Strategi Protokol Konteks Model pada AWS

AWS Bimbingan Preskriptif

AWS Bimbingan Preskriptif: Strategi Protokol Konteks Model pada AWS

Table of Contents

Strategi Protokol Konteks Model pada AWS

Amazon Web Services (kontributor)

Maret 2026 (sejarah dokumen)

Panduan ini dapat membantu Anda mengembangkan dan menerapkan strategi Model Context Protocol (MCP) di seluruh organisasi Anda untuk mendukung perjalanan AI agen Anda. Ketika agen dan model bahasa menjadi semakin sentral untuk operasi bisnis, menetapkan strategi MCP sangat penting untuk solusi agen yang sukses.

Panduan ini mengeksplorasi tiga pilar dasar untuk membangun strategi MCP: desain alat MCP, hosting server MCP, dan tata kelola MCP. Dengan menangani komponen yang saling berhubungan ini, organisasi dapat membuat sistem yang terukur, aman, dan efektif untuk mengelola konteks model di seluruh implementasi AI mereka. Panduan ini memberikan wawasan yang dapat ditindaklanjuti dan panduan strategis untuk organisasi di setiap tahap perjalanan AI organisasi, mulai dari eksperimen awal hingga penerapan produksi skala penuh. Ini membantu mereka mengembangkan solusi MCP yang disesuaikan yang selaras dengan kebutuhan dan tujuan spesifik mereka.

Praktik terbaik ini berasal dari implementasi dunia nyata dari organisasi yang menerapkan MCP pada skala perusahaan, analisis standar spesifikasi MCP saat ini, dan pelajaran yang dipetik dari aplikasi Large Language Model (LLM) kustom dalam produksi.

Sistem AI menggunakan semakin canggih dan kuat LLMs dalam berbagai kasus penggunaan. LLMs unggul dalam memahami bahasa alami, menghasilkan respons seperti manusia, dan penalaran atas informasi yang kompleks. Namun, untuk mengubah LLMs dari antarmuka percakapan menjadi sistem yang dapat secara mandiri menyelesaikan tugas-tugas kompleks, organisasi mengadopsi arsitektur AI agen, sistem AI yang dapat memahami lingkungan mereka, alasan tentang tujuan, membuat keputusan otonom, mengatur berbagai langkah, dan mengambil tindakan untuk mencapai tujuan atas nama pengguna. Pendekatan agen ini membantu organisasi membangun sistem AI yang dapat memahami maksud pengguna melalui bahasa alami, berkoordinasi secara mandiri di berbagai sumber dan alat data, dan memberikan pengalaman yang dipersonalisasi pada skala yang tidak mungkin dilakukan dengan pola permintaan-respons tradisional. Untuk membuat agen ini lebih mampu, organisasi perlu menyediakan akses ke alat dan data yang ada untuk memperkaya pemahaman kontekstual agen dan memungkinkannya bertindak atas nama pengguna.

MCP menyediakan protokol standar untuk integrasi alat AI, memungkinkan komunikasi yang konsisten antara agen dan sumber daya eksternal. Sementara MCP sendiri mendefinisikan standar

komunikasi, menerapkannya secara efektif memerlukan pertimbangan yang cermat terhadap pola arsitektur, model keamanan, praktik operasional, dan strategi pengoptimalan kinerja untuk mencapai solusi yang dapat diskalakan, aman, dan dapat dipelihara.

Panduan ini mensintesis pelajaran yang dipetik dari penerapan MCP perusahaan, memberikan rekomendasi yang dapat ditindaklanjuti yang selaras dengan Kerangka Well-Architected.AWS Ini mencakup strategi untuk desain alat MCP, hosting server MCP, dan tata kelola MCP, yang penting dalam membangun solusi MCP Anda sendiri. Rekomendasi dalam panduan ini memetakan lima pilar Kerangka Kerja AWS Well-Architected berikut:

- Keamanan — Isolasi token, kredensi tercakup, otorisasi terpisah read/write
- Keunggulan Operasional - Metrik akurasi pemilihan alat, kumpulan data emas untuk pengujian regresi
- Keandalan - Pembatasan laju per pengguna dan per alat, penumpahan beban
- Efisiensi kinerja - Alat dengan cakupan alur kerja, penyaringan alat, pencarian semantik untuk mengurangi penggunaan jendela konteks
- Pengoptimalan biaya — Server MCP yang dapat digunakan kembali di seluruh tim, mengurangi biaya token per permintaan melalui pemfilteran alat

Audiens yang dituju

Panduan ini ditujukan untuk arsitek, pengembang, dan pemimpin teknologi yang menerapkan solusi AI agen di organisasi mereka. Untuk memahami konsep dalam panduan ini, Anda harus memahami cara LLMs kerja dan memiliki pengetahuan dasar tentang MCP, alat, dan teknik yang cepat.

Tujuan

Membangun sistem AI Agentic yang siap produksi berarti menyelesaikan tata kelola, pengoptimalan, dan keamanan bersama untuk mendukung kebijakan organisasi Anda. Di bawah ini menjelaskan bagaimana panduan ini membahas tujuan-tujuan ini:

- Tata Kelola — Tanpa tata kelola terpusat, Anda tidak dapat menjawab pertanyaan audit tentang beban kerja AI Anda, termasuk agen mana yang mengakses data mana, dengan izin apa, dan kapan. Anda juga tidak dapat memaksakan pembuatan versi. Bagian strategi hosting MCP dari panduan ini menjelaskan bagaimana pengguna dapat menjalankan server MCP lokal yang sudah ketinggalan zaman dengan kerentanan yang diketahui karena kurangnya penegakan sistematis.

Untuk industri yang diatur, tata kelola sangat penting. Auditor ingin melihat penegakan kebijakan dan pelacakan penggunaan alat di semua agen dari satu panel. Tata kelola MCP menyediakan itu.

Dengan mengikuti rekomendasi dalam panduan ini, Anda dapat meningkatkan akurasi tugas sebesar 28-32% dalam tolok ukur peer-review. Untuk informasi selengkapnya, lihat MARCO: Orkestrasi Obrolan Real-Time Multi-Agen (situs web Antologi ACL). Tata kelola bukan hanya tentang kepatuhan; itu juga meningkatkan kinerja sistem AI agen Anda.

- Optimasi — Tim Anda mungkin membangun integrasi yang sama lebih dari sekali. Misalnya, ketika lima tim yang berbeda menulis skrip kueri database mereka sendiri untuk aplikasi AI mereka untuk berkomunikasi dengan database mereka, itu lima kali biaya pengembangan dan lima set daftar bug yang harus dipertahankan. MCP memungkinkan Anda membangunnya sekali dan membagikannya ke seluruh komunitas teknik. Senyawa tabungan saat jumlah agen Anda bertambah.

  Ada juga masalah biaya per permintaan yang sebagian besar tim tidak menyadarinya pada awalnya. Setiap definisi alat menggunakan token jendela konteks. Pada 20 alat, Anda menghabiskan 5.000-10.000 token per permintaan untuk deskripsi saja, di samping pertanyaan pengguna. Ini meningkatkan biaya inferensi latensi dan LLM dan menurunkan akurasi karena model berjuang untuk memilih alat yang tepat dari daftar alat yang tersedia.

  Agen yang menggunakan pembungkus alat terstruktur kira-kira tiga kali lebih akurat pada tugas database daripada agen yang mengakses APIs secara langsung (untuk informasi selengkapnya, lihat Middleware untuk LLMs: Alat Instrumental untuk Agen Bahasa di Lingkungan Kompleks). Bagaimana Anda mendesain dan menyajikan alat untuk model AI adalah penting. Panduan ini merekomendasikan untuk memberikan skema yang jelas kepada alat, melingkupinya ke alur kerja aktual alih-alih titik akhir mentah, dan membatasi informasi di jendela konteks. Bagian strategi desain alat MCP dari panduan ini menyelami aspek-aspek ini.
- Keamanan dan kepatuhan - Bayangkan sistem AI agen yang berhalusinasi langkah pembersihan dan mencoba menghapus basis data produksi. Jika agen mewarisi kredensi admin lengkap pengguna, penghapusan mungkin akan dilakukan. Dengan isolasi token dan kredenal cakupan yang hanya memberikan akses baca dan buat, gagal dengan aman.

  Alur kerja yang diatur mempertajam ini lebih lanjut. Panduan ini memberikan contoh (jaringan pipa perawatan kesehatan yang memerlukan validasi HIPAA dan anonimisasi informasi yang dapat diidentifikasi secara pribadi sebelum memproses data pasien). Menyematkan logika tersebut dalam alat MCP berarti kepatuhan terjadi secara deterministik setiap saat.

