AWS Well-Architected Framework

Pilar Keunggulan Operasional

Pilar Keunggulan Operasional: AWS Well-Architected Framework

Table of Contents

Pilar Keunggulan Operasional - AWS Well-Architected Framework

Tanggal publikasi: 3 Oktober 2023 (Revisi Dokumen)

Laporan ini berfokus pada pilar keunggulan operasional AWS Well-Architected Framework. Laporan ini menyediakan panduan untuk membantu Anda menerapkan praktik terbaik dalam desain, pengiriman, dan pemeliharaan beban kerja AWS.

Pengantar

Dengan AWS Well-Architected Framework Anda dapat memahami manfaat dan risiko keputusan yang Anda ambil selama membangun beban kerja di AWS. Dengan menggunakan Kerangka Kerja ini, Anda akan mengetahui praktik terbaik operasional dan arsitektur untuk mendesain dan mengoperasikan beban kerja yang andal, aman, efisien, hemat biaya, dan ramah lingkungan di cloud. Kerangka Kerja ini menyediakan cara untuk secara terus menerus menilai operasi dan arsitektur Anda berdasarkan praktik terbaik dan mengidentifikasi area yang perlu diperbaiki. Kami percaya bahwa memiliki beban kerja Well-Architected yang didesain dengan mempertimbangkan operasi sangat meningkatkan kemungkinan keberhasilan bisnis.

Kerangka kerja ini didasarkan pada enam pilar:

- Keunggulan Operasional
- Keamanan
- Keandalan
- Efisiensi Kinerja
- Optimasi Biaya
- Pelestarian Lingkungan

Artikel ini berfokus pada pilar keunggulan operasional dan cara menerapkannya sebagai fondasi solusi Well-Architected Anda. Keunggulan operasional sulit dicapai di lingkungan di mana operasi dianggap sebagai fungsi yang terpisah dan berbeda dari lini-lini bisnis dan tim pengembangan yang didukungnya. Dengan mengadopsi praktik-praktik dalam artikel ini, Anda dapat membangun arsitektur yang menyediakan wawasan tentang statusnya, diaktifkan untuk operasi dan respons peristiwa yang efektif dan efisien, dan dapat terus meningkatkan dan mendukung tujuan bisnis Anda.

Artikel ini dimaksudkan untuk orang-orang yang memiliki peran di bidang teknologi, seperti kepala pejabat teknologi (chief technology officer/CTO), arsitek, developer, dan anggota tim operasi. Setelah membaca artikel ini, Anda akan memahami praktik terbaik AWS dan strategi yang digunakan ketika merancang arsitektur cloud untuk keunggulan operasional. Artikel ini tidak menyediakan detail implementasi atau pola arsitektur. Namun, artikel ini menyertakan referensi ke sumber daya tepat untuk informasi ini.

Keunggulan operasional

Di Amazon, kami mendefinisikan keunggulan operasional sebagai komitmen untuk membangun perangkat lunak dengan benar sambil memberikan pengalaman pelanggan yang luar biasa secara konsisten. Keunggulan operasional berisi praktik terbaik untuk mengatur tim Anda, mendesain beban kerja Anda, mengoperasikannya dalam skala besar, dan mengembangkannya seiring waktu. Keunggulan operasional membantu tim Anda untuk lebih memfokuskan waktunya dalam membangun fitur baru yang menguntungkan pelanggan, dan lebih sedikit waktu untuk pemeliharaan dan pemecahan masalah. Untuk membangun dengan benar, kami mengandalkan praktik terbaik yang menghasilkan sistem yang berjalan dengan baik, beban kerja yang seimbang untuk Anda dan tim Anda, dan yang terpenting, pengalaman pelanggan yang luar biasa.

Sasaran keunggulan operasional adalah untuk menyediakan fitur baru dan perbaikan bug kepada pelanggan dengan cepat dan andal. Organisasi yang berinvestasi dalam keunggulan operasional akan secara konsisten membuat pelanggan puas sambil membangun fitur baru, membuat perubahan, dan menangani kegagalan. Dalam prosesnya, keunggulan operasional akan mengarah ke integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) dengan membantu para developer mencapai hasil berkualitas tinggi secara konsisten.

Prinsip desain

Berikut prinsip desain untuk keunggulan operasional di cloud:

- Jalankan operasi sebagai kode: Di cloud, Anda dapat menerapkan bidang rekayasa yang sama dengan yang Anda gunakan untuk kode aplikasi di seluruh lingkungan Anda. Anda dapat menentukan seluruh beban kerja Anda (aplikasi, infrastruktur, dst.) sebagai kode dan memperbaruinya dengan kode. Anda dapat merencanakan prosedur operasi Anda dan mengotomatiskan prosesnya dengan meluncurkannya saat peristiwa terjadi. Dengan melakukan operasi sebagai kode, Anda membatasi kesalahan manusia dan membuat respons yang sesuai terhadap peristiwa.
- Buat perubahan yang sering, kecil, dan dapat diurungkan: Rancang beban kerja yang dapat diskalakan dan digabungkan secara longgar untuk memungkinkan komponen diperbarui secara teratur. Teknik deployment otomatis bersama dengan perubahan yang lebih kecil dan bertahap mengurangi radius ledakan dan memungkinkan pembalikan lebih cepat ketika terjadi kegagalan. Hal ini meningkatkan kepercayaan diri untuk memberikan perubahan yang menguntungkan pada beban kerja Anda sekaligus mempertahankan kualitas dan beradaptasi dengan cepat terhadap perubahan kondisi pasar.

