian sistem menyimpan byte paling signifikan terlebih dahulu. Little-endian sistem menyimpan byte paling tidak signifikan terlebih dahulu.

titik akhir

Lihat titik akhir layanan.

layanan endpoint

Layanan yang dapat Anda host di cloud pribadi virtual (VPC) untuk dibagikan dengan pengguna lain. Anda dapat membuat layanan endpoint dengan AWS PrivateLink dan memberikan izin kepada prinsipal lain Akun AWS atau ke AWS Identity and Access Management (IAM). Akun atau prinsipal ini dapat terhubung ke layanan endpoint Anda secara pribadi dengan membuat titik akhir VPC antarmuka. Untuk informasi selengkapnya, lihat Membuat layanan titik akhir di dokumentasi Amazon Virtual Private Cloud (Amazon VPC).

perencanaan sumber daya perusahaan (ERP)

Sistem yang mengotomatiskan dan mengelola proses bisnis utama (seperti akuntansi, MES, dan manajemen proyek) untuk suatu perusahaan.

enkripsi amplop

Proses mengenkripsi kunci enkripsi dengan kunci enkripsi lain. Untuk informasi selengkapnya, lihat Enkripsi amplop dalam dokumentasi AWS Key Management Service (AWS KMS).

lingkungan

Sebuah contoh dari aplikasi yang sedang berjalan. Berikut ini adalah jenis lingkungan yang umum dalam komputasi awan:

- Development Environment — Sebuah contoh dari aplikasi yang berjalan yang hanya tersedia untuk tim inti yang bertanggung jawab untuk memelihara aplikasi. Lingkungan pengembangan digunakan untuk menguji perubahan sebelum mempromosikannya ke lingkungan atas. Jenis lingkungan ini kadang-kadang disebut sebagai lingkungan pengujian.

- lingkungan yang lebih rendah — Semua lingkungan pengembangan untuk aplikasi, seperti yang digunakan untuk build awal dan pengujian.
- lingkungan produksi — Sebuah contoh dari aplikasi yang berjalan yang dapat diakses oleh pengguna akhir. Dalam sebuah CI/CD pipeline, lingkungan produksi adalah lingkungan penyebaran terakhir.
- lingkungan atas — Semua lingkungan yang dapat diakses oleh pengguna selain tim pengembangan inti. Ini dapat mencakup lingkungan produksi, lingkungan praproduksi, dan lingkungan untuk pengujian penerimaan pengguna.

epik

Dalam metodologi tangkas, kategori fungsional yang membantu mengatur dan memprioritaskan pekerjaan Anda. Epik memberikan deskripsi tingkat tinggi tentang persyaratan dan tugas implementasi. Misalnya, epos keamanan AWS CAF mencakup manajemen identitas dan akses, kontrol detektif, keamanan infrastruktur, perlindungan data, dan respons insiden. Untuk informasi selengkapnya tentang epos dalam strategi AWS migrasi, lihat panduan implementasi program.

ERP

Lihat perencanaan sumber daya perusahaan.

analisis data eksplorasi (EDA)

Proses menganalisis dataset untuk memahami karakteristik utamanya. Anda mengumpulkan atau mengumpulkan data dan kemudian melakukan penyelidikan awal untuk menemukan pola, mendeteksi anomali, dan memeriksa asumsi. EDA dilakukan dengan menghitung statistik ringkasan dan membuat visualisasi data.

F

tabel fakta

Tabel tengah dalam skema bintang. Ini menyimpan data kuantitatif tentang operasi bisnis. Biasanya, tabel fakta berisi dua jenis kolom: kolom yang berisi ukuran dan yang berisi kunci asing ke tabel dimensi.

gagal cepat

Filosofi yang menggunakan pengujian yang sering dan bertahap untuk mengurangi siklus hidup pengembangan. Ini adalah bagian penting dari pendekatan tangkas.

batas isolasi kesalahan

Dalam AWS Cloud, batas seperti Availability Zone, Wilayah AWS, control plane, atau data plane yang membatasi efek kegagalan dan membantu meningkatkan ketahanan beban kerja. Untuk informasi selengkapnya, lihat Batas Isolasi AWS Kesalahan.

cabang fitur

Lihat cabang.

fitur

Data input yang Anda gunakan untuk membuat prediksi. Misalnya, dalam konteks manufaktur, fitur bisa berupa gambar yang diambil secara berkala dari lini manufaktur.

pentingnya fitur

Seberapa signifikan fitur untuk prediksi model. Ini biasanya dinyatakan sebagai skor numerik yang dapat dihitung melalui berbagai teknik, seperti Shapley Additive Explanations (SHAP) dan gradien terintegrasi. Untuk informasi lebih lanjut, lihat Interpretabilitas model pembelajaran mesin dengan. AWS

transformasi fitur

Untuk mengoptimalkan data untuk proses ML, termasuk memperkaya data dengan sumber tambahan, menskalakan nilai, atau mengekstrak beberapa set informasi dari satu bidang data. Hal ini memungkinkan model ML untuk mendapatkan keuntungan dari data. Misalnya, jika Anda memecah tanggal “2021-05-27 00:15:37" menjadi “2021", “Mei”, “Kamis”, dan “15", Anda dapat membantu algoritme pembelajaran mempelajari pola bernuansa yang terkait dengan komponen data yang berbeda.

beberapa tembakan mendorong

Menyediakan LLM dengan sejumlah kecil contoh yang menunjukkan tugas dan output yang diinginkan sebelum memintanya untuk melakukan tugas serupa. Teknik ini adalah aplikasi pembelajaran dalam konteks, di mana model belajar dari contoh (bidikan) yang tertanam dalam petunjuk. Few-shot prompt bisa efektif untuk tugas-tugas yang membutuhkan pemformatan, penalaran, atau pengetahuan domain tertentu. Lihat juga bidikan nol.

FGAC

Lihat kontrol akses berbutir halus.

kontrol akses berbutir halus (FGAC)

Penggunaan beberapa kondisi untuk mengizinkan atau menolak permintaan akses.

migrasi flash-cut

Metode migrasi database yang menggunakan replikasi data berkelanjutan melalui pengambilan data perubahan untuk memigrasikan data dalam waktu sesingkat mungkin, alih-alih menggunakan pendekatan bertahap. Tujuannya adalah untuk menjaga downtime seminimal mungkin.

FM

Lihat model pondasi.

model pondasi (FM)

Jaringan saraf pembelajaran mendalam yang besar yang telah melatih kumpulan data besar-besaran data umum dan tidak berlabel. FM mampu melakukan berbagai tugas umum, seperti memahami bahasa, menghasilkan teks dan gambar, dan berbicara dalam bahasa alami. Untuk informasi selengkapnya, lihat Apa itu Model Foundation.

Gerbang FM

Perantara terpusat yang mengontrol dan menormalkan akses ke model pondasi. Juga dikenal sebagai gateway LLM.