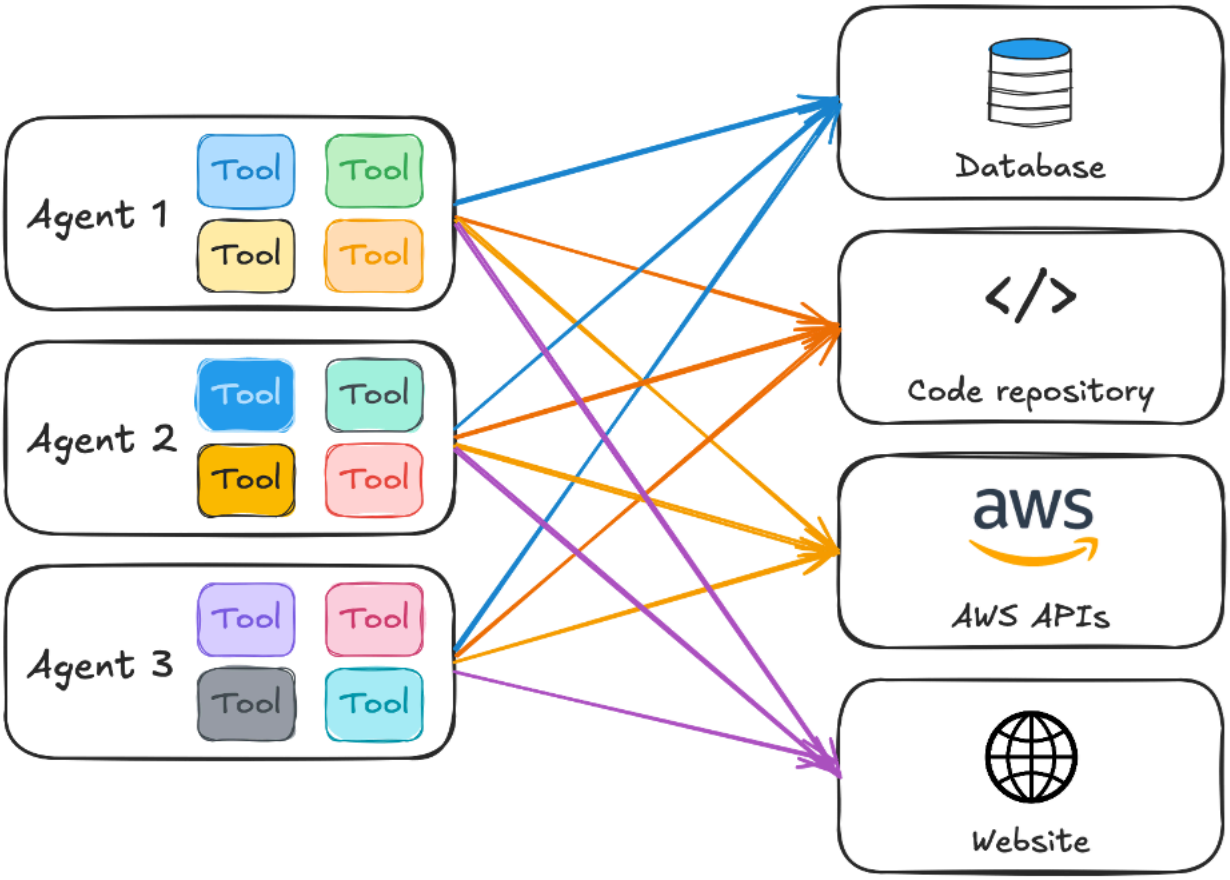
Apa itu MCP?

LLMs bekerja dengan memprediksi jawaban atas prompt berdasarkan data pelatihan mereka. Ini berarti bahwa LLM hanya dapat memberikan jawaban tentang data dan peristiwa yang telah dilihatnya. Metode seperti Retrieval Augmented Generation (RAG) dan basis pengetahuan memungkinkan Anda untuk memasukkan data kontekstual. Namun, jika Anda bertanya kepada LLM apa ramalan cuaca besok atau berapa banyak pelanggan di database Anda, kemungkinan akan berhalusinasi atau tidak dapat memberikan jawaban karena ini berada di luar pengetahuan pra-terlatih LLM. Untuk dapat menjawab pertanyaan semacam ini, agen membutuhkan akses ke kemampuan eksternal, data, dan APIs di luar konteks asli LLM.

Memahami alat

Kami dapat memberikan akses LLM ke sistem dan konteks tambahan melalui alat.Alat adalah fungsi yang diberikan kepada LLM untuk mencapai tujuan yang jelas. Alat dapat memanggil API, menanyakan database, melakukan operasi kalkulator, mengoperasikan kotak pasir kode, melakukan pencarian web, dan bahkan memanggil sistem AI lain atau. agent-as-a-tool Setiap alat harus menyertakan deskripsi yang memberi tahu LLM apa yang dilakukan alat, kapan menggunakannya, dan parameter apa yang diterimanya. Hal ini memungkinkan LLM untuk membuat keputusan bernuansa tentang alat atau kombinasi alat mana yang akan dipanggil berdasarkan masukan pengguna. LLM diberitahu tentang alat apa yang tersedia untuk agen, memungkinkannya menghasilkan tanggapan yang menginstruksikan agen untuk memanggil alat tersebut. Misalnya, ketika Anda bertanya LLM berapa banyak pelanggan dalam database Anda, LLM akan mengirim tanggapan kembali ke agen yang meminta untuk menjalankan `query_database` alat dengan parameter input tertentu. LLM menentukan alat mana yang akan dipanggil dan input untuk panggilan alat. Agen kemudian mengeksekusi alat, yang mengubah input bahasa alami menjadi panggilan fungsi yang benar secara sintaksis dan menjalankan kueri. Agen memanggil alat atau alat berdasarkan instruksi dari LLM, dan hasil tersebut diteruskan kembali ke LLM. Ini mengambil keuntungan dari kemampuan LLM untuk bernalar atas input berbasis teks dan memilih alat yang sesuai untuk pekerjaan itu.

Gambar berikut menunjukkan bagaimana setiap agen mengelola set alatnya sendiri untuk setiap target.



Akses alat penskalaan dapat menghadirkan tantangan untuk solusi AI agen:

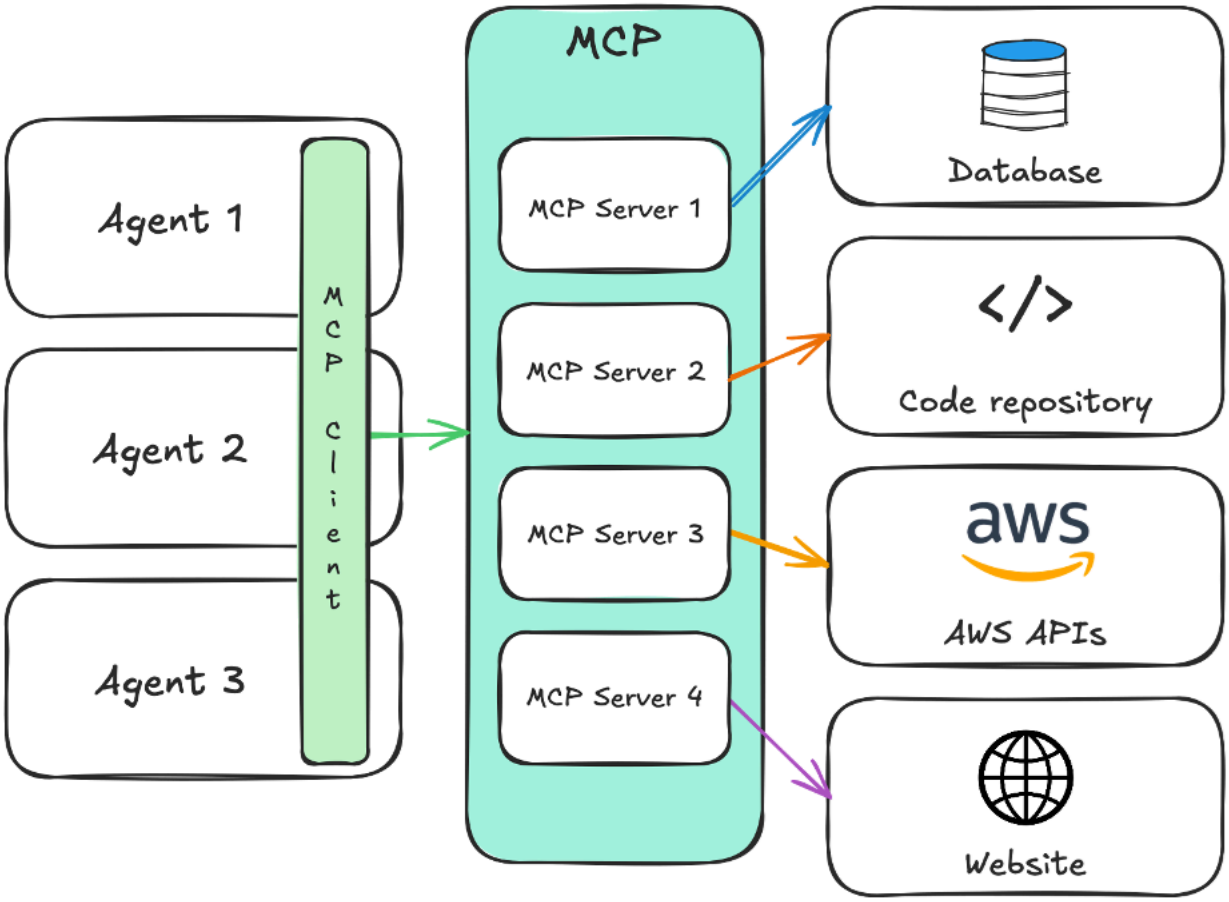
- Jika setiap pengembang membuat alat mereka sendiri untuk kemampuan eksternal yang sama, ada banyak upaya duplikat dan cara non-standar untuk berinteraksi dengan kemampuan eksternal ini. Ini menghasilkan implementasi yang tidak konsisten di seluruh agen Anda. Meskipun Anda dapat memecahkan masalah itu dengan mengembangkan alat standar di perpustakaan dan mendistribusikannya, ini tidak memiliki tata kelola terpusat. Hal ini membuat sulit untuk menegakkan kebijakan keamanan, melacak penggunaan alat, mengelola pembuatan versi di seluruh tim, atau memastikan kepatuhan dengan standar organisasi. Selain itu, ketika Anda menyematkan alat secara langsung dengan agen, Anda harus menerapkan kembali agen Anda setiap kali alat baru dibuat atau yang sudah ada diperbarui.
- Menyediakan alat untuk LLM mengkonsumsi jendela konteksnya. Jendela konteks adalah jumlah token (unit teks yang LLMs memproses — biasanya mewakili kata-kata, bagian kata, atau tanda baca) yang dapat dipertimbangkan oleh model pada satu waktu. LLMs memiliki batas jendela konteks. Alat dan dokumentasinya menggunakan jendela konteks terbatas itu bersama dengan permintaan sistem dan permintaan pengguna. Saat jendela konteks terisi, LLMs dapat mengalami penurunan kinerja karena beberapa faktor: kesulitan dalam mengidentifikasi informasi yang

relevan, peningkatan kompleksitas pemrosesan, dan berkurangnya kapasitas untuk penalaran. Tantangannya diperparah ketika definisi alat, petunjuk sistem, dan riwayat percakapan bersaing untuk ruang jendela konteks terbatas karena disediakan pada setiap pemanggilan LLM.

Dengan demikian, jumlah alat dan bagaimana mereka didokumentasikan memiliki dampak langsung pada kinerja LLM, seperti waktu respons, dan akurasi.