- Tingkatkan prosedur operasi secara berkala: Seiring Anda mengembangkan beban kerja, tingkatkan juga operasi Anda dengan semestinya. Saat Anda menggunakan prosedur operasi, carilah peluang untuk meningkatkannya. Lakukan peninjauan rutin dan validasikan bahwa semua prosedur sudah efektif dan dipahami dengan baik oleh tim. Jika kesenjangan diidentifikasi, perbarui prosedur yang sesuai. Komunikasikan pembaruan prosedural kepada semua pemangku kepentingan dan tim. Ciptakan mekanisme yang menyenangkan dalam operasi Anda untuk berbagi praktik terbaik dan mengedukasi tim.
- Antisipasi kegagalan: Lakukan uji pre-mortem untuk mengidentifikasi sumber yang memiliki potensi kegagalan sehingga dapat dihapus atau dimitigasi. Uji skenario kegagalan Anda dan validasi pemahaman Anda tentang dampaknya. Uji prosedur respons Anda untuk memastikan prosedur sudah efektif dan tim sudah memahami prosesnya. Atur game day secara rutin untuk menguji beban kerja dan respons tim terhadap simulasi peristiwa.
- Ambil pelajaran dari kegagalan operasional yang telah terjadi: Dorong peningkatan dengan belajar dari semua peristiwa dan kegagalan operasional yang telah terjadi. Bagikan materi yang telah dipelajari kepada seluruh tim dan organisasi.
- Gunakan layanan terkelola: Kurangi beban operasional menggunakan layanan terkelola AWS jika memungkinkan. Bangun prosedur operasional seputar interaksi dengan layanan tersebut.
- Terapkan observabilitas untuk wawasan yang dapat ditindaklanjuti: Dapatkan pemahaman komprehensif tentang perilaku beban kerja, performa, keandalan, biaya, dan kesehatan. Tetapkan indikator kinerja utama (KPI) dan manfaatkan telemetri observabilitas untuk membuat keputusan yang lebih tepat dan mengambil tindakan cepat ketika hasil bisnis berisiko. Tingkatkan performa, keandalan, dan biaya secara proaktif berdasarkan data observabilitas yang dapat ditindaklanjuti.

Definisi

Ada empat area praktik terbaik untuk keunggulan operasional di cloud:

- Organisasi
- Persiapan
- Jalankan
- Evolusi

Kepemimpinan organisasi Anda menentukan tujuan bisnis. Organisasi Anda harus memahami kebutuhan dan prioritas serta menggunakannya untuk mengatur dan melakukan pekerjaan guna mendukung pencapaian hasil bisnis. Beban kerja Anda harus memberikan informasi yang diperlukan

untuk mendukungnya. Mengimplementasi layanan untuk mengaktifkan integrasi, deployment, dan penyediaan beban kerja Anda akan menciptakan peningkatan alur perubahan yang menguntungkan menuju produksi dengan mengotomatiskan proses yang repetitif.

Operasi beban kerja Anda dapat memiliki risiko tersendiri. Anda harus memahami risiko tersebut dan mengambil keputusan yang matang untuk memasuki produksi. Tim Anda harus mampu mendukung beban kerja Anda. Dengan metrik bisnis dan operasional yang didapatkan dari hasil bisnis yang diinginkan, Anda dapat memahami kondisi beban kerja, aktivitas operasi, serta respons Anda terhadap insiden. Prioritas Anda akan berubah sesuai dengan kebutuhan bisnis Anda dan perubahan lingkungan bisnis. Gunakan ini sebagai loop umpan balik untuk mendorong peningkatan secara berkelanjutan bagi organisasi Anda dan operasi beban kerja Anda.

Organisasi

Anda perlu memahami prioritas organisasi, struktur organisasi, dan cara organisasi Anda mendukung anggota tim, sehingga mereka dapat mendukung hasil bisnis.

Untuk mewujudkan keunggulan operasional, Anda harus memahami hal-hal berikut:

Topik

Prioritas organisasi

Tim Anda perlu memiliki pemahaman bersama tentang seluruh beban kerja Anda, peran mereka di dalamnya, dan tujuan bisnis bersama untuk menetapkan prioritas yang akan mendukung kesuksesan bisnis. Prioritas yang ditentukan dengan baik akan memaksimalkan manfaat dari upaya Anda. Tinjau prioritas Anda secara rutin sehingga dapat diperbarui sesuai perubahan kebutuhan organisasi.

Praktik terbaik

OPS01-BP01 Mengevaluasi kebutuhan pelanggan eksternal

Libatkan pemangku kepentingan utama, termasuk tim bisnis, pengembangan, dan operasional, untuk menentukan ke mana harus memfokuskan usaha terkait kebutuhan pelanggan eksternal. Hal ini akan memastikan bahwa Anda memiliki pemahaman menyeluruh mengenai dukungan operasi yang dibutuhkan untuk mencapai hasil bisnis yang diinginkan.