G

AI generatif

Subset model AI yang telah dilatih pada sejumlah besar data dan yang dapat menggunakan prompt teks sederhana untuk membuat konten dan artefak baru, seperti gambar, video, teks, dan audio. Untuk informasi lebih lanjut, lihat Apa itu AI Generatif.

pemblokiran geografis

Lihat pembatasan geografis.

pembatasan geografis (pemblokiran geografis)

Di Amazon CloudFront, opsi untuk mencegah pengguna di negara tertentu mengakses distribusi konten. Anda dapat menggunakan daftar izinkan atau daftar blokir untuk menentukan negara yang disetujui dan dilarang. Untuk informasi selengkapnya, lihat Membatasi distribusi geografis konten Anda dalam dokumentasi. CloudFront

Alur kerja Gitflow

Pendekatan di mana lingkungan bawah dan atas menggunakan cabang yang berbeda dalam repositori kode sumber. Alur kerja Gitflow dianggap warisan, dan alur kerja berbasis batang adalah pendekatan modern yang lebih disukai.

gambar emas

Sebuah snapshot dari sistem atau perangkat lunak yang digunakan sebagai template untuk menyebarkan instance baru dari sistem atau perangkat lunak itu. Misalnya, di bidang manufaktur, gambar emas dapat digunakan untuk menyediakan perangkat lunak pada beberapa perangkat dan membantu meningkatkan kecepatan, skalabilitas, dan produktivitas dalam operasi manufaktur perangkat.

strategi greenfield

Tidak adanya infrastruktur yang ada di lingkungan baru. Saat mengadopsi strategi greenfield untuk arsitektur sistem, Anda dapat memilih semua teknologi baru tanpa batasan kompatibilitas dengan infrastruktur yang ada, juga dikenal sebagai brownfield. Jika Anda memperluas infrastruktur yang ada, Anda dapat memadukan strategi brownfield dan greenfield.

pagar pembatas

Aturan tingkat tinggi yang membantu mengatur sumber daya, kebijakan, dan kepatuhan di seluruh unit organisasi (OU). Pagar pembatas preventif menegakkan kebijakan untuk memastikan keselarasan dengan standar kepatuhan. Mereka diimplementasikan dengan menggunakan kebijakan kontrol layanan dan batas izin IAM. Detective guardrails mendeteksi pelanggaran kebijakan dan masalah kepatuhan, dan menghasilkan peringatan untuk remediasi. Mereka diimplementasikan dengan menggunakan AWS Config, AWS Security Hub CSPM, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector, dan pemeriksaan khusus AWS Lambda .

pagar pembatas (AI)

Mekanisme keamanan yang menyaring, memvalidasi, dan membatasi input dan output agen untuk membantu memastikan perilaku AI yang bertanggung jawab dan aman.

H

HA

Lihat ketersediaan tinggi.

migrasi database heterogen

Memigrasi database sumber Anda ke database target yang menggunakan mesin database yang berbeda (misalnya, Oracle ke Amazon Aurora). Migrasi heterogen biasanya merupakan bagian dari upaya arsitektur ulang, dan mengubah skema dapat menjadi tugas yang kompleks. AWS menyediakan AWS SCT yang membantu dengan konversi skema.

ketersediaan tinggi (HA)

Kemampuan beban kerja untuk beroperasi terus menerus, tanpa intervensi, jika terjadi tantangan atau bencana. Sistem HA dirancang untuk gagal secara otomatis, secara konsisten memberikan kinerja berkualitas tinggi, dan menangani beban dan kegagalan yang berbeda dengan dampak kinerja minimal.

modernisasi sejarawan

Pendekatan yang digunakan untuk memodernisasi dan meningkatkan sistem teknologi operasional (OT) untuk melayani kebutuhan industri manufaktur dengan lebih baik. Sejarawan adalah jenis database yang digunakan untuk mengumpulkan dan menyimpan data dari berbagai sumber di pabrik.

data penahanan

Sebagian dari data historis berlabel yang ditahan dari kumpulan data yang digunakan untuk melatih model pembelajaran mesin. Anda dapat menggunakan data penahanan untuk mengevaluasi kinerja model dengan membandingkan prediksi model dengan data penahanan.

manusia-dalam-lingkaran (HiTL)

Pola alur kerja di mana eksekusi agen berhenti untuk peninjauan dan persetujuan manusia pada titik keputusan kritis.

migrasi database homogen

Memigrasi database sumber Anda ke database target yang berbagi mesin database yang sama (misalnya, Microsoft SQL Server ke Amazon RDS for SQL Server). Migrasi homogen biasanya merupakan bagian dari upaya rehosting atau replatforming. Anda dapat menggunakan utilitas database asli untuk memigrasi skema.

data panas

Data yang sering diakses, seperti data real-time atau data translasi terbaru. Data ini biasanya memerlukan tingkat atau kelas penyimpanan berkinerja tinggi untuk memberikan respons kueri yang cepat.

perbaikan terbaru

Perbaikan mendesak untuk masalah kritis dalam lingkungan produksi. Karena urgensinya, perbaikan terbaru biasanya dibuat di luar alur kerja DevOps rilis biasa.

periode hypercare

Segera setelah cutover, periode waktu ketika tim migrasi mengelola dan memantau aplikasi yang dimigrasi di cloud untuk mengatasi masalah apa pun. Biasanya, periode ini panjangnya 1-4 hari. Pada akhir periode hypercare, tim migrasi biasanya mentransfer tanggung jawab untuk aplikasi ke tim operasi cloud.

I

IAc

Lihat infrastruktur sebagai kode.

kebijakan berbasis identitas

Kebijakan yang dilampirkan pada satu atau beberapa prinsip IAM yang mendefinisikan izin mereka dalam lingkungan. AWS Cloud

aplikasi idle

Aplikasi yang memiliki penggunaan CPU dan memori rata-rata antara 5 dan 20 persen selama periode 90 hari. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini atau mempertahankannya di tempat.

IIoT

Lihat Internet of Things industri.

infrastruktur yang tidak dapat diubah

Model yang menyebarkan infrastruktur baru untuk beban kerja produksi alih-alih memperbarui, menambal, atau memodifikasi infrastruktur yang ada. Infrastruktur yang tidak dapat diubah secara inheren lebih konsisten, andal, dan dapat diprediksi daripada infrastruktur yang dapat berubah. Untuk informasi selengkapnya, lihat praktik terbaik Deploy using immutable infrastructure in the Framework. AWS Well-Architected

masuk (masuknya) VPC

Dalam arsitektur AWS multi-akun, VPC yang menerima, memeriksa, dan merutekan koneksi jaringan dari luar aplikasi. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan

akun Jaringan Anda dengan VPC masuk, keluar, dan inspeksi untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

migrasi inkremental

Strategi cutover di mana Anda memigrasikan aplikasi Anda dalam bagian-bagian kecil alih-alih melakukan satu cutover penuh. Misalnya, Anda mungkin hanya memindahkan beberapa layanan mikro atau pengguna ke sistem baru pada awalnya. Setelah Anda memverifikasi bahwa semuanya berfungsi dengan baik, Anda dapat secara bertahap memindahkan layanan mikro atau pengguna tambahan hingga Anda dapat menonaktifkan sistem lama Anda. Strategi ini mengurangi risiko yang terkait dengan migrasi besar.

Industri 4.0

Sebuah istilah yang diperkenalkan oleh Klaus Schwab pada tahun 2016 untuk merujuk pada modernisasi proses manufaktur melalui kemajuan dalam konektivitas, data real-time, otomatisasi, analitik, dan. AI/ML

infrastruktur

Semua sumber daya dan aset yang terkandung dalam lingkungan aplikasi.

infrastruktur sebagai kode (IAc)

Proses penyediaan dan pengelolaan infrastruktur aplikasi melalui satu set file konfigurasi. IAc dirancang untuk membantu Anda memusatkan manajemen infrastruktur, menstandarisasi sumber daya, dan menskalakan dengan cepat sehingga lingkungan baru dapat diulang, andal, dan konsisten.