MCP menetapkan standar universal untuk menghubungkan agen ke kemampuan eksternal. Ini biasanya disebut sebagai “USB-C untuk aplikasi AI.” Alih-alih mendaftarkan alat secara langsung dengan agen, server MCP bertindak sebagai perantara untuk alat hosting yang ditemukan dan dipanggil melalui JSON-RPC 2.0. Alih-alih menambahkan puluhan atau ratusan alat berbeda ke agen Anda dan mempertahankannya dari waktu ke waktu, MCP memungkinkan Anda mendaftarkan server MCP yang merangkum alat yang dapat diakses agen Anda. Pendekatan ini menstandarisasi bagaimana alat dikemas, disajikan, dan dipanggil. Ini dapat membantu mengatasi tantangan skala dan tata kelola penggunaan alat di dalam agen Anda. Ini juga memisahkan pengembangan dan operasi agen dari alat yang digunakannya untuk kemampuan eksternal.

Gambar berikut menunjukkan agen yang menggunakan MCP untuk mengakses sumber daya eksternal.



Namun, standar MCP tidak menyelesaikan semua tantangan penskalaan dan tata kelola. Implementasi server MCP harus dikombinasikan dengan desain alat yang efektif, hosting, dan strategi tata kelola perusahaan. Panduan ini memberikan praktik terbaik untuk setiap strategi untuk membantu Anda membangun dan menggunakan MCP sebagai bagian dari solusi AI agen Anda.

Kapan menggunakan MCP

MCP menyediakan infrastruktur strategis untuk menskalakan inisiatif AI agen Anda. Dengan memusatkan manajemen dan tata kelola alat, server MCP mengurangi biaya kumulatif untuk membangun dan memelihara integrasi khusus di beberapa agen. Ini memberikan pengembalian yang meningkat saat ekosistem agen Anda berkembang.

MCP kemungkinan menjadi bagian dari strategi Anda ketika:

- Anda memerlukan tata kelola terpusat untuk bagaimana agen mengakses sistem dan layanan perusahaan, seperti database, alat internal APIs, dan integrasi pihak ketiga.

- Pengembang menghabiskan terlalu banyak waktu untuk menulis integrasi khusus yang tidak konsisten di seluruh implementasi.
- Anda memiliki alat duplikat yang dapat melayani kemampuan umum.
- Anda ingin menawarkan alat atau data milik Anda kepada konsumen eksternal atau sistem agen pihak ketiga melalui antarmuka MCP standar yang diatur, membuka aliran pendapatan baru sambil menjaga keamanan dan kontrol.

Setelah Anda memutuskan bahwa server MCP akan menjadi bagian dari strategi Anda, evaluasi apakah implementasi server MCP open-source yang ada memenuhi kebutuhan Anda, apakah mereka memerlukan peningkatan, atau apakah Anda perlu membangun server khusus. Banyak implementasi server MCP pra-bangun tersedia di repositori publik, dan mencakup kemampuan umum seperti akses sistem file, penjelajahan web, kotak pasir kode, akses basis data, dan integrasi API.

Dalam banyak kasus, server MCP yang sudah ada sudah cukup. Misalnya, AWS menyediakan, server MCP jarak jauh terkelola yang menyediakan asisten dan agen AI akses yang aman dan terautentikasi Layanan AWS melalui interaksi bahasa alami. AWS MCP Server Anda dapat menggunakannya AWS MCP Server untuk melakukan AWS tugas multi-langkah yang kompleks dengan menggabungkan akses real-time ke AWS dokumentasi, panggilan API yang benar secara sintaksis, dan alur kerja pra-bangun yang disebut Agen SOPs yang mengikuti praktik terbaik. AWS AWS terus menguji AWS MCP Server untuk memastikan bahwa agen pelanggan dapat menggunakannya dengan sukses.

Anda harus menguji server MCP yang ada ini dengan agen Anda untuk menentukan apakah mereka memenuhi kasus penggunaan Anda. Jika agen gagal menyelesaikan alur kerja, menghasilkan respons yang salah atau kurang optimal, gagal menavigasi proses multi-langkah yang kompleks, atau melewatkan praktik terbaik atau pertimbangan keamanan khusus domain yang penting, Anda harus mempertimbangkan peningkatan dalam beberapa dimensi.

Ketika server MCP yang ada tidak sepenuhnya memenuhi kebutuhan Anda dan mereka berjuang untuk menggunakan alat yang ada dengan benar atau menghasilkan respons yang akurat, pertimbangkan pendekatan peningkatan ini sebelum membangun server khusus:

- Perkaya konteks agen — Jika agen Anda berjuang untuk menggunakan alat dengan benar atau efisien di server MCP yang ada, pertimbangkan untuk melengkapi definisi alat tersebut dengan dokumentasi atau contoh tambahan. Ini membantu memberikan konteks tambahan untuk LLM.

- Tambahkan alat pelengkap — Perluas server MCP yang ada dengan alat yang mengakses data organisasi atau konteks tambahan yang dibutuhkan agen untuk menyelesaikan alur kerja dengan sukses.
- Tingkatkan yang mendasari APIs — Sederhanakan layanan Anda APIs agar lebih ramah LLM dengan mengurangi kompleksitas parameter, memberikan pesan kesalahan yang lebih jelas, dan menawarkan default yang masuk akal yang dapat digunakan agen.

Saat menggunakan implementasi server MCP yang ada mempercepat pengembangan untuk kemampuan umum, membangun server MCP kustom adalah suatu keharusan ketika kasus penggunaan Anda memerlukan fungsionalitas khusus. Server MCP khusus membantu Anda merangkum keahlian domain, menegakkan standar organisasi, meningkatkan keandalan agen untuk alur kerja yang kompleks, dan mendukung kepatuhan terhadap persyaratan keamanan. Pertimbangkan untuk membangun server MCP khusus dalam situasi berikut:

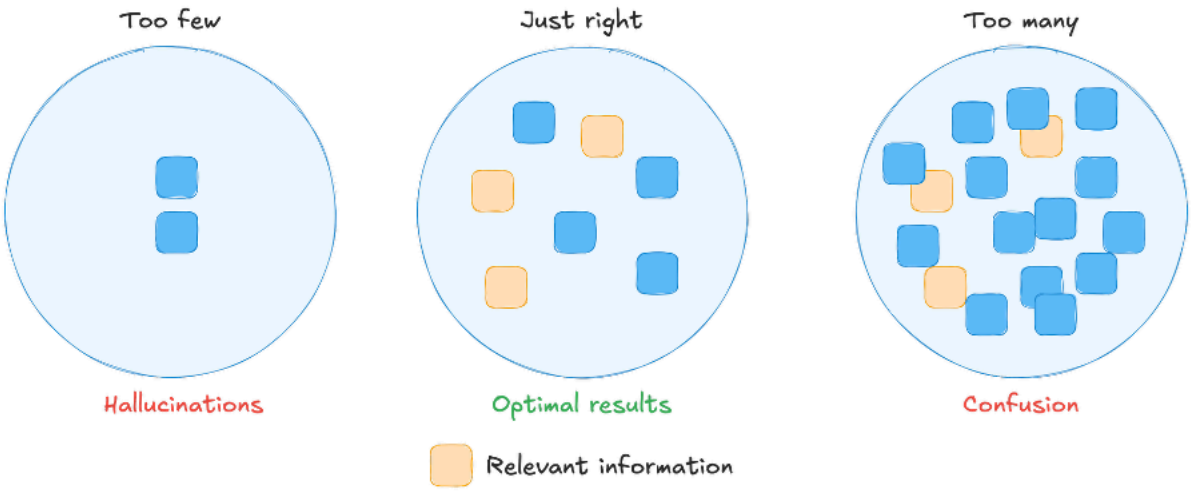
- Alur kerja khusus domain — Alur kerja multi-langkah yang membutuhkan keahlian domain harus dienkapsulasi dalam alat MCP khusus ketika pengetahuan yang diperlukan tidak ditangkap dalam dokumentasi API. Misalnya, alih-alih membiarkan agen mengatur jalur data perawatan kesehatan yang kompleks yang harus memvalidasi kepatuhan Undang-Undang Portabilitas dan Akuntabilitas Asuransi Kesehatan (HIPAA), menganonimkan PII, dan mengubah ke HL7 format FHIR, sediakan alat yang menyematkan keahlian domain secara langsung. `process_patient_data` Ini menghilangkan ketergantungan pada LLM untuk mengatur dan menjalankan langkah-langkah alur kerja dengan benar, yang meningkatkan konsistensi dan kepatuhan.
- Abstraksi jalur emas — Agen mungkin berjuang untuk menerapkan pendekatan yang optimal karena mereka tidak memiliki konteks organisasi dan default ke pola dasar daripada praktik terbaik organisasi. Dalam skenario ini, Anda dapat menerapkan standar preskriptif untuk biaya, kinerja, atau keamanan dengan merangkum jalur emas ini dalam alat MCP khusus. Misalnya, alih-alih membiarkan agen menyebarkan infrastruktur dengan pengaturan default yang mungkin tidak aman atau tidak efisien, sediakan `deploy_secure_infrastructure` alat yang secara langsung menyematkan standar organisasi Anda.
- Orkestrasi multi-layanan yang kompleks — Alih-alih membuat agen mengatur alur kerja yang kompleks dengan mencoba menyimpulkan urutan dan kumpulan layanan yang benar untuk digunakan di setiap langkah, Anda dapat secara deterministik membangun logika itu di dalam alat MCP. Anda mungkin juga ingin memberikan keahlian tentang pola integrasi layanan optimal yang mungkin tidak disadari oleh agen. Ini juga dapat meningkatkan akurasi dan efisiensi agen Anda.
- Praktik terbaik khusus layanan — Ini umum untuk alat yang berfokus pada keamanan yang membantu agen menerapkan kebijakan enkripsi, kontrol akses, dan pola kepatuhan khusus untuk

layanan yang diakses melalui alat agen. Selain itu, jika ada praktik terbaik operasional khusus layanan yang tidak jelas, menggunakan server MCP dapat membantu Anda memastikan bahwa mereka diterapkan dan tidak diserahkan kepada agen untuk dipikirkan.

Strategi desain alat MCP

Tugas utama klien dan server MCP adalah menemukan dan menyajikan alat ke LLM sehingga dapat menggunakannya untuk meningkatkan tanggapannya. Ini menjadikan desain alat MCP salah satu strategi terpenting untuk membangun solusi MCP yang efektif. Dari perspektif model, alat adalah fungsi yang dapat mereka panggil sesuai kebutuhan untuk memberikan respons yang lebih akurat dan lengkap. Antarmuka fungsi mengabstraksi implementasi dasar alat, yang dapat berkisar dari pembungkus di sekitar panggilan API tunggal hingga logika alur kerja yang kompleks.