Antipola umum:

- Anda memutuskan untuk tidak menyediakan dukungan pelanggan di luar jam kerja, tetapi Anda belum meninjau riwayat data permintaan dukungan. Anda tidak tahu apakah hal ini akan memengaruhi pelanggan Anda.
- Anda mengembangkan fitur baru tetapi belum melibatkan pelanggan untuk mencari tahu apakah hal tersebut diinginkan—jika diinginkan, dalam bentuk apa—dan belum menjalankan eksperimen untuk memvalidasi kebutuhan serta metode pengiriman.

Manfaat menerapkan praktik terbaik ini: Pelanggan yang kebutuhannya terpenuhi berpotensi menjadi pelanggan tetap. Mengevaluasi dan memahami kebutuhan pelanggan eksternal akan menginformasikan cara Anda memprioritaskan usaha untuk memberikan nilai bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Pahami kebutuhan bisnis: Kesuksesan bisnis dapat terwujud dengan adanya tujuan dan pemahaman bersama di seluruh pemangku kepentingan, termasuk tim bisnis, pengembangan, dan operasional.
  - Tinjau tujuan bisnis, kebutuhan, dan prioritas pelanggan eksternal: Libatkan pemangku kepentingan utama, termasuk tim bisnis, pengembangan, dan operasional, untuk mendiskusikan tujuan, kebutuhan, dan prioritas pelanggan eksternal. Hal ini memastikan bahwa Anda memiliki pemahaman menyeluruh mengenai dukungan operasional yang dibutuhkan untuk mencapai hasil bisnis dan pelanggan.
  - Tetapkan pemahaman bersama: Tetapkan pemahaman bersama terkait fungsi bisnis beban kerja, peran masing-masing tim dalam mengoperasikan beban kerja, dan bagaimana faktor-faktor ini mendukung tujuan bisnis bersama bagi seluruh pelanggan internal dan eksternal.

Sumber daya

Dokumen terkait:

- Konsep Kerangka Kerja AWS Well-Architected – Loop umpan balik

OPS01-BP02 Mengevaluasi kebutuhan pelanggan internal

Libatkan pemangku kepentingan utama, termasuk tim bisnis, pengembangan, dan operasional, untuk menentukan ke mana harus memfokuskan usaha terkait kebutuhan pelanggan internal. Hal ini akan memastikan bahwa Anda memiliki pemahaman menyeluruh mengenai dukungan operasi yang dibutuhkan untuk mencapai hasil bisnis yang diinginkan.

Gunakan prioritas yang ditetapkan untuk memfokuskan usaha peningkatan yang dapat memberikan dampak paling besar (misalnya, mengembangkan keterampilan tim, meningkatkan kinerja beban kerja, mengurangi biaya, mengotomatiskan runbook, atau meningkatkan pemantauan). Perbarui prioritas Anda sesuai perubahan kebutuhan.

Antipola umum:

- Anda memutuskan untuk mengubah alokasi alamat IP untuk tim produk tanpa berkonsultasi dengan mereka agar manajemen jaringan menjadi lebih mudah. Anda tidak tahu dampak yang akan ditimbulkan kepada tim produk.
- Anda mengimplementasikan alat pengembangan baru tetapi belum melibatkan pelanggan internal untuk mencari tahu apakah alat itu dibutuhkan atau kompatibel dengan praktik yang sudah ada.
- Anda mengimplementasikan sistem pemantauan baru tetapi belum menghubungi pelanggan internal untuk mencari tahu apakah mereka memiliki kebutuhan pemantauan atau pelaporan yang perlu dipertimbangkan.

Manfaat menerapkan praktik terbaik ini: Mengevaluasi dan memahami kebutuhan pelanggan internal akan menginformasikan cara Anda memprioritaskan usaha untuk memberikan nilai bisnis.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Pahami kebutuhan bisnis: Kesuksesan bisnis dapat terwujud dengan tujuan dan pemahaman bersama di seluruh pemangku kepentingan, termasuk tim bisnis, pengembangan, dan operasional.
  - Tinjau tujuan bisnis, kebutuhan, dan prioritas pelanggan internal: Libatkan pemangku kepentingan utama, termasuk tim bisnis, pengembangan, dan operasional, untuk mendiskusikan tujuan, kebutuhan, dan prioritas pelanggan internal. Hal ini memastikan bahwa Anda memiliki pemahaman menyeluruh mengenai dukungan operasional yang dibutuhkan untuk mencapai hasil bisnis dan pelanggan.

- Tetapkan pemahaman bersama: Tetapkan pemahaman bersama terkait fungsi bisnis beban kerja, peran masing-masing tim dalam mengoperasikan beban kerja, dan bagaimana faktor-faktor ini mendukung tujuan bisnis bersama bagi seluruh pelanggan internal dan eksternal.