Internet of Things industri (IIoT)

Penggunaan sensor dan perangkat yang terhubung ke internet di sektor industri, seperti manufaktur, energi, otomotif, perawatan kesehatan, ilmu kehidupan, dan pertanian. Untuk informasi selengkapnya, lihat Membangun strategi transformasi digital Internet of Things (IIoT) industri.

inspeksi VPC

Dalam arsitektur AWS multi-akun, VPC terpusat yang mengelola inspeksi lalu lintas jaringan antara VPC (dalam hal yang sama atau berbeda Wilayah AWS), internet, dan jaringan lokal. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan VPC masuk, keluar, dan inspeksi untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

Internet of Things (IoT)

Jaringan objek fisik yang terhubung dengan sensor atau prosesor tertanam yang berkomunikasi dengan perangkat dan sistem lain melalui internet atau melalui jaringan komunikasi lokal. Untuk informasi selengkapnya, lihat Apa itu IoT?

interpretabilitas

Karakteristik model pembelajaran mesin yang menggambarkan sejauh mana manusia dapat memahami bagaimana prediksi model bergantung pada inputnya. Untuk informasi lebih lanjut, lihat Interpretabilitas model pembelajaran mesin dengan. AWS

IoT

Lihat Internet of Things.

Perpustakaan informasi TI (ITIL)

Serangkaian praktik terbaik untuk memberikan layanan TI dan menyelaraskan layanan ini dengan persyaratan bisnis. ITIL menyediakan dasar untuk ITSM.

Manajemen layanan TI (ITSM)

Kegiatan yang terkait dengan merancang, menerapkan, mengelola, dan mendukung layanan TI untuk suatu organisasi. Untuk informasi tentang mengintegrasikan operasi cloud dengan alat ITSM, lihat panduan integrasi operasi.

ITIL

Lihat perpustakaan informasi TI.

ITSM

Lihat manajemen layanan TI.

L

kontrol akses berbasis label (LBAC)

Implementasi kontrol akses wajib (MAC) di mana pengguna dan data itu sendiri masing-masing secara eksplisit diberi nilai label keamanan. Persimpangan antara label keamanan pengguna dan label keamanan data menentukan baris dan kolom mana yang dapat dilihat oleh pengguna.

landing zone

Landing zone adalah AWS lingkungan multi-akun yang dirancang dengan baik yang dapat diskalakan dan aman. Ini adalah titik awal dari mana organisasi Anda dapat dengan cepat meluncurkan dan menyebarkan beban kerja dan aplikasi dengan percaya diri dalam lingkungan keamanan dan infrastruktur mereka. Untuk informasi selengkapnya tentang zona pendaratan, lihat Menyiapkan lingkungan multi-akun AWS yang aman dan dapat diskalakan.

model bahasa besar (LLM)

Model AI pembelajaran mendalam yang dilatih sebelumnya pada sejumlah besar data. LLM dapat melakukan beberapa tugas, seperti menjawab pertanyaan, meringkas dokumen, menerjemahkan teks ke bahasa lain, dan menyelesaikan kalimat. Untuk informasi lebih lanjut, lihat Apa itu LLM.

migrasi besar

Migrasi 300 atau lebih server.

LBAC

Lihat kontrol akses berbasis label.

hak istimewa paling sedikit

Praktik keamanan terbaik untuk memberikan izin minimum yang diperlukan untuk melakukan tugas. Untuk informasi selengkapnya, lihat Menerapkan izin hak istimewa terkecil dalam dokumentasi IAM.

angkat dan geser

Lihat 7 Rs.

sistem endian kecil

Sebuah sistem yang menyimpan byte paling tidak signifikan terlebih dahulu. Lihat juga endianness.

LLM

Lihat model bahasa besar.

lingkungan yang lebih rendah

Lihat lingkungan.

M

pembelajaran mesin (ML)

Jenis kecerdasan buatan yang menggunakan algoritma dan teknik untuk pengenalan pola dan pembelajaran. ML menganalisis dan belajar dari data yang direkam, seperti data Internet of Things (IoT), untuk menghasilkan model statistik berdasarkan pola. Untuk informasi selengkapnya, lihat Machine Learning.

cabang utama

Lihat cabang.

malware

Perangkat lunak yang dirancang untuk membahayakan keamanan atau privasi komputer. Malware dapat mengganggu sistem komputer, membocorkan informasi sensitif, atau mendapatkan akses yang tidak sah. Contoh malware termasuk virus, worm, ransomware, Trojan horse, spyware, dan keyloggers.

layanan terkelola

Layanan AWS yang AWS mengoperasikan lapisan infrastruktur, sistem operasi, dan platform, dan Anda mengakses titik akhir untuk menyimpan dan mengambil data. Amazon Simple Storage Service (Amazon S3) dan Amazon DynamoDB adalah contoh layanan terkelola. Ini juga dikenal sebagai layanan abstrak.

sistem eksekusi manufaktur (MES)

Sistem perangkat lunak untuk melacak, memantau, mendokumentasikan, dan mengendalikan proses produksi yang mengubah bahan baku menjadi produk jadi di lantai toko.

PETA

Lihat Program Percepatan Migrasi.

MCP

Lihat Protokol Konteks Model.

Protokol Konteks Model (MCP)

Protokol stateless untuk komunikasi agen -to- alat.

Server MCP

Layanan yang mengekspos satu atau lebih alat melalui Protokol Konteks Model.

mekanisme

Proses lengkap di mana Anda membuat alat, mendorong adopsi alat, dan kemudian memeriksa hasilnya untuk melakukan penyesuaian. Mekanisme adalah siklus yang memperkuat dan meningkatkan dirinya sendiri saat beroperasi. Untuk informasi selengkapnya, lihat Membangun mekanisme dalam AWS Well-Architected Kerangka Kerja.

akun anggota

Semua Akun AWS selain akun manajemen yang merupakan bagian dari organisasi di AWS Organizations. Akun dapat menjadi anggota dari hanya satu organisasi pada suatu waktu.

MES

Lihat sistem eksekusi manufaktur.

Transportasi Telemetri Antrian Pesan (MQTT)

Protokol komunikasi mesin-ke-mesin (M2M) yang ringan, berdasarkan pola publish/subscribe, untuk perangkat IoT yang dibatasi sumber daya.

layanan mikro

Layanan kecil dan independen yang berkomunikasi melalui API yang terdefinisi dengan baik dan biasanya dimiliki oleh tim kecil yang mandiri. Misalnya, sistem asuransi mungkin mencakup layanan mikro yang memetakan kemampuan bisnis, seperti penjualan atau pemasaran, atau subdomain, seperti pembelian, klaim, atau analitik. Manfaat layanan mikro termasuk kelincahan, penskalaan yang fleksibel, penyebaran yang mudah, kode yang dapat digunakan kembali, dan ketahanan. Untuk informasi selengkapnya, lihat Mengintegrasikan layanan mikro dengan menggunakan layanan tanpa AWS server.

arsitektur microservices

Pendekatan untuk membangun aplikasi dengan komponen independen yang menjalankan setiap proses aplikasi sebagai layanan mikro. Layanan mikro ini berkomunikasi melalui antarmuka yang terdefinisi dengan baik dengan menggunakan API ringan. Setiap layanan mikro dalam arsitektur ini dapat diperbarui, digunakan, dan diskalakan untuk memenuhi permintaan fungsi tertentu dari suatu aplikasi. Untuk informasi selengkapnya, lihat Menerapkan layanan mikro di AWS.