Namun, Anda harus mencapai keseimbangan dengan jumlah alat yang disediakan untuk LLM. Jika ada terlalu sedikit alat, LLM mungkin tidak dapat mengumpulkan konteks dan informasi yang tepat, sehingga akan membutuhkan tebakan terbaik dengan informasi yang tersedia dalam model. Jika ada terlalu banyak alat, LLM mungkin bingung tentang pemilihan dan urutan alat yang tepat, yang mengarah ke halusinasi. Tujuan Anda adalah untuk mendapatkan jumlah alat yang tepat. Gambar berikut menunjukkan tantangan alat yang terlalu sedikit dan terlalu banyak.



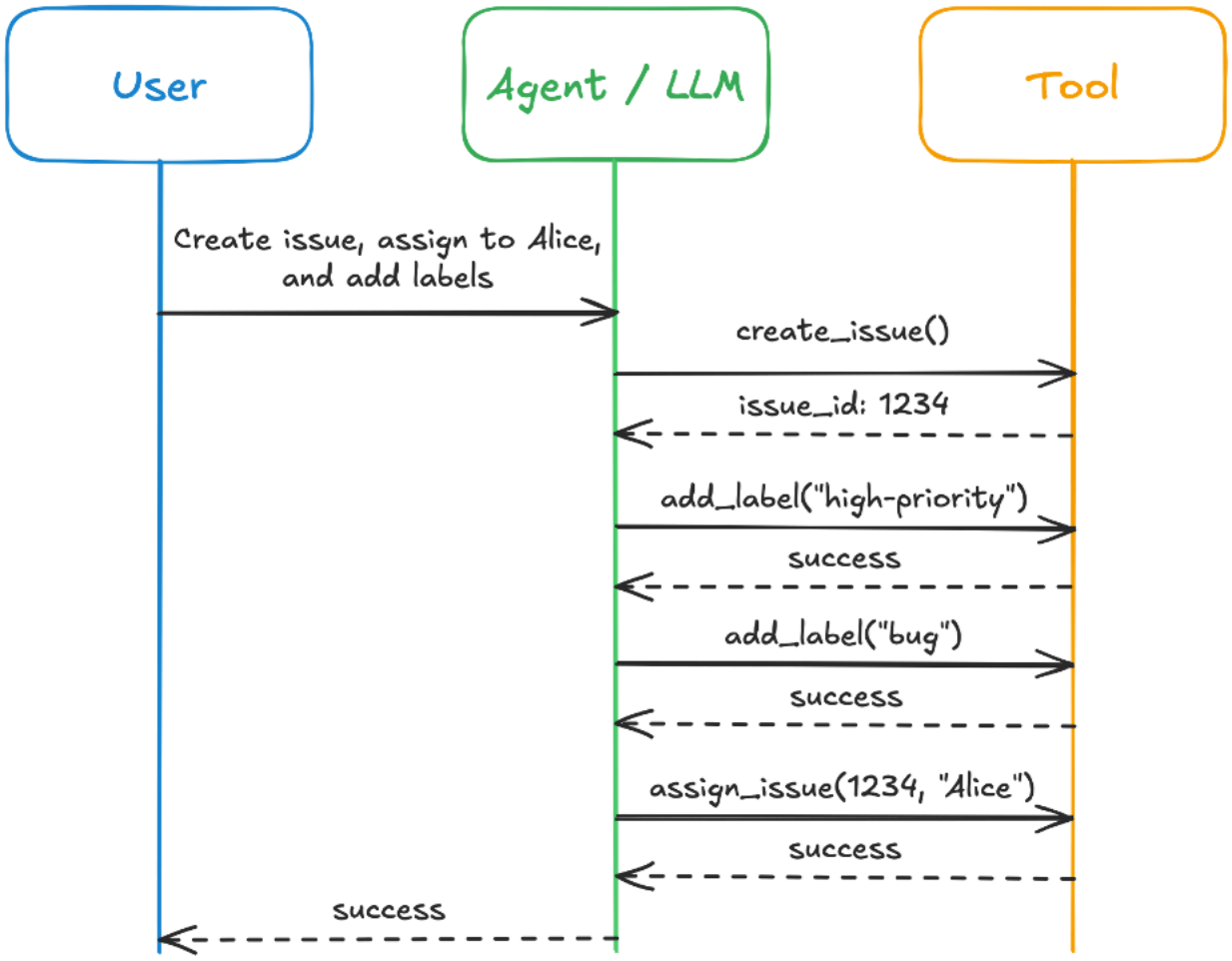
Solusinya membutuhkan pemahaman berapa banyak alat yang harus disediakan dan bagaimana cakupan setiap alat. Perincian alat Anda, apakah mereka memetakan ke panggilan API individual atau alur kerja lengkap, secara langsung memengaruhi jumlah alat yang dibutuhkan agen dan seberapa efektif mereka dapat menggunakannya. Bagian ini memberikan praktik terbaik untuk pelingkupan alat MCP, membuat definisi alat, menemukan alat, dan mengaturnya.

Ruang lingkup alat

Ada dua pendekatan untuk mengembangkan alat: granular dan berbutir kasar.

Granular

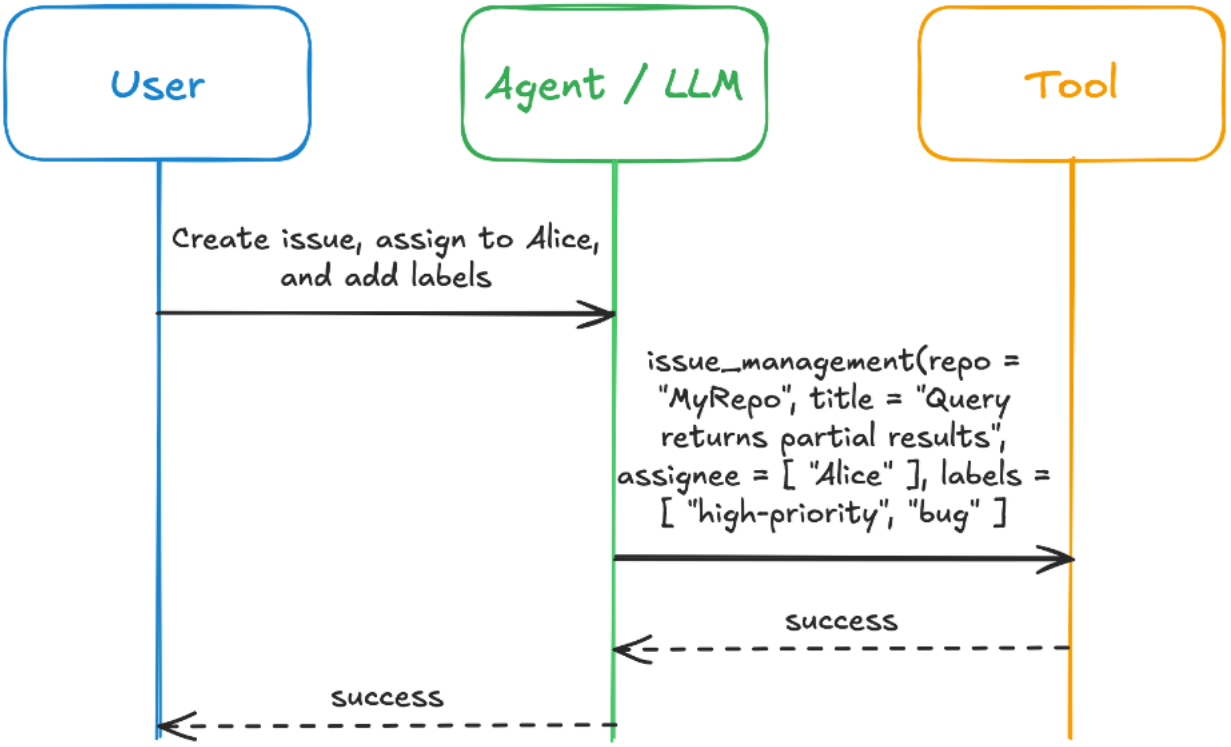
Dalam pendekatan granular, Anda akan membuat alat per API, tindakan, atau kueri. Misalnya, Anda dapat membuat`create_issue`,,, `get_issue` `add_labelassign_issue`, dan `close_issue` alat untuk repositori Git Anda. Ini akan memungkinkan LLM untuk membuat panggilan granular ke setiap API dan mengatur masing-masing sesuai kebutuhan. Pertimbangkan prompt berikut: “Buat masalah untuk layanan produk yang disebut 'Kueri hanya mengembalikan hasil parsial', beri label sebagai bug dan prioritas tinggi, dan tetapkan ke Alice.” Gambar berikut menunjukkan bagaimana tool-per-API pendekatan akan merespons prompt ini.



Dalam pendekatan ini prompt sistem dan setiap definisi alat terdaftar disediakan untuk LLM pada setiap panggilan. Ini menggunakan konteks tambahan dan menimbulkan penalti latensi karena setiap panggilan alat mewakili panggilan individu ke LLM. Hal ini juga meningkatkan kompleksitas penanganan kesalahan dalam alur kerja.

Berbutir kasar

Pendekatan berbutir kasar, atau didorong oleh alur kerja, akan menjadi alat yang berorientasi pada alur kerja. Alat ini berfokus pada maksud end-to-end pengguna atas struktur API. Alih-alih a tool-per-API, Anda memiliki satu alat yang secara deterministik memanggil banyak. APIs Menggunakan contoh repositori Git sebelumnya, Anda dapat membuat `create_and_setup_issue` alat yang dipanggil sekali oleh agen. Implementasi alat menciptakan masalah, menambahkan label, dan menetapkannya ke pengguna, berdasarkan parameter yang disediakan untuk alat. Gambar berikut menunjukkan bagaimana pendekatan berbutir kasar akan memproses prompt yang sama.