Sumber daya

Dokumen terkait:

- Konsep Kerangka Kerja AWS Well-Architected – Loop umpan balik

OPS01-BP03 Evaluasi persyaratan tata kelola

Tata kelola adalah serangkaian kebijakan, aturan, atau kerangka kerja yang digunakan perusahaan untuk mencapai tujuan bisnisnya. Persyaratan tata kelola dibuat dari dalam organisasi Anda. Persyaratan ini dapat memengaruhi jenis teknologi yang Anda pilih atau memengaruhi cara Anda mengoperasikan beban kerja Anda. Sertakan persyaratan tata kelola organisasi ke dalam beban kerja Anda. Konformitas adalah kemampuan untuk menunjukkan bahwa Anda telah mengimplementasikan persyaratan tata kelola.

Hasil yang diinginkan:

- Persyaratan tata kelola disertakan ke dalam operasi dan desain arsitektur beban kerja Anda.
- Anda dapat memberikan bukti bahwa Anda telah mengikuti persyaratan tata kelola.
- Persyaratan tata kelola ditinjau dan diperbarui secara teratur.

Antipola umum:

- Organisasi Anda memerintahkan agar akun root memiliki autentikasi multi-faktor. Anda gagal mengimplementasikan persyaratan ini dan akun root terancam bahaya.
- Selama desain beban kerja Anda, Anda memilih jenis instans yang tidak disetujui oleh departemen IT. Anda tidak dapat meluncurkan beban kerja Anda dan harus mendesain ulang.
- Anda diwajibkan memiliki rencana pemulihan bencana. Anda tidak membuat rencana pemulihan bencana dan beban kerja Anda mengalami pemadaman yang berdurasi lama.
- Tim Anda ingin menggunakan instans baru tetapi persyaratan tata kelola Anda belum diperbarui untuk memungkinkannya.

Manfaat menjalankan praktik terbaik ini:

- Mengikuti persyaratan tata kelola akan menyelaraskan beban kerja Anda dengan kebijakan lebih besar dalam organisasi.
- Persyaratan tata kelola mencerminkan standar industri dan praktik terbaik untuk organisasi Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Identifikasi persyaratan tata kelola melalui kerja sama dengan pemangku kepentingan dan organisasi tata kelola. Sertakan persyaratan tata kelola ke dalam beban kerja Anda. Dapat menunjukkan bukti bahwa Anda telah mengikuti persyaratan tata kelola.

Contoh pelanggan

Di AnyCompany Retail, tim operasi cloud bekerja sama dengan pemangku kepentingan di seluruh organisasi untuk mengembangkan persyaratan tata kelola. Contohnya, mereka melarang akses SSH ke instans Amazon EC2. Jika tim memerlukan akses ke sistem, mereka harus menggunakan AWS Systems Manager Session Manager. Tim operasi cloud secara teratur memperbarui persyaratan tata kelola saat layanan baru tersedia.

Langkah implementasi

1. Identifikasi pemangku kepentingan untuk beban kerja Anda, termasuk semua tim terpusat.
2. Bekerja sama dengan pemangku kepentingan untuk mengidentifikasi persyaratan tata kelola.
3. Setelah Anda membuat daftar, prioritaskan item untuk ditingkatkan, dan mulai implementasikan ke dalam beban kerja Anda.
    a. Gunakan layanan seperti AWS Config untuk membuat tata kelola sebagai kode dan validasikan bahwa persyaratan tata kelola telah diikuti.
    b. Jika Anda menggunakan AWS Organizations, Anda dapat memanfaatkan Kebijakan Kontrol Layanan untuk mengimplementasikan persyaratan tata kelola.
4. Berikan dokumentasi yang memvalidasi implementasinya.

Tingkat upaya untuk rencana implementasi: Sedang. Mengimplementasikan persyaratan tata kelola yang tidak ada dapat mengakibatkan beban kerja Anda harus dikerjakan ulang.

Sumber daya

Praktik terbaik terkait:

- OPS01-BP04 Evaluasi persyaratan kepatuhan - Kepatuhan sama seperti tata kelola tetapi berasal dari luar organisasi.

Dokumen terkait:

- Panduan untuk Manajemen dan Tata Kelola Lingkungan Cloud AWS
- Praktik Terbaik untuk Kebijakan Kontrol Layanan AWS Organizations dalam Lingkungan Multi-Akun
- Tata Kelola dalam AWS Cloud: Keseimbangan yang Tepat Antara Ketangkasan dan Keamanan
- Apa yang dimaksud Tata Kelola, Risiko, dan Kepatuhan (GRC)?

Video terkait:

- AWS Manajemen dan Tata Kelola: Konfigurasi, Kepatuhan, dan Audit - Online Tech Talks AWS
- AWS re:Inforce 2019: Tata Kelola untuk Era Cloud (DEM12-R1)
- AWS re:Invent 2020: Capai kepatuhan sebagai kode menggunakan AWS Config
- AWS re:Invent 2020: Tata kelola tangkas di AWS GovCloud (US)

Contoh terkait:

- Sampel Paket Kepatuhan AWS Config

Layanan terkait:

- AWS Config
- AWS Organizations - Kebijakan Kontrol Layanan

OPS01-BP04 Evaluasi persyaratan kepatuhan

Persyaratan kepatuhan peraturan, industri, dan internal merupakan pendorong penting dalam menentukan prioritas organisasi Anda. Kerangka kerja kepatuhan Anda dapat menghalangi Anda

menggunakan teknologi atau lokasi geografi tertentu. Terapkan uji tuntas jika tidak ada kerangka kerja kepatuhan eksternal yang diidentifikasi. Buat audit atau laporan yang memvalidasi kepatuhan.