Program Percepatan Migrasi (MAP)

AWS Program yang menyediakan dukungan konsultasi, pelatihan, dan layanan untuk membantu organisasi membangun fondasi operasional yang kuat untuk pindah ke cloud, dan untuk membantu mengimbangi biaya awal migrasi. MAP mencakup metodologi migrasi untuk

mengeksekusi migrasi lama dengan cara metodis dan seperangkat alat untuk mengotomatisasi dan mempercepat skenario migrasi umum.

migrasi dalam skala

Proses memindahkan sebagian besar portofolio aplikasi ke cloud dalam gelombang, dengan lebih banyak aplikasi bergerak pada tingkat yang lebih cepat di setiap gelombang. Fase ini menggunakan praktik terbaik dan pelajaran yang dipetik dari fase sebelumnya untuk mengimplementasikan pabrik migrasi tim, alat, dan proses untuk merampingkan migrasi beban kerja melalui otomatisasi dan pengiriman tangkas. Ini adalah fase ketiga dari strategi AWS migrasi.

pabrik migrasi

Cross-functional tim yang merampingkan migrasi beban kerja melalui pendekatan otomatis dan gesit. Tim pabrik migrasi biasanya mencakup operasi, analis dan pemilik bisnis, insinyur migrasi, pengembang, dan DevOps profesional yang bekerja di sprint. Antara 20 dan 50 persen portofolio aplikasi perusahaan terdiri dari pola berulang yang dapat dioptimalkan dengan pendekatan pabrik. Untuk informasi selengkapnya, lihat diskusi tentang pabrik migrasi dan panduan Pabrik Migrasi Cloud di kumpulan konten ini.

metadata migrasi

Informasi tentang aplikasi dan server yang diperlukan untuk menyelesaikan migrasi. Setiap pola migrasi memerlukan satu set metadata migrasi yang berbeda. Contoh metadata migrasi termasuk subnet target, grup keamanan, dan akun. AWS

pola migrasi

Tugas migrasi berulang yang merinci strategi migrasi, tujuan migrasi, dan aplikasi atau layanan migrasi yang digunakan. Contoh: Rehost migrasi ke Amazon EC2 AWS dengan Layanan Migrasi Aplikasi.

Penilaian Portofolio Migrasi (MPA)

Alat online yang menyediakan informasi untuk memvalidasi kasus bisnis untuk bermigrasi ke. AWS Cloud MPA menyediakan penilaian portofolio terperinci (ukuran kanan server, harga, perbandingan TCO, analisis biaya migrasi) serta perencanaan migrasi (analisis data aplikasi dan pengumpulan data, pengelompokan aplikasi, prioritas migrasi, dan perencanaan gelombang). Alat MPA (memerlukan login) tersedia gratis untuk semua AWS konsultan dan konsultan APN Partner.

Penilaian Kesiapan Migrasi (MRA)

Proses mendapatkan wawasan tentang status kesiapan cloud organisasi, mengidentifikasi kekuatan dan kelemahan, dan membangun rencana aksi untuk menutup kesenjangan yang diidentifikasi, menggunakan CAF. AWS Untuk informasi selengkapnya, lihat panduan kesiapan migrasi. MRA adalah tahap pertama dari strategi AWS migrasi.

strategi migrasi

Pendekatan yang digunakan untuk memigrasikan beban kerja ke. AWS Cloud Untuk informasi lebih lanjut, lihat entri 7 Rs di glosarium ini dan lihat Memobilisasi organisasi Anda untuk mempercepat migrasi skala besar.

ML

Lihat pembelajaran mesin.

modernisasi

Mengubah aplikasi usang (warisan atau monolitik) dan infrastrukturnya menjadi sistem yang gesit, elastis, dan sangat tersedia di cloud untuk mengurangi biaya, mendapatkan efisiensi, dan memanfaatkan inovasi. Untuk informasi selengkapnya, lihat Strategi untuk memodernisasi aplikasi di. AWS Cloud

penilaian kesiapan modernisasi

Evaluasi yang membantu menentukan kesiapan modernisasi aplikasi organisasi; mengidentifikasi manfaat, risiko, dan dependensi; dan menentukan seberapa baik organisasi dapat mendukung keadaan masa depan aplikasi tersebut. Hasil penilaian adalah cetak biru arsitektur target, peta jalan yang merinci fase pengembangan dan tonggak untuk proses modernisasi, dan rencana aksi untuk mengatasi kesenjangan yang diidentifikasi. Untuk informasi lebih lanjut, lihat Mengevaluasi kesiapan modernisasi untuk aplikasi di. AWS Cloud

aplikasi monolitik (monolit)

Aplikasi yang berjalan sebagai layanan tunggal dengan proses yang digabungkan secara ketat. Aplikasi monolitik memiliki beberapa kelemahan. Jika satu fitur aplikasi mengalami lonjakan permintaan, seluruh arsitektur harus diskalakan. Menambahkan atau meningkatkan fitur aplikasi monolitik juga menjadi lebih kompleks ketika basis kode tumbuh. Untuk mengatasi masalah ini, Anda dapat menggunakan arsitektur microservices. Untuk informasi lebih lanjut, lihat Mengurai monolit menjadi layanan mikro.

MPA

Lihat Penilaian Portofolio Migrasi.

MQTT

Lihat Transportasi Telemetri Antrian Pesan.

klasifikasi multiclass

Sebuah proses yang membantu menghasilkan prediksi untuk beberapa kelas (memprediksi satu dari lebih dari dua hasil). Misalnya, model ML mungkin bertanya "Apakah produk ini buku, mobil, atau telepon?" atau "Kategori produk mana yang paling menarik bagi pelanggan ini?"

infrastruktur yang bisa berubah

Model yang memperbarui dan memodifikasi infrastruktur yang ada untuk beban kerja produksi. Untuk meningkatkan konsistensi, keandalan, dan prediktabilitas, AWS Well-Architected Framework merekomendasikan penggunaan infrastruktur yang tidak dapat diubah sebagai praktik terbaik.

O

OAC

Lihat kontrol akses asal.

OAI

Lihat identitas akses asal.

OCM

Lihat manajemen perubahan organisasi.

migrasi offline

Metode migrasi di mana beban kerja sumber diturunkan selama proses migrasi. Metode ini melibatkan waktu henti yang diperpanjang dan biasanya digunakan untuk beban kerja kecil dan tidak kritis.

OI

Lihat integrasi operasi.

OLA

Lihat perjanjian tingkat operasional.

migrasi online

Metode migrasi di mana beban kerja sumber disalin ke sistem target tanpa diambil offline. Aplikasi yang terhubung ke beban kerja dapat terus berfungsi selama migrasi. Metode ini melibatkan waktu henti nol hingga minimal dan biasanya digunakan untuk beban kerja produksi yang kritis.

OPC-UA

Lihat Komunikasi Proses Terbuka - Arsitektur Terpadu.

Komunikasi Proses Terbuka - Arsitektur Terpadu () OPC-UA

Protokol komunikasi mesin-ke-mesin (M2M) untuk otomasi industri. OPC-UA menyediakan standar interoperabilitas dengan enkripsi data, otentikasi, dan skema otorisasi.

perjanjian tingkat operasional (OLA)

Perjanjian yang menjelaskan apa yang dijanjikan kelompok TI fungsional untuk diberikan satu sama lain, untuk mendukung perjanjian tingkat layanan (SLA).