Pendekatan ini menunjukkan bagaimana semua kompleksitas tetap tersembunyi dari lapisan LLM. Ketika logika orkestrasi tertanam dalam implementasi alat, semua langkah berurutan, logging, logika coba lagi, pemutus sirkuit, dan pembatasan laju dilakukan secara deterministik di alat. Pendekatan berbasis alur kerja membuatnya lebih mudah bagi LLM untuk menggunakan alat yang benar dengan parameter yang tepat. Penting untuk dicatat bahwa beberapa API mungkin sudah menyediakan maksud alur kerja, seperti Amazon `RunInstances` EC2 API. Dalam kasus ini, a tool-per-API dapat memberikan desain berorientasi alur kerja yang Anda inginkan.

Namun, alat bisa menjadi terlalu kasar juga. Jika alat alur kerja tunggal Anda mencoba melakukan terlalu banyak hal dan memiliki banyak parameter yang mungkin, LLM dapat berjuang untuk bernalar tentang cara menggunakan alat dengan benar. Itu juga dapat menciptakan tantangan dengan

pemilihan parameter dan penanganan kesalahan. Dengan demikian, pengembangan alat harus mencapai keseimbangan yang selaras dengan maksud pengguna dan menghindari terlalu sedikit atau terlalu banyak fungsionalitas dalam satu alat. Kami menyarankan Anda merancang alat di sekitar alur kerja pengguna yang lengkap, operasi bundling yang biasanya terjadi bersamaan (seperti tiga atau lebih panggilan API). Kami juga menyarankan Anda menguraikan alat yang melebihi delapan parameter atau lebih atau menangani beberapa maksud pengguna yang berbeda. Uji dengan petunjuk nyata untuk memverifikasi bahwa agen dapat menggunakan setiap alat dengan benar.

Jika Anda memiliki alur kerja yang kompleks dan dinamis yang tidak dapat dengan mudah dienkapsulasi sebagai alat deterministik, Anda dapat mempertimbangkan untuk menggunakan pola tersebut. agent-as-tool Alih-alih agen utama Anda mencoba mengatur tugas-tugas kompleks dalam alur kerja, agen khusus dapat bertindak sebagai alat. Jenis alat ini dapat menerapkan pengambilan keputusan dan percabangan tingkat lanjut, dan mereka dapat menangani kesalahan dan mencoba kembali logika yang tidak dapat dengan mudah dikelola dalam kode deterministik. Ini mirip dengan, tetapi berbeda dari, protokol Agent2Agent (A2A). Protokol A2A saling melengkapi, menyediakan interoperabilitas dan kolaborasi antar agen dalam kerangka kerja agen apa pun.

Kami menyarankan Anda memulai dengan analisis alur kerja Anda dengan memetakan alur kerja pengguna yang paling umum untuk mengidentifikasi kemampuan inti yang dibutuhkan setiap agen. Ini menetapkan toolset minimum Anda yang layak. Berdasarkan pengalaman kami mengembangkan server MCP dalam skala besar, kami merekomendasikan praktik-praktik berikut. Ketika praktik ini bertentangan, prioritaskan maksud pengguna dan alur kerja.

Praktik terbaik untuk pelingkupan alat MCP

- Pikirkan cerita pengguna dan bundel operasi umum — Alat harus memetakan langsung untuk menyelesaikan interaksi pengguna daripada memerlukan orkestrasi beberapa operasi. Jika alur kerja biasanya memerlukan tiga atau lebih panggilan terpisah, gabungkan mereka menjadi satu alat. Ini mengurangi beban kognitif pada LLM, meminimalkan jumlah panggilan alat, mengurangi konsumsi konteks dan latensi yang diperlukan untuk menyelesaikan tugas, dan meningkatkan akurasi dan latensi.
- Batasi parameter hingga delapan atau kurang — Jika alat melebihi delapan parameter, uraikan menjadi beberapa alat. LLMs berjuang dengan pemilihan parameter saat kompleksitas meningkat.

> **Note**
>
> Jika operasi bundling membutuhkan lebih dari delapan parameter, prioritaskan bundling di atas jumlah parameter karena menyederhanakan alur kerja lebih berharga daripada batas parameter yang ketat.

- Operasi baca dan tulis terpisah - Sediakan alat yang berbeda untuk membaca data dan memodifikasinya. Pemisahan ini membuatnya eksplisit ketika agen melakukan operasi yang berpotensi merusak, memungkinkan kebijakan otorisasi yang berbeda, dan mengurangi risiko modifikasi yang tidak diinginkan selama pengumpulan informasi.
- Berikan default yang masuk akal - Alat desain sehingga LLM hanya perlu menentukan parameter yang spesifik untuk permintaan individu. Default mengurangi kompleksitas parameter dan meningkatkan akurasi pemilihan alat dengan meminimalkan informasi yang harus dipertimbangkan LLM.
- Lebih suka eksekusi deterministik — Jadikan eksekusi alat dan output deterministik bila memungkinkan. Alat deterministik lebih andal dan lebih mudah diuji. Untuk alur kerja kompleks yang memerlukan orkestrasi cerdas, logika percabangan, atau penanganan kesalahan lanjutan yang tidak dapat dengan mudah dikelola dalam kode deterministik, pertimbangkan untuk menggunakan agen khusus sebagai alat. Namun, gunakan pola ini secara selektif karena menambah kompleksitas.

Definisi alat

Ketika LLM menerima permintaan yang tidak dapat ditangani secara langsung, LLM akan meninjau alat yang tersedia untuk membantunya menyelesaikan permintaan. LLM memilih alat berdasarkan pemahaman semantiknya tentang nama dan deskripsi alat yang disediakan dan instruksi apa pun yang diberikan dalam prompt. Ini kemudian akan membuat input berdasarkan skema input yang ditentukan dan mengharapkan output berdasarkan skema output. Oleh karena itu, membuat definisi alat deskriptif dan skema input dan output yang divalidasi sangat penting dalam membantu LLM memilih alat secara efektif. Umumnya ada dua pendekatan untuk membuat dokumentasi ini: pendekatan spesifikasi alat dan pendekatan docstring.

Pendekatan spesifikasi alat

Pendekatan yang disarankan adalah langsung mengikuti spesifikasi alat MCP saat mendefinisikan alat. Contoh berikut ditampilkan menggunakan dekorator alat Strands Agent:

```
@tool(
  name = "search_website",
  description = "This tool searches the provided website for semantic matches to the
 query provided",
  inputSchema = {
    "json": {
      "type": "object",
      "properties": {
        "url": {
          "type": "string",
          "description": "The url of the website to load and search."
        },
        "query": {
          "type": "string",
          "description": "The content you want to try and match in the website."
      }
    },
    "required": ["url", "query"]
  },
  outputSchema = {
    "json": {
      "type": "object",
      "properties": {
      "results": {
          "type": "array",
          "items": {
            "type": "string"
          }
        }
      }
    }
  }
)
def search_website:
  …
```

Menggunakan bidang standar, sepertiname,, `descriptioninputSchema`, dan `outputSchema` memastikan bahwa setiap alat memiliki dokumentasi yang konsisten yang dapat dipahami oleh LLM dan manusia. Setiap alat harus mendefinisikan bidang ini minimal dan secara opsional memberikan judul dan anotasi, yang merupakan petunjuk opsional tentang perilaku alat. Jika memungkinkan, gunakan enum untuk nilai parameter untuk memudahkan LLM memilih opsi yang benar. Enum bekerja paling baik untuk set terbatas, seperti status atau nilai prioritas, tetapi tidak cocok untuk

teks bentuk bebas, nilai dinamis, angka arbitrer, atau pengidentifikasi sumber daya. Dalam kasus tersebut, berikan deskripsi dan contoh yang jelas sebagai gantinya. Juga sertakan nilai default bila memungkinkan sehingga LLM tidak perlu menebak apa opsi yang benar. Perlu diingat bahwa definisi alat disertakan dalam prompt LLM pada setiap pemanggilan, menghabiskan ruang jendela konteks bersama instruksi sistem dan riwayat percakapan.

Pendekatan Docstring

Pendekatan lain, jika Anda menulis alat Anda dengan Python, menggunakan docstrings untuk memberikan deskripsi, penggunaan, dan output alat. Berikut ini adalah contoh dari pendekatan ini:

```
def search_website(url: str, query: str) -> list:

    """
    This tool loads the specified website and then attempts to find content that
 matches the provided query through semantic search. It provides back a list of strings
 that are the sentences that match the query.
    Args:
      url: the website url to load
      query: the content you want to semantically match in the website
    """
```

Docstrings tidak menerapkan skema atau format standar. Menggunakan pendekatan ini mungkin menghasilkan hasil yang tidak konsisten berdasarkan bagaimana pengembang alat memilih untuk mendokumentasikan setiap alat. Mendefinisikan dan menegakkan standar di seluruh organisasi sangat penting jika Anda mengikuti pendekatan ini.

Praktik terbaik untuk definisi alat MCP

- Ikuti spesifikasi alat MCP - Menyediakan`name`,`description`,`inputSchema`, dan `outputSchema` bidang untuk setiap alat. Untuk implementasi Python, gunakan model Pydantic untuk menyediakan dokumentasi sebaris melalui deskripsi bidang, validasi tipe otomatis, dan nilai terbatas melalui enum. Ini membuat skema mendokumentasikan diri dan meningkatkan pemahaman LLM tentang opsi parameter yang valid.
- Tulis deskripsi sebagai petunjuk — Deskripsi alat adalah instruksi yang memandu pengambilan keputusan LLM. Sertakan komponen penting dari tujuan alat (apa yang dilakukan alat), kapan menggunakannya (pola atau skenario maksud pengguna), konteks output (untuk apa output digunakan), parameter, dan kondisi kesalahan.

- Berikan contoh konkret — Menyertakan contoh alur kerja dengan nilai aktual adalah cara paling efektif untuk memandu LLMs tentang penggunaan alat yang benar.
- Dependensi dokumen secara eksplisit — Sertakan prasyarat, urutan bernomor, perubahan status, dan tindakan tindak lanjut.

Penemuan alat

Ada tiga pendekatan untuk menemukan dan mendaftarkan alat di agen Anda dengan server MCP: definisi statis, penemuan dinamis, dan fungsi pencarian.