Jika Anda mengiklankan bahwa produk Anda memenuhi standar kepatuhan tertentu, Anda harus memiliki proses internal untuk memastikan kepatuhan yang berkelanjutan. Contoh standar kepatuhan antara lain PCI DSS, FedRAMP, dan HIPAA. Standar kepatuhan yang berlaku akan ditentukan oleh berbagai faktor, seperti jenis data yang disimpan atau dikirim oleh solusi dan wilayah geografis mana yang didukung oleh solusi.

Hasil yang diinginkan:

- Persyaratan kepatuhan peraturan, industri, dan internal disertakan ke dalam pemilihan arsitektur.
- Anda dapat memvalidasi kepatuhan dan membuat laporan audit.

Antipola umum:

- Bagian dari beban kerja Anda termasuk dalam kerangka kerja Standar Keamanan Data Industri Kartu Pembayaran (Payment Card Industry Data Security Standard, PCI-DSS) tetapi beban kerja Anda menyimpan data kartu kredit tanpa enkripsi.
- Arsitek dan developer perangkat lunak Anda tidak mengetahui kerangka kerja kepatuhan yang harus ditaati oleh organisasi Anda.
- Audit tahunan Kontrol Sistem dan Organisasi (SOC2) Tipe II akan segera diadakan dan Anda tidak dapat memverifikasi bahwa kontrol sudah ada.

Manfaat menjalankan praktik terbaik ini:

- Mengevaluasi dan memahami persyaratan kepatuhan yang berlaku untuk beban kerja Anda akan menginformasikan bagaimana Anda memprioritaskan usaha untuk memberikan nilai bisnis.
- Anda memilih teknologi dan lokasi yang tepat yang selaras dengan kerangka kerja kepatuhan Anda.
- Mendesain beban kerja Anda agar dapat diaudit memungkinkan Anda membuktikan bahwa Anda menaati kerangka kerja kepatuhan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Mengimplementasikan praktik terbaik ini berarti Anda menyertakan persyaratan kepatuhan ke dalam proses desain arsitektur. Anggota tim Anda mengetahui kerangka kerja kepatuhan yang diperlukan. Anda memvalidasi kepatuhan selaras dengan kerangka kerja.

Contoh pelanggan

AnyCompany Retail menyimpan informasi kartu kredit bagi pelanggan. Developer di tim penyimpanan kartu memahami bahwa mereka harus mematuhi kerangka kerja PCI-DSS. Mereka telah mengambil langkah untuk memverifikasi bahwa informasi kartu kredit disimpan dan diakses dengan aman sesuai dengan kerangka kerja PCI-DSS. Setiap tahun mereka bekerja sama dengan tim keamanan mereka untuk memvalidasi kepatuhan.

Langkah implementasi

1. Bekerjasamalah dengan tim tata kelola dan kepatuhan Anda untuk menentukan kerangka kerja kepatuhan industri, peraturan, atau internal apa yang harus ditaati beban kerja Anda. Sertakan kerangka kerja kepatuhan ke dalam beban kerja Anda.
   a. Validasi kepatuhan sumber daya AWS yang berkelanjutan dengan layanan seperti AWS Compute Optimizer dan AWS Security Hub.
2. Didik anggota tim Anda tentang persyaratan kepatuhan sehingga mereka dapat mengoperasikan dan mengubah beban kerja sesuai dengan persyaratan kepatuhan. Persyaratan kepatuhan harus disertakan dalam pilihan arsitektur dan teknologi.
3. Tergantung pada kerangka kerja kepatuhannya, Anda mungkin diharuskan untuk membuat laporan kepatuhan atau audit. Bekerjasamalah dengan organisasi Anda untuk mengotomatiskan proses ini sebanyak mungkin.
   a. Gunakan layanan seperti AWS Audit Manager untuk memvalidasi kepatuhan dan membuat laporan audit.
   b. Anda dapat mengunduh dokumen kepatuhan dan keamanan AWS dengan AWS Artifact.

Tingkat upaya untuk rencana implementasi: Sedang. Mengimplementasikan kerangka kerja kepatuhan bisa sulit dilakukan. Membuat laporan audit atau dokumen kepatuhan menambahkan kompleksitas tambahan.

Sumber daya

Praktik terbaik terkait:

- SEC01-BP03 Mengidentifikasi dan memvalidasi tujuan kontrol - Tujuan kontrol keamanan merupakan bagian penting dari kepatuhan secara keseluruhan.
- SEC01-BP06 Mengotomatiskan pengujian dan validasi kontrol keamanan di pipeline - Sebagai bagian dari pipeline Anda, validasikan kontrol keamanan. Anda juga dapat membuat dokumentasi kepatuhan untuk perubahan baru.
- SEC07-BP02 Menentukan kontrol perlindungan data - Sejumlah besar kerangka kerja kepatuhan didasarkan pada penanganan data dan kebijakan penyimpanan.
- SEC10-BP03 Menyiapkan kemampuan forensik - Kemampuan forensik terkadang dapat digunakan dalam mengaudit kepatuhan.