Tinjauan Kesiapan Operasional (ORR)

Daftar pertanyaan dan praktik terbaik terkait yang membantu Anda memahami, mengevaluasi, mencegah, atau mengurangi ruang lingkup insiden dan kemungkinan kegagalan. Untuk informasi selengkapnya, lihat Ulasan Kesiapan Operasional (ORR) dalam Kerangka Kerja. AWS Well-Architected

teknologi operasional (OT)

Sistem perangkat keras dan perangkat lunak yang bekerja dengan lingkungan fisik untuk mengendalikan operasi industri, peralatan, dan infrastruktur. Di bidang manufaktur, integrasi sistem OT dan teknologi informasi (TI) adalah fokus utama untuk transformasi Industri 4.0.

integrasi operasi (OI)

Proses modernisasi operasi di cloud, yang melibatkan perencanaan kesiapan, otomatisasi, dan integrasi. Untuk informasi selengkapnya, lihat panduan integrasi operasi.

jejak organisasi

Jejak yang dibuat oleh AWS CloudTrail itu mencatat semua peristiwa untuk semua Akun AWS dalam organisasi di AWS Organizations. Jejak ini dibuat di setiap Akun AWS bagian organisasi dan melacak aktivitas di setiap akun. Untuk informasi selengkapnya, lihat Membuat jejak untuk organisasi dalam CloudTrail dokumentasi.

manajemen perubahan organisasi (OCM)

Kerangka kerja untuk mengelola transformasi bisnis utama yang mengganggu dari perspektif orang, budaya, dan kepemimpinan. OCM membantu organisasi mempersiapkan, dan transisi ke, sistem dan strategi baru dengan mempercepat adopsi perubahan, mengatasi masalah transisi, dan mendorong perubahan budaya dan organisasi. Dalam strategi AWS migrasi, kerangka kerja ini disebut percepatan orang, karena kecepatan perubahan yang diperlukan dalam proyek adopsi cloud. Untuk informasi lebih lanjut, lihat panduan OCM.

kontrol akses asal (OAC)

Di CloudFront, opsi yang disempurnakan untuk membatasi akses untuk mengamankan konten Amazon Simple Storage Service (Amazon S3) Anda. OAC mendukung semua bucket S3 di semua Wilayah AWS, enkripsi sisi server dengan AWS KMS (SSE-KMS), dan dinamis `PUT` dan `DELETE` permintaan ke bucket S3.

identitas akses asal (OAI)

Di CloudFront, opsi untuk membatasi akses untuk mengamankan konten Amazon S3 Anda. Saat Anda menggunakan OAI, CloudFront buat prinsipal yang dapat diautentikasi oleh Amazon S3. Prinsipal yang diautentikasi dapat mengakses konten dalam bucket S3 hanya melalui distribusi tertentu. CloudFront Lihat juga OAC, yang menyediakan kontrol akses yang lebih terperinci dan ditingkatkan.

ORR

Lihat tinjauan kesiapan operasional.

OT

Lihat teknologi operasional.

keluar (jalan keluar) VPC

Dalam arsitektur AWS multi-akun, VPC yang menangani koneksi jaringan yang dimulai dari dalam aplikasi. Arsitektur Referensi AWS Keamanan merekomendasikan pengaturan akun Jaringan Anda dengan VPC masuk, keluar, dan inspeksi untuk melindungi antarmuka dua arah antara aplikasi Anda dan internet yang lebih luas.

P

batas izin

Kebijakan manajemen IAM yang dilampirkan pada prinsipal IAM untuk menetapkan izin maksimum yang dapat dimiliki pengguna atau peran. Untuk informasi selengkapnya, lihat Batas izin dalam dokumentasi IAM.

Informasi Identifikasi Pribadi (PII)

Informasi yang, jika dilihat secara langsung atau dipasangkan dengan data terkait lainnya, dapat digunakan untuk menyimpulkan identitas individu secara wajar. Contoh PII termasuk nama, alamat, dan informasi kontak.

PII

Lihat informasi yang dapat diidentifikasi secara pribadi.

buku pedoman

Serangkaian langkah yang telah ditentukan sebelumnya yang menangkap pekerjaan yang terkait dengan migrasi, seperti mengirimkan fungsi operasi inti di cloud. Buku pedoman dapat berupa skrip, runbook otomatis, atau ringkasan proses atau langkah-langkah yang diperlukan untuk mengoperasikan lingkungan modern Anda.

PLC

Lihat pengontrol logika yang dapat diprogram.

PLM

Lihat manajemen siklus hidup produk.

kebijakan

Objek yang dapat menentukan izin (lihat kebijakan berbasis identitas), menentukan kondisi akses (lihat kebijakanberbasis sumber daya), atau menentukan izin maksimum untuk semua akun dalam organisasi di (lihat kebijakan kontrol layanan). AWS Organizations

persistensi poliglot

Secara independen memilih teknologi penyimpanan data microservice berdasarkan pola akses data dan persyaratan lainnya. Jika layanan mikro Anda memiliki teknologi penyimpanan data yang sama, mereka dapat menghadapi tantangan implementasi atau mengalami kinerja yang buruk. Layanan mikro lebih mudah diimplementasikan dan mencapai kinerja dan skalabilitas yang lebih baik jika mereka menggunakan penyimpanan data yang paling sesuai dengan kebutuhan mereka.

penilaian portofolio

Proses menemukan, menganalisis, dan memprioritaskan portofolio aplikasi untuk merencanakan migrasi. Untuk informasi selengkapnya, lihat Mengevaluasi kesiapan migrasi.

predikat

Kondisi kueri yang mengembalikan `true` atau`false`, biasanya terletak di `WHERE` klausa.

predikat pushdown

Teknik pengoptimalan kueri database yang menyaring data dalam kueri sebelum transfer. Ini mengurangi jumlah data yang harus diambil dan diproses dari database relasional, dan meningkatkan kinerja kueri.

kontrol preventif

Kontrol keamanan yang dirancang untuk mencegah suatu peristiwa terjadi. Kontrol ini adalah garis pertahanan pertama untuk membantu mencegah akses tidak sah atau perubahan yang tidak diinginkan ke jaringan Anda. Untuk informasi selengkapnya, lihat Kontrol pencegahan dalam Menerapkan kontrol keamanan pada. AWS

principal

Entitas AWS yang dapat melakukan tindakan dan mengakses sumber daya. Entitas ini biasanya merupakan pengguna root untuk Akun AWS, peran IAM, atau pengguna. Untuk informasi selengkapnya, lihat Prinsip dalam istilah dan konsep Peran dalam dokumentasi IAM.

privasi berdasarkan desain

Pendekatan rekayasa sistem yang memperhitungkan privasi melalui seluruh proses pengembangan.

zona host pribadi

Container yang menyimpan informasi tentang bagaimana Anda ingin Amazon Route 53 merespons kueri DNS untuk domain dan subdomainnya dalam satu atau beberapa VPC. Untuk informasi selengkapnya, lihat Bekerja dengan zona yang dihosting pribadi di dokumentasi Route 53.