Definisi statis

Pertama, Anda dapat secara statis mendefinisikan alat yang tersedia secara langsung dalam kode agen. Dalam pendekatan ini Anda mendefinisikan alat jarak jauh (objek referensi sisi klien dalam kerangka kerja seperti Strands Agent SDK) untuk setiap alat yang disediakan oleh server MCP yang diakses oleh klien MCP. Contoh berikut menggunakan transportasi HTTP streamable:

```
from mcp.client.streamable_http import streamablehttp_client
from strands import Agent
from strands.tools.mcp import MCPClient

streamable_http_mcp_client = MCPClient(
    lambda: streamablehttp_client("https://mcp1:8000/mcp")
)

reverse_text = RemoteTool(
    name="reverseText",
    client=streamable_http_mcp_client
)

agent = Agent(tools=[reverse_text])
```

Pendaftaran alat individual membantu Anda menjadi sangat selektif tentang alat yang Anda sediakan untuk LLM, yang meminimalkan jumlah jendela konteks yang digunakan. Tradeoff adalah bahwa hal itu membutuhkan mengetahui nama-nama alat yang tersedia dan bisa rapuh jika alat yang tersedia berubah di server MCP.

Penemuan dinamis

Pendekatan selanjutnya adalah menggunakan penemuan dinamis dan mendaftarkan semua alat yang tersedia dengan agen. Pendekatan ini menggunakan konteks secara linier karena lebih banyak alat ditambahkan ke server MCP. Berikut ini adalah contoh dari pendekatan ini:

```
from mcp.client.streamable_http import streamablehttp_client
from strands import Agent
from strands.tools.mcp import MCPClient

streamable_http_mcp_client = MCPClient(
    lambda: streamablehttp_client("https://mcp1:8000/mcp")
)

with streamable_http_mcp_client:
    tools = streamable_http_mcp_client.list_tools_sync()
    agent = Agent(tools=tools)
```

Pertimbangkan skenario di mana definisi alat khas mengkonsumsi sekitar 250-500 token (termasuk nama, deskripsi, dan skema). Mendaftarkan 20 alat akan menghabiskan 5.000-10.000 token dari jendela konteks Anda. Ketika Anda memiliki sejumlah kecil server MCP dan Anda memiliki kontrol atas jumlah alat, opsi ini adalah yang paling sederhana untuk diterapkan. Namun, jika daftar alat diharapkan tumbuh, itu dapat menciptakan masalah manajemen konteks diam di agen Anda. Variasi alternatif dari pendekatan ini adalah dengan menggunakan parameter filter alat saat memanggil`list_tools`, seperti yang disediakan oleh Strands Agents SDK, untuk mengurangi jumlah alat yang terdaftar di agen.

Fungsi pencarian

Opsi ketiga adalah menggunakan fungsi pencarian untuk menemukan alat yang relevan selama runtime. Anda mencantumkan semua alat yang tersedia dari server MCP Anda dan kemudian melakukan pencarian semantik atas alat tersebut berdasarkan prompt pengguna. Kemudian, alat yang dihasilkan terdaftar di agen Anda. Amazon Bedrock AgentCore Gateway menyediakan kemampuan pencarian semantik asli yang dapat membuat solusi semacam ini lebih mudah diterapkan.

Praktik terbaik untuk penemuan alat MCP

- Pelestarian jendela konteks — Pilih pendekatan penemuan dan pendaftaran alat yang menghemat sebanyak mungkin jendela konteks Anda.

- Gunakan pemfilteran alat atau kemampuan pencarian semantik - Secara dinamis menyediakan LLM dengan seperangkat alat yang dapat dipilih, yang meningkatkan akurasi dan efektivitasnya dalam memilih alat yang tepat. Pemfilteran alat dapat beroperasi pada nama alat (pencocokan atau pola yang tepat), deskripsi alat (pencocokan semantik), atau tag domain atau kategori. Pencarian semantik sangat efektif untuk mencocokkan maksud pengguna dengan deskripsi alat. Kedua pendekatan mengurangi penggunaan jendela konteks.

Organisasi alat

Menemukan alat yang tepat dan memastikan LLM dapat menggunakannya secara efektif adalah salah satu bagian paling penting dari pengembangan alat yang efektif. Saat Anda mulai mengembangkan server MCP, Anda memerlukan strategi yang menentukan:

- Berapa banyak alat yang masuk ke satu server MCP
- Alat apa yang tidak boleh dimasukkan ke server MCP yang sama
- Cara memberi nama alat agar dapat dicari dan mencegah tabrakan nama (alat berbeda dengan nama yang sama)
- Cara mendokumentasikan alat dan server MCP agar mudah digunakan oleh LLM

Organisasi namespace adalah pola desain yang mencegah tabrakan nama alat, mengelompokkan fungsionalitas terkait, dan memfasilitasi identifikasi alat yang efisien oleh. LLMs Pola ini menetapkan kategorisasi terstruktur yang analog dengan sistem penyimpanan terorganisir daripada akumulasi tidak terstruktur. Kami merekomendasikan domain-noun-verbpola untuk penamaan alat. Misalnya,`github_issue_create`,`github_issue_list`,`github_issue_update`, `github_pullrequest_creategithub_pullrequest_list`, atau`github_pullrequest_merge`. Keuntungan dari pola ini terbukti ketika memeriksa perilaku penyortiran abjad. Ketika alat dicantumkan menurut abjad, semua operasi terkait masalah mengelompokkan bersama-sama (`create`,,`update`)`list`, diikuti oleh operasi permintaan tarik (`create`,,). `list` `merge` Kata benda (tipe sumber daya) berfungsi sebagai batas organisasi. Struktur ini memfasilitasi pemindaian alat LLM dan navigasi dokumentasi manusia karena fungsionalitas terkait secara alami mengelompokkan bersama.

Server MCP harus dibatasi pada tingkat domain tetapi dapat dibagi lagi berdasarkan pemisahan tugas untuk kemampuan yang disediakannya. Misalnya, Anda mungkin memiliki server MCP terpisah untuk operasi tulis dan operasi baca ke database. Untuk menerapkan pemisahan ini, Anda disarankan untuk menerapkan pagar pembatas di tingkat agen yang membatasi server MCP mana

yang dapat diakses berdasarkan maksud dan izin pengguna. Hal ini dapat dicapai melalui kombinasi berikut:

- Pemuatan server bersyarat - Muat server MCP hanya-baca hanya ketika agen mendeteksi operasi baca di input pengguna.
- Pemfilteran berbasis izin - Gunakan otorisasi pengguna untuk memberikan akses ke hanya server MCP yang sesuai.

Akhirnya, Anda akan ingin membuat batas atas pada jumlah alat yang disediakan oleh server MCP. Jangan membuat asumsi tentang bagaimana agen akan menggunakan server MCP Anda. Mereka mungkin secara naif mencantumkan semua alat yang tersedia dan memberikan semuanya ke LLM. Jika Anda memiliki lebih dari 50 alat dalam satu server, Anda harus mempertimbangkan untuk membaginya menjadi beberapa server.

Praktik terbaik untuk organisasi alat MPC

- Gunakan standar domain-noun-verb penamaan untuk alat — Terapkan strategi untuk mencegah tabrakan nama di server MCP dan agen.
- Tetapkan batas atas — Batasi jumlah alat dalam satu server MCP.
- Membagi server MCP — Gunakan pemisahan tugas untuk membagi server MCP menjadi kelompok-kelompok logis.

Strategi hosting MCP

Mengabstraksi alat yang tersedia ke server MCP memisahkan pengembangan agen Anda dari alat yang tersedia. Ini memperkenalkan tantangan di mana Anda meng-host server MCP Anda dan bagaimana alat diatur di dalam server tersebut.

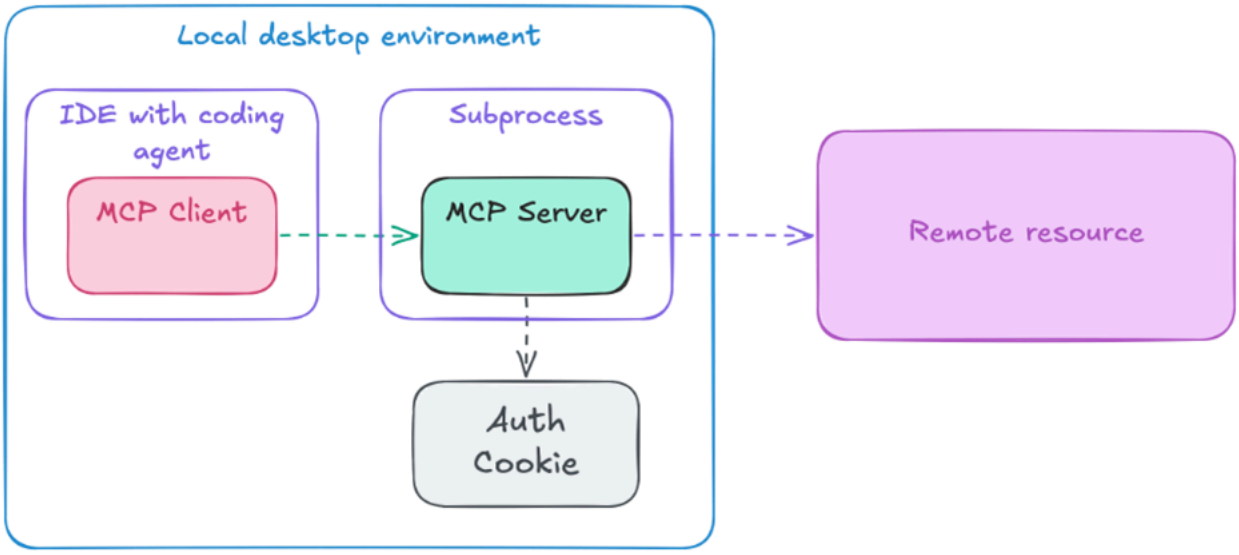
Pendekatan hosting

Ada tiga opsi untuk menghosting server MCP Anda: menjalankannya secara lokal di mesin pengguna akhir, menghosting mereka dari jarak jauh, atau menghosting mereka melalui gateway MCP. Setiap opsi memiliki kelebihan dan pengorbanan.

Hosting lokal

Hosting lokal menjalankan server MCP sebagai subproses pada mesin lokal Anda bersama dengan agen yang berkomunikasi dengan server dengan menggunakan JSON-RPC melalui aliran input dan output standar. Pendekatan ini tidak memerlukan otentikasi antara klien dan server. Alat dapat berinteraksi dengan aplikasi dan file lokal, menggunakan kredensil yang disimpan secara lokal, dan mereka mewarisi akses jaringan dari mesin lokal pengguna. Ini adalah pola hosting yang paling sederhana dan memiliki beberapa manfaat.