Dokumen terkait:

- Pusat Kepatuhan AWS
- Sumber Daya Kepatuhan AWS
- Laporan Resmi Kepatuhan dan Risiko AWS
- Model Tanggung Jawab Bersama AWS
- Layanan AWS dalam cakupan berdasarkan program kepatuhan

Video terkait:

- AWS re:Invent 2020: Capai kepatuhan sebagai kode menggunakan AWS Compute Optimizer
- AWS re:Invent 2021 - Kepatuhan cloud, jaminan, dan audit
- AWS Summit ATL 2022 - Mengimplementasikan kepatuhan, jaminan, dan audit di AWS (COP202)

Contoh terkait:

- PCI DSS dan Praktik Terbaik Keamanan Mendasar AWS di AWS

Layanan terkait:

- AWS Artifact
- AWS Audit Manager
- AWS Compute Optimizer
- AWS Security Hub

OPS01-BP05 Mengevaluasi lanskap ancaman

Evaluasi ancaman pada bisnis (misalnya, persaingan, risiko dan kewajiban bisnis, risiko operasional, serta ancaman keamanan informasi) dan pelihara informasi yang ada di registri risiko. Sertakan dampak risiko ketika menentukan ke mana upaya harus difokuskan.

Kerangka kerja Well-Architected Framework menekankan pembelajaran, pengukuran, dan peningkatan. Framework menyediakan pendekatan yang konsisten bagi Anda untuk mengevaluasi arsitektur, dan mengimplementasikan desain yang akan mengalami penskalaan seiring waktu. AWS menyediakan AWS Well-Architected Tool untuk membantu Anda meninjau pendekatan sebelum pengembangan, status beban kerja Anda sebelum produksi, dan status beban kerja Anda dalam produksi. Anda dapat membandingkannya dengan praktik terbaik arsitektur AWS terkini, memantau keseluruhan status beban kerja Anda, dan mendapatkan wawasan tentang potensi risiko.

Pelanggan AWS memenuhi syarat untuk Tinjauan Well-Architected terpandu tentang beban kerja misi penting mereka untuk mengukur arsitektur mereka berdasarkan praktik terbaik AWS. Pelanggan Dukungan Korporat memenuhi syarat untuk Tinjauan Operasi, yang dirancang untuk membantu mereka mengidentifikasi celah dalam pendekatan operasi di cloud mereka.

Interaksi lintas tim pada tinjauan ini membantu membangun pemahaman bersama tentang beban kerja Anda serta bagaimana peran tim membantu meraih keberhasilan. Kebutuhan yang diidentifikasi melalui tinjauan dapat membantu membentuk prioritas Anda.

AWS Trusted Advisor adalah alat yang menyediakan akses ke set inti pemeriksaan yang menyarankan optimalisasi yang dapat membantu membentuk prioritas Anda. Pelanggan Dukungan Bisnis dan Korporat menerima akses ke pemeriksaan tambahan yang berfokus pada keamanan, keandalan, kinerja, dan optimalisasi biaya yang dapat membantu membentuk prioritas mereka lebih lanjut.

Antipola umum:

- Anda menggunakan pustaka perangkat lunak versi lama dalam produk Anda. Anda tidak tahu bahwa ada pembaruan keamanan pustaka untuk masalah yang mungkin memiliki dampak yang tidak diinginkan pada beban kerja Anda.
- Kompetitor Anda baru saja merilis versi produk mereka yang mengatasi keluhan pelanggan Anda tentang produk Anda. Anda belum memprioritaskan penanganan masalah-masalah yang dikenal ini.

- Pembuat peraturan telah menyasar perusahaan yang tidak mematuhi persyaratan kepatuhan hukum seperti Anda. Anda belum memprioritaskan penanganan persyaratan kepatuhan Anda yang belum terpenuhi.

Manfaat menjalankan praktik terbaik ini: Identifikasi dan pemahaman tentang ancaman terhadap organisasi dan beban kerja Anda dapat membantu Anda menentukan ancaman mana yang harus ditangani, tingkat prioritasnya, serta sumber daya yang diperlukan untuk melakukannya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

- Evaluasi lanskap ancaman: Evaluasi ancaman terhadap bisnis (misalnya kompetisi, risiko dan kewajiban bisnis, risiko operasional, dan ancaman keamanan informasi), sehingga Anda dapat menyertakan dampaknya ketika menentukan ke mana upaya perlu difokuskan.
  - Buletin Keamanan Terkini AWS
  - AWS Trusted Advisor
  - Pelihara model ancaman: Buat dan pelihara model ancaman yang mengidentifikasi potensi ancaman, mitigasi terencana dan sedang diterapkan, serta prioritasnya. Tinjau kemungkinan ancaman yang berwujud insiden, biaya untuk melakukan pemulihan dari insiden tersebut serta perkiraan bahaya yang ditimbulkan, dan biaya untuk mencegah insiden tersebut. Revisi proritas seiring perubahan konten model ancaman.