kontrol proaktif

Kontrol keamanan yang dirancang untuk mencegah penyebaran sumber daya yang tidak sesuai. Kontrol ini memindai sumber daya sebelum disediakan. Jika sumber daya tidak sesuai dengan kontrol, maka itu tidak disediakan. Untuk informasi selengkapnya, lihat panduan referensi Kontrol

dalam AWS Control Tower dokumentasi dan lihat Kontrol proaktif dalam Menerapkan kontrol keamanan pada AWS.

manajemen siklus hidup produk (PLM)

Manajemen data dan proses untuk suatu produk di seluruh siklus hidupnya, mulai dari desain, pengembangan, dan peluncuran, melalui pertumbuhan dan kematangan, hingga penurunan dan penghapusan.

lingkungan produksi

Lihat lingkungan.

pengontrol logika yang dapat diprogram (PLC)

Di bidang manufaktur, komputer yang sangat andal dan mudah beradaptasi yang memantau mesin dan mengotomatiskan proses manufaktur.

rantai cepat

Menggunakan output dari satu prompt LLM sebagai input untuk prompt berikutnya untuk menghasilkan respons yang lebih baik. Teknik ini digunakan untuk memecah tugas yang kompleks menjadi subtugas, atau untuk secara iteratif memperbaiki atau memperluas respons awal. Ini membantu meningkatkan akurasi dan relevansi respons model dan memungkinkan hasil yang lebih terperinci dan dipersonalisasi.

pseudonimisasi

Proses penggantian pengenal pribadi dalam kumpulan data dengan nilai placeholder. Pseudonimisasi dapat membantu melindungi privasi pribadi. Data pseudonim masih dianggap sebagai data pribadi.

publish/subscribe (pub/sub)

Pola yang memungkinkan komunikasi asinkron antara layanan mikro untuk meningkatkan skalabilitas dan daya tanggap. Misalnya, dalam MES berbasis layanan mikro, layanan mikro dapat mempublikasikan pesan peristiwa ke saluran yang dapat berlangganan layanan mikro lainnya. Sistem dapat menambahkan layanan mikro baru tanpa mengubah layanan penerbitan.

Q

rencana kueri

Serangkaian langkah, seperti instruksi, yang digunakan untuk mengakses data dalam sistem database relasional SQL.

regresi rencana kueri

Ketika pengoptimal layanan database memilih rencana yang kurang optimal daripada sebelum perubahan yang diberikan ke lingkungan database. Hal ini dapat disebabkan oleh perubahan statistik, kendala, pengaturan lingkungan, pengikatan parameter kueri, dan pembaruan ke mesin database.

R

Matriks RACI

Lihat bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI).

LAP

Lihat Retrieval Augmented Generation.

ransomware

Perangkat lunak berbahaya yang dirancang untuk memblokir akses ke sistem komputer atau data sampai pembayaran dilakukan.

Matriks RASCI

Lihat bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI).

RCAC

Lihat kontrol akses baris dan kolom.

replika baca

Salinan database yang digunakan untuk tujuan read-only. Anda dapat merutekan kueri ke replika baca untuk mengurangi beban pada database utama Anda.

arsitek ulang

Lihat 7 Rs.

tujuan titik pemulihan (RPO)

Jumlah waktu maksimum yang dapat diterima sejak titik pemulihan data terakhir. Ini menentukan apa yang dianggap sebagai kehilangan data yang dapat diterima antara titik pemulihan terakhir dan gangguan layanan.

tujuan waktu pemulihan (RTO)

Penundaan maksimum yang dapat diterima antara gangguan layanan dan pemulihan layanan.

refactor

Lihat 7 Rs.

Region

Kumpulan AWS sumber daya di wilayah geografis. Masing-masing Wilayah AWS terisolasi dan independen dari yang lain untuk memberikan toleransi kesalahan, stabilitas, dan ketahanan. Untuk informasi selengkapnya, lihat Menentukan Wilayah AWS akun yang dapat digunakan.

regresi

Teknik ML yang memprediksi nilai numerik. Misalnya, untuk memecahkan masalah “Berapa harga rumah ini akan dijual?” Model ML dapat menggunakan model regresi linier untuk memprediksi harga jual rumah berdasarkan fakta yang diketahui tentang rumah (misalnya, luas persegi).

rehost

Lihat 7 Rs.

melepaskan

Dalam proses penyebaran, tindakan mempromosikan perubahan pada lingkungan produksi.

memindahkan

Lihat 7 Rs.

memplatform ulang

Lihat 7 Rs.

pembelian kembali

Lihat 7 Rs.

ketahanan

Kemampuan aplikasi untuk melawan atau pulih dari gangguan. Ketersediaan tinggi dan pemulihan bencana adalah pertimbangan umum ketika merencanakan ketahanan di. AWS Cloud Untuk informasi lebih lanjut, lihat AWS Cloud Ketahanan.

kebijakan berbasis sumber daya

Kebijakan yang dilampirkan ke sumber daya, seperti bucket Amazon S3, titik akhir, atau kunci enkripsi. Jenis kebijakan ini menentukan prinsipal mana yang diizinkan mengakses, tindakan yang didukung, dan kondisi lain yang harus dipenuhi.

matriks yang bertanggung jawab, akuntabel, dikonsultasikan, diinformasikan (RACI)

Matriks yang mendefinisikan peran dan tanggung jawab untuk semua pihak yang terlibat dalam kegiatan migrasi dan operasi cloud. Nama matriks berasal dari jenis tanggung jawab yang didefinisikan dalam matriks: bertanggung jawab (R), akuntabel (A), dikonsultasikan (C), dan diinformasikan (I). Jenis dukungan (S) adalah opsional. Jika Anda menyertakan dukungan, matriks disebut matriks RASCI, dan jika Anda mengecualikannya, itu disebut matriks RACI.

kontrol responsif

Kontrol keamanan yang dirancang untuk mendorong remediasi efek samping atau penyimpangan dari garis dasar keamanan Anda. Untuk informasi selengkapnya, lihat Kontrol responsif dalam Menerapkan kontrol keamanan pada AWS.

melestarikan

Lihat 7 Rs.

pensiun

Lihat 7 Rs.

Retrieval Augmented Generation (RAG)

Teknologi AI generatif di mana LLM mereferensikan sumber data otoritatif yang berada di luar sumber data pelatihannya sebelum menghasilkan respons. Misalnya, model RAG mungkin melakukan pencarian semantik dari basis pengetahuan organisasi atau data kustom. Untuk informasi lebih lanjut, lihat Apa itu RAG.

rotasi

Proses memperbarui rahasia secara berkala untuk membuatnya lebih sulit bagi penyerang untuk mengakses kredensil.

kontrol akses baris dan kolom (RCAC)

Penggunaan ekspresi SQL dasar dan fleksibel yang telah menetapkan aturan akses. RCAC terdiri dari izin baris dan topeng kolom.

RPO

Lihat tujuan titik pemulihan.

RTO

Lihat tujuan waktu pemulihan.

buku runbook

Satu set prosedur manual atau otomatis yang diperlukan untuk melakukan tugas tertentu. Ini biasanya dibangun untuk merampingkan operasi berulang atau prosedur dengan tingkat kesalahan yang tinggi.