Banyak pelanggan memulai dengan MCP menggunakan server lokal. Mereka memungkinkan para insinyur untuk dengan cepat mengulangi dan memecahkan berbagai masalah dari lingkungan lokal mereka. Pertimbangkan server MCP yang terhubung ke repositori Git yang digunakan asisten pengkodean insinyur yang kami gunakan. Menjaga server MCP lokal sangat masuk akal karena dapat menggunakan kredensil unik insinyur untuk mengakses repositori, dan itu tidak menambahkan panggilan jaringan tambahan ke server MCP jarak jauh. Gambar berikut menunjukkan server MCP yang dihosting secara lokal yang digunakan dengan agen pengkodean dalam IDE.



Untuk jenis penyebaran ini, Anda harus mempertimbangkan bagaimana server MCP dikembangkan dan didistribusikan. Sebagian besar pelanggan mengembangkan registri MCP di mana server dapat didaftarkan dan diunduh oleh pengguna akhir. Ini sangat mirip dengan registri kontainer di mana pengguna dapat mencari kemampuan tertentu dan menemukan server MCP yang sesuai dengan kebutuhan mereka.

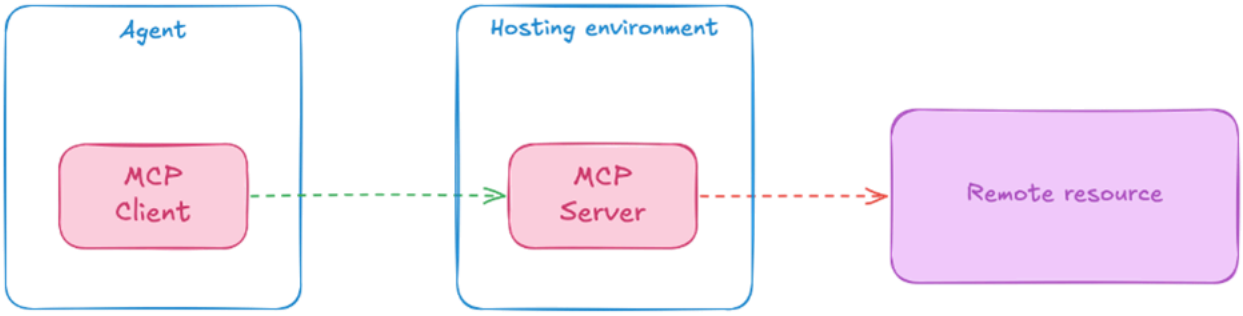
Ada pendaftar MCP publik, seperti Registri MCP Resmi, dan ada pendaftar yang dihosting secara pribadi. Organizations biasanya menyelaraskan strategi registri MCP mereka dengan kebijakan yang ada seputar distribusi perangkat lunak open source, pendaftar kontainer, dan manajemen paket internal. Anda harus mempertimbangkan faktor-faktor seperti pemindaian keamanan, alur kerja persetujuan, dan persyaratan kepatuhan.

Namun, hosting lokal memperkenalkan tantangan operasional yang harus dipertimbangkan organisasi. Pertama, pengguna akhir harus menemukan, mengunduh, dan mengkonfigurasi server MCP secara independen. Ini dapat menambah kerumitan untuk memulai dengan setiap server MCP individu yang mereka gunakan secara lokal. Kedua, Anda tidak dapat mengontrol siklus hidup server MCP, yang berarti bahwa pengguna dapat terus menjalankan versi usang secara lokal dengan kerentanan keamanan atau fitur yang hilang. Ini dapat mempersulit memenuhi persyaratan kepatuhan. Beberapa IDEs dan alat CLI, seperti Kiro, memungkinkan organisasi untuk mengelola dan mengontrol alat MCP mana yang tersedia, memastikan konsistensi dan keamanan di seluruh tim.

Hosting jarak jauh

Opsi kedua adalah meng-host server MCP jarak jauh yang diakses melalui HTTP atau HTTPS. Ini menyediakan akses ke klien yang terhubung dengan jaringan. Menggunakan hosting jarak

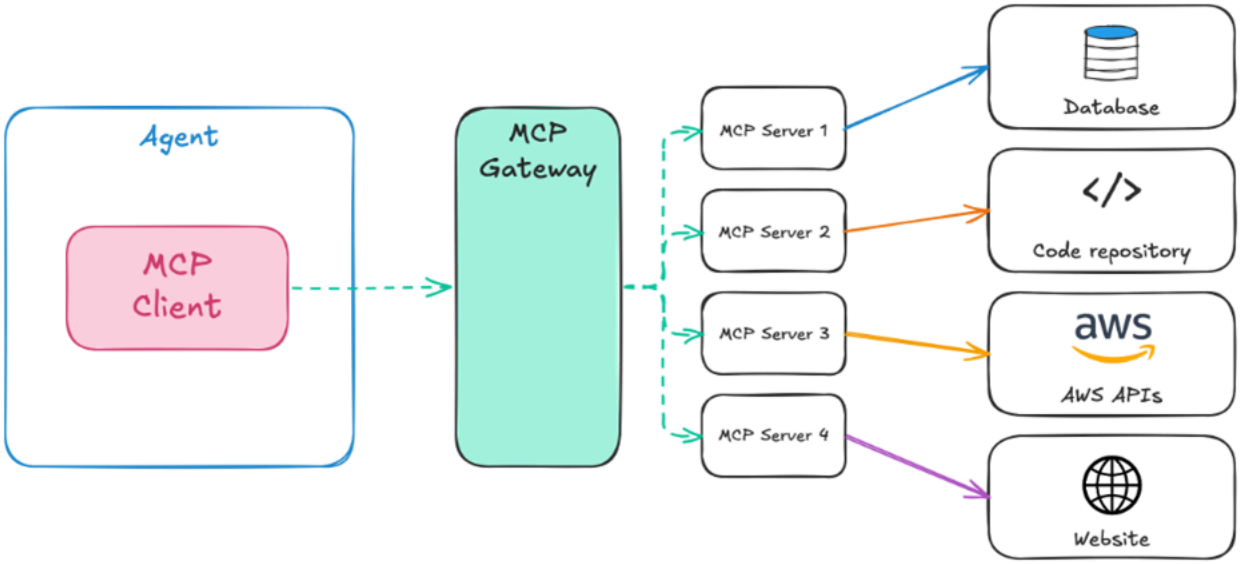
jauh memungkinkan Anda untuk mengontrol akses ke sumber daya dan kemampuan MCP secara terpusat, menerapkan otentikasi dan otorisasi, dan mengontrol versi dan pembaruan logika server MCP. Hosting jarak jauh masih memerlukan penggunaan registri MCP sehingga pengguna akhir dapat menemukan server MCP yang ingin mereka gunakan dengan agen mereka. Gambar berikut menunjukkan pendekatan hosting jarak jauh.



Dari perspektif pengembangan agen, pengalamannya serupa apakah server MCP lokal atau jarak jauh. Perubahan yang paling signifikan adalah menerapkan otentikasi dan otorisasi, termasuk akses agen ke server MCP dan akses server ke sumber daya eksternal. Implementasi server MCP jarak jauh harus direncanakan dengan cermat untuk mempertimbangkan akses multi-penyewa dan manajemen hak istimewa. Bab strategi tata kelola MCP berisi informasi lebih lanjut tentang pertimbangan otentikasi dan otorisasi.

Gerbang MCP

Opsi terakhir adalah menggunakan gateway MCP. Gateway MCP bertindak sebagai proxy terpusat antara klien dan server MCP, dan mereka mengatur akses ke server MCP terdaftar. Tanpa gateway, setiap agen perlu mendaftarkan setiap server MCP jarak jauh yang mungkin ingin digunakan. Gateway memungkinkan agen untuk terhubung ke satu titik akhir yang mengelola otentikasi, otorisasi, perutean, dan terjemahan protokol. Server dan alat MCP baru dapat ditambahkan secara dinamis dan segera tersedia untuk agen. Gambar berikut menunjukkan pendekatan gateway MCP.



Beberapa solusi gateway, seperti Docker MCP Gateway, juga mengelola siklus hidup server MCP, meluncurkan server sesuai permintaan sesuai kebutuhan. Gerbang MCP, seperti Amazon Bedrock AgentCore Gateway, juga dapat membantu mengelola penemuan alat dengan menyediakan kemampuan pencarian semantik asli. Ini memberi agen satu titik akhir untuk terhubung dengan klien MCP dan membantu mengoptimalkan penggunaan jendela konteks mereka. Hasilnya adalah agen sederhana yang dapat memilih dan menggunakan alat MCP secara efektif. Namun, ia memiliki tantangan terkait identitas yang serupa dengan pendekatan server MCP jarak jauh.

Praktik terbaik untuk hosting server MCP

- Spektrum opsi hosting tidak satu ukuran cocok untuk semua. Sebagian besar penggunaan server MCP saat ini bersifat lokal.
- Saat Anda mulai menggunakan server MCP jarak jauh, pertimbangan utama Anda adalah otentikasi dan otorisasi yang konsisten ke server MCP dan bagaimana server MCP melakukan otentikasi dan otorisasi ke sumber daya hilir.
- MCP gateway menyederhanakan konektivitas dan otentikasi dan otorisasi untuk hosting beberapa server MCP jarak jauh. Mereka juga menyediakan kemampuan untuk meningkatkan manajemen jendela konteks dengan mencari alat yang berlaku.

Strategi tata kelola MCP

Kemampuan penting lainnya yang ditawarkan MCP kepada organisasi adalah dukungan untuk tata kelola terpusat. Strategi tata kelola MCP Anda harus menangani otentikasi dan otorisasi ke server MCP serta sumber daya yang mereka akses. Ini juga harus mengatasi pembatasan tarif untuk melindungi sumber daya hilir, metrik operasional untuk memantau penggunaan dan kinerja alat, dan mengelola penyebaran dan distribusi server MCP.

Autentikasi dan otorisasi

Salah satu bagian terpenting dari strategi otentikasi dan otorisasi Anda adalah mengelola akses sumber daya hilir dari server MCP. Ketika pengguna memanggil agen, otentikasi dan otorisasi dilakukan untuk memastikan pengguna memiliki izin untuk memanggil agen. Kemudian, agen mengatur memanggil alat khusus di server MCP. Anda perlu memutuskan bagaimana mengotorisasi akses berdasarkan per-alat.