Sumber daya

Dokumen terkait:

- Kepatuhan AWS Cloud
- Buletin Keamanan Terkini AWS
- AWS Trusted Advisor

OPS01-BP06 Mengevaluasi kompromi

Evaluasi dampak kompromi antarkepentingan yang bertentangan atau pendekatan alternatif, untuk membantu mengambil keputusan yang matang saat menentukan ke mana upaya perlu difokuskan atau memiliki opsi tindakan. Misalnya, meningkatkan kecepatan masuk pasar untuk fitur baru dapat

diprioritaskan daripada optimalisasi biaya, atau Anda bisa memilih basis data relasional untuk data non-relasional guna menyederhanakan upaya migrasi sistem, dibandingkan bermigrasi ke basis data yang dioptimalkan untuk tipe data Anda dan memperbarui aplikasi Anda.

AWS dapat membantu mengedukasi tim Anda tentang AWS dan layanannya untuk meningkatkan pemahaman mereka tentang bagaimana pilihan mereka dapat menimbulkan dampak pada beban kerja. Anda harus menggunakan sumber daya yang disediakan oleh AWS Support (Pusat Pengetahuan AWS, Forum Diskusi AWS, dan Pusat AWS Support) dan Dokumentasi AWS untuk mengedukasi tim Anda. Hubungi AWS Support melalui Pusat AWS Support jika Anda memiliki pertanyaan seputar AWS.

AWS juga membagikan pola dan praktik terbaik yang telah kami pelajari melalui operasi AWS di Amazon Builders' Library. Beragam jenis informasi berguna lainnya dapat diakses melalui Blog AWS dan Podcast AWS Resmi.

Antipola umum:

- Anda menggunakan basis data relasional untuk mengelola data seri waktu dan non-relasional. Terdapat opsi-opsi basis data yang dioptimalkan untuk mendukung tipe data yang Anda gunakan tetapi Anda tidak menyadari manfaatnya karena Anda belum mengevaluasi kompromi antarsolusi.
- Investor Anda meminta Anda mendemonstrasikan kepatuhan terhadap Standar Keamanan Data Industri Kartu Pembayaran (PCI DSS). Anda tidak mempertimbangkan kompromi antara memenuhi permintaan mereka dan melanjutkan upaya pengembangan Anda saat ini. Alih-alih, Anda melanjutkan upaya pengembangan tanpa menunjukkan kepatuhan. Investor Anda menghentikan dukungan untuk perusahaan Anda karena mengkhawatirkan keamanan platform Anda serta investasi mereka.

Manfaat menjalankan praktik terbaik ini: Memahami implikasi dan konsekuensi pilihan yang Anda ambil dapat membantu Anda membuat prioritas opsi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

- Evaluasi kompromi: Evaluasi dampak kompromi antarkepentingan yang bertentangan untuk membantu mengambil keputusan yang matang saat menentukan ke mana upaya perlu difokuskan. Misalnya, mempercepat waktu masuk pasar untuk fitur baru dapat diprioritaskan daripada optimalisasi biaya.

- AWS dapat membantu mengedukasi tim Anda tentang AWS dan layanannya untuk meningkatkan pemahaman mereka tentang bagaimana pilihan mereka dapat menimbulkan dampak pada beban kerja. Anda harus menggunakan sumber daya yang disediakan oleh AWS Support (Pusat Pengetahuan AWS, Forum DiskusiAWS, dan Pusat AWS Support) serta Dokumentasi AWS untuk mengedukasi tim Anda. Hubungi AWS Support melalui Pusat AWS Support jika Anda memiliki pertanyaan seputar AWS.
- AWS juga membagikan pola dan praktik terbaik yang telah kami pelajari melalui operasi AWS di Amazon Builders' Library. Beragam jenis informasi berguna lainnya dapat diakses melalui Blog AWS dan Podcast AWS Resmi.

Sumber daya

Dokumen terkait:

- Blog AWS
- Kepatuhan AWS Cloud
- Forum Diskusi AWS
- Dokumentasi AWS
- Pusat Pengetahuan AWS
- AWS Support
- Pusat AWS Support
- Amazon Builders' Library
- Podcast AWS Resmi

OPS01-BP07 Kelola manfaat dan risiko

Kelola manfaat dan risiko untuk mengambil keputusan yang bijaksana ketika menentukan di mana akan memfokuskan upaya. Contohnya, mungkin akan bermanfaat untuk melakukan deploy beban kerja dengan masalah yang tak terselesaikan sehingga fitur baru yang signifikan dapat dibuat tersedia bagi pelanggan. Risiko terkait mungkin dapat dimitigasi, atau membiarkan risiko tetap ada mungkin menjadi tidak dapat diterima, jika demikian, Anda akan mengambil tindakan untuk mengatasi risiko tersebut.