D

SAML 2.0

Standar terbuka yang digunakan oleh banyak penyedia identitas (IdPs). Fitur ini memungkinkan sistem masuk tunggal gabungan (SSO), sehingga pengguna dapat masuk ke Konsol Manajemen AWS atau memanggil operasi AWS API tanpa Anda harus membuat pengguna di IAM untuk semua orang di organisasi Anda. Untuk informasi lebih lanjut tentang federasi berbasis SAMP 2.0, lihat Tentang federasi berbasis SAMP 2.0 dalam dokumentasi IAM.

SCADA

Lihat kontrol pengawasan dan akuisisi data.

SCP

Lihat kebijakan kontrol layanan.

Rahasia

Dalam AWS Secrets Manager, informasi rahasia atau terbatas, seperti kata sandi atau kredensil pengguna, yang Anda simpan dalam bentuk terenkripsi. Ini terdiri dari nilai rahasia dan metadata-nya. Nilai rahasia dapat berupa biner, string tunggal, atau beberapa string. Untuk informasi selengkapnya, lihat Apa yang ada di rahasia Secrets Manager? dalam dokumentasi Secrets Manager.

keamanan dengan desain

Pendekatan rekayasa sistem yang memperhitungkan keamanan melalui seluruh proses pengembangan.

kontrol keamanan

Pagar pembatas teknis atau administratif yang mencegah, mendeteksi, atau mengurangi kemampuan pelaku ancaman untuk mengeksploitasi kerentanan keamanan. Ada empat jenis kontrol keamanan utama: preventif, detektif, responsif, dan proaktif.

pengerasan keamanan

Proses mengurangi permukaan serangan untuk membuatnya lebih tahan terhadap serangan. Ini dapat mencakup tindakan seperti menghapus sumber daya yang tidak lagi diperlukan, menerapkan praktik keamanan terbaik untuk memberikan hak istimewa paling sedikit, atau menonaktifkan fitur yang tidak perlu dalam file konfigurasi.

sistem informasi keamanan dan manajemen acara (SIEM)

Alat dan layanan yang menggabungkan sistem manajemen informasi keamanan (SIM) dan manajemen acara keamanan (SEM). Sistem SIEM mengumpulkan, memantau, dan menganalisis data dari server, jaringan, perangkat, dan sumber lain untuk mendeteksi ancaman dan pelanggaran keamanan, dan untuk menghasilkan peringatan.

otomatisasi respons keamanan

Tindakan yang telah ditentukan dan diprogram yang dirancang untuk secara otomatis merespons atau memulihkan peristiwa keamanan. Otomatisasi ini berfungsi sebagai kontrol keamanan detektif atau responsif yang membantu Anda menerapkan praktik terbaik AWS keamanan. Contoh tindakan respons otomatis termasuk memodifikasi grup keamanan VPC, menambal instans Amazon EC2, atau memutar kredensil.

enkripsi sisi server

Enkripsi data di tujuannya, oleh Layanan AWS yang menerimanya.

kebijakan kontrol layanan (SCP)

Kebijakan yang menyediakan kontrol terpusat atas izin untuk semua akun di organisasi. AWS Organizations SCP menentukan pagar pembatas atau menetapkan batasan pada tindakan yang dapat didelegasikan oleh administrator kepada pengguna atau peran. Anda dapat menggunakan SCP sebagai daftar izin atau daftar penolakan, untuk menentukan layanan atau tindakan mana

yang diizinkan atau dilarang. Untuk informasi selengkapnya, lihat Kebijakan kontrol layanan dalam AWS Organizations dokumentasi.

titik akhir layanan

URL titik masuk untuk file Layanan AWS. Anda dapat menggunakan endpoint untuk terhubung secara terprogram ke layanan target. Untuk informasi selengkapnya, lihat Layanan AWS titik akhir di Referensi Umum AWS.

perjanjian tingkat layanan (SLA)

Perjanjian yang menjelaskan apa yang dijanjikan oleh tim TI untuk diberikan kepada pelanggan mereka, seperti uptime dan kinerja layanan.

indikator tingkat layanan (SLI)

Pengukuran aspek kinerja layanan, seperti tingkat kesalahan, ketersediaan, atau throughputnya.

tujuan tingkat layanan (SLO)

Metrik target yang mewakili kesehatan layanan, yang diukur dengan indikator tingkat layanan.

model tanggung jawab bersama

Model yang menjelaskan tanggung jawab yang Anda bagikan AWS untuk keamanan dan kepatuhan cloud. AWS bertanggung jawab atas keamanan cloud, sedangkan Anda bertanggung jawab atas keamanan di cloud. Untuk informasi selengkapnya, lihat Model tanggung jawab bersama.

Bayangan AI

Aplikasi AI yang tidak sah dibuat atau digunakan di luar saluran yang diatur dalam suatu organisasi.

SIEM

Lihat informasi keamanan dan sistem manajemen acara.

titik kegagalan tunggal (SPOF)

Kegagalan dalam satu komponen penting dari aplikasi yang dapat mengganggu sistem.

SLA

Lihat perjanjian tingkat layanan.

SLI

Lihat indikator tingkat layanan.

SLO

Lihat tujuan tingkat layanan.

model split-and-seed

Pola untuk menskalakan dan mempercepat proyek modernisasi. Ketika fitur baru dan rilis produk didefinisikan, tim inti berpisah untuk membuat tim produk baru. Ini membantu meningkatkan kemampuan dan layanan organisasi Anda, meningkatkan produktivitas pengembang, dan mendukung inovasi yang cepat. Untuk informasi lebih lanjut, lihat Pendekatan bertahap untuk memodernisasi aplikasi di. AWS Cloud

SPOF

Lihat satu titik kegagalan.

skema bintang

Struktur organisasi database yang menggunakan satu tabel fakta besar untuk menyimpan data transaksional atau terukur dan menggunakan satu atau lebih tabel dimensi yang lebih kecil untuk menyimpan atribut data. Struktur ini dirancang untuk digunakan dalam gudang data atau untuk tujuan intelijen bisnis.

pola ara pencekik

Pendekatan untuk memodernisasi sistem monolitik dengan menulis ulang secara bertahap dan mengganti fungsionalitas sistem sampai sistem warisan dapat dinonaktifkan. Pola ini menggunakan analogi pohon ara yang tumbuh menjadi pohon yang sudah mapan dan akhirnya mengatasi dan menggantikan inangnya. Pola ini diperkenalkan oleh Martin Fowler sebagai cara untuk mengelola risiko saat menulis ulang sistem monolitik. Untuk contoh cara menerapkan pola ini, lihat Memodernisasi layanan web ASP.NET Microsoft (ASMX) lama secara bertahap menggunakan container dan Amazon API Gateway.

subnet

Rentang alamat IP dalam VPC Anda. Subnet harus berada di Availability Zone tunggal.

kontrol pengawasan dan akuisisi data (SCADA)

Di bidang manufaktur, sistem yang menggunakan perangkat keras dan perangkat lunak untuk memantau aset fisik dan operasi produksi.

enkripsi simetris

Algoritma enkripsi yang menggunakan kunci yang sama untuk mengenkripsi dan mendekripsi data.

pengujian sintetis

Menguji sistem dengan cara yang mensimulasikan interaksi pengguna untuk mendeteksi potensi masalah atau untuk memantau kinerja. Anda dapat menggunakan Amazon CloudWatch Synthetics untuk membuat tes ini.

sistem prompt

Teknik untuk memberikan konteks, instruksi, atau pedoman ke LLM untuk mengarahkan perilakunya. Permintaan sistem membantu mengatur konteks dan menetapkan aturan untuk interaksi dengan pengguna.