Salah satu opsi adalah machine-to-machine otorisasi, di mana persetujuan atau interaksi pengguna tidak diperlukan. Misalnya, pemanggilan agen berbasis waktu menggunakan server MCP untuk mengumpulkan log dari aplikasi dan menganalisisnya. Dalam skenario ini, agen pra-otorisasi untuk mengakses data yang ditentukan. Opsi kedua adalah akses yang didelegasikan pengguna, di mana pengguna memberikan persetujuan mereka untuk mengakses data dan sumber daya khusus pengguna.

Tabel berikut menunjukkan otentikasi dan otorisasi pola.

Faktor	Akses yang didelegasikan pengguna	M achine-to-machine
Kepemilikan data	Otorisasi khusus pengguna untuk data	Sistem atau data seluruh organisasi
Interaksi pengguna	Pengguna hadir dan dapat menyetujui	Tidak ada interaksi pengguna
Waktu operasi	Interaktif atau real-time	Latar belakang, terjadwal, atau batch

Ruang lingkup izin	Izin bervariasi menurut pengguna	Izin yang konsisten di tingkat agen

Akses yang didelegasikan pengguna memerlukan implementasi yang cermat dan harus dikembangkan bersama tim keamanan Anda. Agen harus dapat mengevaluasi alat mana yang telah dipilih LLM dan apakah mereka memerlukan otorisasi tambahan. Alat MCP harus menyertakan deskripsi untuk menunjukkan persyaratan otentikasi dan otorisasi mereka dan di mana untuk mengambil token akses. Aplikasi klien harus mendukung permintaan otentikasi perantara, dan klien MCP harus memberikan kredensi yang diambil kembali ke agen untuk setiap panggilan alat.

Anda harus memastikan bahwa alat MCP selalu memiliki token mereka sendiri untuk mengakses kemampuan eksternal dan bahwa akses dicatat dan diaudit. Kredensi pengguna tidak boleh disebarkan melalui sistem agen Anda. Misalnya, server MCP Anda tidak boleh menggunakan token yang sama untuk mengakses data yang digunakan untuk memanggil agen. Panggilan hilir harus menggunakan token yang dibuat khusus dengan cakupan khusus. Ini membantu menyediakan pagar pembatas tambahan untuk mencegah akses data yang tidak diinginkan atas nama tindakan. Ini juga dapat membantu mencegah halusinasi menghasilkan hasil yang tidak diinginkan. Bayangkan bahwa pengguna dengan izin admin penuh meminta agen untuk mengkloning database produksi untuk digunakan dalam pra-produksi. Untuk melakukannya, pengguna hanya perlu `READ` dan `CREATE` izin. Katakanlah LLM berhalusinasi dan percaya perlu membersihkan database lama sebagai bagian dari permintaan ini. Jika menggunakan kembali kredensil pengguna, kemungkinan akan berhasil karena kredensil asli pengguna memiliki izin. `DELETE` Sebaliknya, jika server MCP menggunakan token yang sengaja dicakup-down untuk permintaan dengan adil `READ` dan `CREATE` izin, upaya untuk menghapus basis data produksi akan gagal.

Anda dapat menggunakan Amazon Bedrock AgentCore Identity untuk membantu menerapkan pola-pola ini. Pastikan Anda membuat pilihan yang disengaja tentang apakah izin untuk membuat daftar dan memanggil alat yang dihosting oleh server MCP menyiratkan izin ke kemampuan eksternal yang mengekspos server MCP. Aliran identitas ini dari server MCP ke sumber daya dan kembali ke pengguna tergantung pada jenis layanan otentikasi dan otorisasi yang digunakan. Anda harus memutuskan bagaimana ini ditangani dalam skala besar untuk server MCP Anda.

Saat merancang pola otentikasi dan otorisasi Anda, terapkan mekanisme isolasi token yang mengambil token akses berbeda untuk setiap alat yang diakses. Jangan gunakan kembali token antara alat dan server. AgentCore Identitas menyediakan kemampuan isolasi token ini. Ini secara otomatis mengelola token beban kerja (untuk machine-to-machine otentikasi) dan token pengguna

(untuk akses yang didelegasikan pengguna) untuk memastikan pemisahan yang tepat dan mencegah eskalasi izin. Ini sangat penting ketika menggabungkan server MCP jarak jauh atau gateway MCP.

Praktik terbaik untuk otentikasi dan otorisasi MCP

- Pemisahan token — Jangan berikan token pembawa dari penelepon ke layanan hilir. Validasi bidang aud (audiens) cocok dengan server yang menerima token. Klaim audiens menentukan layanan mana yang dimaksudkan untuk token, mencegah penggunaan kembali token yang tidak sah di berbagai server MCP.
- Pilih pendekatan akses — Pilih antara machine-to-machine dan akses yang didelegasikan pengguna untuk setiap alat yang disediakan server MCP Anda. Pertimbangkan untuk mengelompokkan alat bersama di server MCP yang sama yang menggunakan pola otentikasi yang sama.

Mengontrol beban

Seperti halnya sistem terdistribusi, Anda harus mempertimbangkan cara mengontrol beban di armada server MCP Anda. Pertama, Anda mempertimbangkan apakah akan menerapkan pembatasan tarif di server MCP Anda dan di mana menerapkan batasan. Jika Anda memilih untuk tidak menerapkan pembatasan tarif, Anda meneruskan pembatasan tarif apa pun yang dilakukan oleh sumber daya hilir. Banyak sistem memilih untuk menilai batas berdasarkan atribut permintaan, seperti ID pengguna atau akun. Validasi bahwa permintaan yang dikirim ke layanan hilir membawa atribut yang sama sehingga beberapa pengguna tidak terpengaruh oleh beban yang didorong oleh pengguna lain.

Jika Anda memilih untuk menerapkan pembatasan tarif, pendekatan yang disarankan adalah menerapkan pembatasan tarif primer di tingkat server MCP, dengan layanan backend memberikan perlindungan sekunder dan agen menyesuaikan perilaku mereka berdasarkan umpan balik batas tarif. Pertimbangkan apakah batas tarifnya adalah server per MCP atau per-alat. Batas tingkat server per MCP membantu melindungi armada dan layanan server MCP Anda di lingkungan multi-penyewa. Namun, itu bisa sangat kasar. Batas tarif per alat dirancang untuk mencegah sumber daya hilir yang berlebihan yang mungkin tidak cukup membatasi diri. Jika alat memanggil beberapa APIs, Anda harus menetapkan batas tarif untuk menyelaraskan ke tingkat terendah yang diizinkan oleh mereka APIs.

Melewati informasi batas tingkat dalam header HTTP juga dapat menjadi metrik yang berguna bagi pengguna dan sistem otomatis untuk membantu mengelola tingkat permintaan mereka sendiri dan

strategi coba lagi. Misalnya, Anda dapat mengirim header ini kembali ke agen dari server MCP Anda, seperti yang ditunjukkan pada contoh berikut:

```
X-RateLimit-Limit: 100
X-RateLimit-Remaining: 45
X-RateLimit-Reset: 1640995200
```

Selain itu, pertimbangkan penumpahan beban untuk melindungi layanan secara keseluruhan ketika tidak ada pelanggan tunggal yang melebihi batas laju tetapi beban memengaruhi kinerja sistem.

Praktik terbaik untuk mengendalikan beban

- Pilih pendekatan pembatas kecepatan - Rencanakan untuk menilai membatasi pengguna individu berdasarkan penggunaan sumber daya hilir atau melalui penggunaan server dan alat MCP Anda.
- Pertimbangkan penumpahan beban — Lindungi armada server MCP Anda dari kelebihan beban umum yang tidak didorong oleh satu atau segelintir pelanggan.

Metrik operasional

Metrik kunci untuk menangkap implementasi MCP harus fokus pada pengalaman pelanggan yang mereka berikan. Metrik ini biasanya mencakup penggunaan token, akurasi pemilihan alat, jumlah alat yang terdaftar dengan agen, dan latensi alat. Misalnya, memantau token keluaran yang dikembalikan oleh setiap alat memungkinkan Anda menyetel alarm saat alat melebihi ambang batas untuk penggunaan jendela konteks. Ketika alat melebihi ambang batas itu, Anda mungkin ingin meninjau perilaku alat tersebut. Ini terkait dengan strategi desain alat MCP juga. Metrik akurasi pemilihan alat menunjukkan seberapa baik agen memilih alat yang sesuai untuk tugas tertentu, sementara kecepatan eksekusi dan tingkat keberhasilan menyoroti kemacetan kinerja dan masalah keandalan.

Misalnya, untuk mengevaluasi metrik akurasi pemilihan alat dan penggunaan alat, AWS tim membuat kumpulan data emas untuk pengujian regresi. Kumpulan data dihasilkan secara sintetis dengan menggunakan LLMs log pemanggilan API historis pada kueri pengguna. Menggunakan metrik pemilihan alat dan penggunaan alat yang telah ditentukan sebelumnya (seperti akurasi pemilihan alat, akurasi parameter alat, dan akurasi panggilan fungsi multi-putaran), AWS tim dapat secara objektif mengevaluasi kemampuan agen AI untuk mengidentifikasi alat yang sesuai dengan benar, mengisi parameter mereka dengan nilai yang akurat, dan mempertahankan urutan pemanggilan alat yang koheren di seluruh belokan percakapan.

Mengukur metrik tentang jumlah alat yang terdaftar dengan agen dapat membantu Anda mengidentifikasi tantangan manajemen jendela konteks potensial serta perubahan dalam alat yang tersedia yang disajikan oleh server MCP. Anda harus secara teratur meninjau metrik operasional yang menunjukkan pengalaman pengguna dengan server dan alat MCP Anda.

Kontributor

Mengotorisasi

- Alex Torres, Arsitek Solusi Sr., AWS
- Saikat Gomes, Manajer Solusi Pelanggan Sr., AWS
- Mike Haken, Sr. Arsitek Solusi Utama, AWS
- Sreeja Das, Insinyur Utama, AWS

Meninjau

- Ted Swinyar, Manajer Arsitek Solusi, AWS
- Raju Patil, Ilmuwan Data Senior, AWS

Penulisan teknis

- Lilly AbouHarb, Penulis Teknis Senior, 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	—	Maret 16, 2026

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.