T

tag

Key-value pasangan yang bertindak sebagai metadata untuk mengatur sumber daya Anda AWS . Tag membantu Anda mengelola, mengidentifikasi, mengatur, dan memfilter sumber daya. Untuk informasi selengkapnya, lihat Menandai sumber daya AWS.

variabel target

Nilai yang Anda coba prediksi dalam ML yang diawasi. Ini juga disebut sebagai variabel hasil. Misalnya, dalam pengaturan manufaktur, variabel target bisa menjadi cacat produk.

daftar tugas

Alat yang digunakan untuk melacak kemajuan melalui runbook. Daftar tugas berisi ikhtisar runbook dan daftar tugas umum yang harus diselesaikan. Untuk setiap tugas umum, itu termasuk perkiraan jumlah waktu yang dibutuhkan, pemilik, dan kemajuan.

lingkungan uji

Lihat lingkungan.

pelatihan

Untuk menyediakan data bagi model ML Anda untuk dipelajari. Data pelatihan harus berisi jawaban yang benar. Algoritma pembelajaran menemukan pola dalam data pelatihan yang memetakan atribut data input ke target (jawaban yang ingin Anda prediksi). Ini menghasilkan model ML yang menangkap pola-pola ini. Anda kemudian dapat menggunakan model ML untuk membuat prediksi pada data baru yang Anda tidak tahu targetnya.

alat

Fungsi atau API yang dapat dipanggil agen untuk melakukan operasi di sistem eksternal.

gerbang transit

Hub transit jaringan yang dapat Anda gunakan untuk menghubungkan VPC dan jaringan lokal Anda. Untuk informasi selengkapnya, lihat Apa itu gateway transit dalam AWS Transit Gateway dokumentasi.

alur kerja berbasis batang

Pendekatan di mana pengembang membangun dan menguji fitur secara lokal di cabang fitur dan kemudian menggabungkan perubahan tersebut ke cabang utama. Cabang utama kemudian dibangun untuk pengembangan, praproduksi, dan lingkungan produksi, secara berurutan.

akses tepercaya

Memberikan izin ke layanan yang Anda tentukan untuk melakukan tugas di organisasi Anda di dalam AWS Organizations dan di akunnya atas nama Anda. Layanan tepercaya menciptakan peran terkait layanan di setiap akun, ketika peran itu diperlukan, untuk melakukan tugas manajemen untuk Anda. Untuk informasi selengkapnya, lihat Menggunakan AWS Organizations dengan AWS layanan lain dalam AWS Organizations dokumentasi.

penyetelan

Untuk mengubah aspek proses pelatihan Anda untuk meningkatkan akurasi model ML. Misalnya, Anda dapat melatih model ML dengan membuat set pelabelan, menambahkan label, dan kemudian mengulangi langkah-langkah ini beberapa kali di bawah pengaturan yang berbeda untuk mengoptimalkan model.

tim dua pizza

Sebuah DevOps tim kecil yang bisa Anda beri makan dengan dua pizza. Ukuran tim dua pizza memastikan peluang terbaik untuk berkolaborasi dalam pengembangan perangkat lunak.

U

waswas

Sebuah konsep yang mengacu pada informasi yang tidak tepat, tidak lengkap, atau tidak diketahui yang dapat merusak keandalan model ML prediktif. Ada dua jenis ketidakpastian:

ketidakpastian epistemik disebabkan oleh data yang terbatas dan tidak lengkap, sedangkan ketidakpastian aleatorik disebabkan oleh kebisingan dan keacakan yang melekat dalam data.

tugas yang tidak terdiferensiasi

Juga dikenal sebagai angkat berat, pekerjaan yang diperlukan untuk membuat dan mengoperasikan aplikasi tetapi itu tidak memberikan nilai langsung kepada pengguna akhir atau memberikan keunggulan kompetitif. Contoh tugas yang tidak terdiferensiasi termasuk pengadaan, pemeliharaan, dan perencanaan kapasitas.

lingkungan atas

Lihat lingkungan.

V

menyedot debu

Operasi pemeliharaan database yang melibatkan pembersihan setelah pembaruan tambahan untuk merebut kembali penyimpanan dan meningkatkan kinerja.

kendali versi

Proses dan alat yang melacak perubahan, seperti perubahan kode sumber dalam repositori.

Peering VPC

Koneksi antara dua VPC yang memungkinkan Anda merutekan lalu lintas dengan menggunakan alamat IP pribadi. Untuk informasi selengkapnya, lihat Apa itu peering VPC di dokumentasi VPC Amazon.

kerentanan

Kelemahan perangkat lunak atau perangkat keras yang membahayakan keamanan sistem.

W

cache hangat

Cache buffer yang berisi data terkini dan relevan yang sering diakses. Instance database dapat membaca dari cache buffer, yang lebih cepat daripada membaca dari memori utama atau disk.

data hangat

Data yang jarang diakses. Saat menanyakan jenis data ini, kueri yang cukup lambat biasanya dapat diterima.

fungsi jendela

Fungsi SQL yang melakukan perhitungan pada sekelompok baris yang berhubungan dengan catatan saat ini. Fungsi jendela berguna untuk memproses tugas, seperti menghitung rata-rata bergerak atau mengakses nilai baris berdasarkan posisi relatif dari baris saat ini.

beban kerja

Kumpulan sumber daya dan kode yang memberikan nilai bisnis, seperti aplikasi yang dihadapi pelanggan atau proses backend.

aliran kerja

Grup fungsional dalam proyek migrasi yang bertanggung jawab atas serangkaian tugas tertentu. Setiap alur kerja independen tetapi mendukung alur kerja lain dalam proyek. Misalnya, alur kerja portofolio bertanggung jawab untuk memprioritaskan aplikasi, perencanaan gelombang, dan mengumpulkan metadata migrasi. Alur kerja portofolio mengirimkan aset ini ke alur kerja migrasi, yang kemudian memigrasikan server dan aplikasi.

CACING

Lihat menulis sekali, baca banyak.

WQF

Lihat AWS Kerangka Kualifikasi Beban Kerja.

tulis sekali, baca banyak (WORM)

Model penyimpanan yang menulis data satu kali dan mencegah data dihapus atau dimodifikasi. Pengguna yang berwenang dapat membaca data sebanyak yang diperlukan, tetapi mereka tidak dapat mengubahnya. Infrastruktur penyimpanan data ini dianggap tidak dapat diubah.

Z

eksploitasi zero-day

Serangan, biasanya malware, yang memanfaatkan kerentanan zero-day.

kerentanan zero-day

Cacat atau kerentanan yang tak tanggung-tanggung dalam sistem produksi. Aktor ancaman dapat menggunakan jenis kerentanan ini untuk menyerang sistem. Pengembang sering menyadari kerentanan sebagai akibat dari serangan tersebut.

bisikan zero-shot

Memberikan LLM dengan instruksi untuk melakukan tugas tetapi tidak ada contoh (tembakan) yang dapat membantu membimbingnya. LLM harus menggunakan pengetahuan pra-terlatih untuk menangani tugas. Efektivitas bidikan nol tergantung pada kompleksitas tugas dan kualitas prompt. Lihat juga beberapa bidikan yang diminta.

aplikasi zombie

Aplikasi yang memiliki CPU rata-rata dan penggunaan memori di bawah 5 persen. Dalam proyek migrasi, adalah umum untuk menghentikan aplikasi ini.

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