Anda mungkin mendapatkan bahwa Anda ingin menekankan subset kecil prioritas pada titik waktu tertentu. Gunakan pendekatan yang seimbang dalam jangka panjang untuk memastikan

pengembangan kemampuan yang diperlukan dan pengelolaan risiko. Perbarui prioritas Anda sesuai perubahan kebutuhan

Antipola umum:

- Anda telah memutuskan untuk menyertakan pustaka yang melakukan semua yang Anda perlukan yang ditemukan salah satu developer Anda di internet. Anda belum mengevaluasi risiko adopsi pustaka ini dari sumber tak dikenal dan Anda tidak tahu jika pustaka memiliki kelemahan atau kode jahat.
- Anda telah memutuskan untuk mengembangkan dan melakukan deploy fitur baru dan bukannya memperbaiki masalah yang ada. Anda belum mengevaluasi risiko meninggalkan masalah sampai fitur dilakukan deploy dan Anda tidak tahu apa saja dampaknya pada pelanggan Anda.
- Anda telah memutuskan untuk tidak melakukan deploy fitur yang sering diminta oleh pelanggan karena masalah yang tidak jelas dari tim kepatuhan Anda.

Manfaat menerapkan praktik terbaik ini: Mengidentifikasi manfaat yang tersedia dari pilihan Anda, dan menyadari risiko terhadap organisasi Anda, memungkinkan Anda untuk mengambil keputusan yang bijaksana.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

- Kelola manfaat dan risiko: Seimbangkan manfaat dari keputusan terhadap risiko yang terlibat.
  - Identifikasi manfaat: Identifikasi manfaat berdasarkan tujuan, kebutuhan, dan prioritas bisnis. Contohnya antara lain waktu masuk pasar, keamanan, keandalan, performa, dan biaya.
  - Identifikasi risiko: Identifikasi risiko berdasarkan tujuan, kebutuhan, dan prioritas bisnis. Contohnya antara lain waktu masuk pasar, keamanan, keandalan, performa, dan biaya.
  - Evaluasi manfaat dibandingkan risiko dan ambil keputusan yang bijaksana: Tentukan dampak manfaat dan risiko berdasarkan tujuan, kebutuhan, dan prioritas pemangku kepentingan utama Anda, termasuk bagian bisnis, pengembangan, dan operasi. Evaluasi nilai manfaat dibandingkan dengan probabilitas terwujudnya risiko dan kerugian dampaknya. Contohnya, menekankan kecepatan masuk pasar dan bukannya keandalan dapat memberikan keunggulan yang bersaing. Tetapi, ini dapat mengakibatkan berkurangnya waktu aktif jika ada masalah keandalan.

Model operasi

Tim Anda harus memahami bagian mereka dalam mencapai hasil bisnis. Tim harus memahami peran mereka dalam kesuksesan tim lain, peran tim lain dalam kesuksesan mereka, dan memiliki tujuan bersama. Memahami tanggung jawab, kepemilikan, bagaimana keputusan diambil, dan siapa yang memiliki otoritas untuk mengambil keputusan dapat membantu memfokuskan upaya dan memaksimalkan manfaat dari tim Anda.

Kebutuhan sebuah tim akan dibentuk oleh industri, organisasi, formasi tim, dan karakteristik beban kerja mereka. Satu model operasi saja tentunya tidak cukup untuk mendukung semua tim dan beban kerja mereka.

Jumlah model operasi yang ada dalam organisasi cenderung naik dengan jumlah tim pengembangan. Anda mungkin perlu menggunakan kombinasi model operasi.

Dengan mengadopsi standar dan menggunakan layanan, operasi menjadi sederhana dan beban dukungan dalam model operasi Anda dapat dibatasi. Manfaat upaya pengembangan pada standar bersama meningkat dengan makin banyaknya jumlah tim yang mengadopsi standar dan yang akan mengadopsi fitur baru.

Tersedianya mekanisme sangat penting untuk meminta penambahan, perubahan, dan pengecualian terhadap standar guna mendukung aktivitas tim. Tanpa opsi ini, standar akan menjadi penghambat inovasi. Sebaiknya setujui permintaan apabila memungkinkan dan dianggap tepat setelah dilakukan evaluasi manfaat dan risiko.

Seperangkat tanggung jawab yang ditentukan dengan baik akan mengurangi frekuensi upaya yang redundan dan bermasalah. Hasil bisnis akan lebih mudah dicapai saat ada kesesuaian dan hubungan yang kuat antara bisnis, pengembangan, dan tim operasi.

Representasi model operasi 2 per 2

Representasi model operasi 2 per 2 merupakan ilustrasi untuk membantu Anda memahami hubungan antartim dalam lingkungan. Diagram ini berfokus pada tindakan dan pelaksananya serta hubungan antartim, selain itu kami juga akan mendiskusikan tata kelola dan pembuatan keputusan yang berkaitan dengan contoh ini.

Tim kami dapat memiliki tanggung jawab dalam berbagai model bergantung pada beban kerja yang mereka dukung. Anda mungkin ingin mencoba area disiplin yang lebih terspesialisasi daripada

yang dijelaskan. Variasi model ini dapat menjadi tak terbatas seiring Anda memisahkan atau menggabungkan aktivitas, atau menyebarkan tim dan memberikan detail spesifik.

Anda dapat mengidentifikasi bahwa Anda memiliki kemampuan yang tumpang tindih atau tidak diketahui untuk seluruh tim, yang dapat memberikan keuntungan tambahan, atau mendorong efisiensi. Anda juga dapat mengidentifikasi kebutuhan yang belum terpenuhi dalam organisasi dan dapat merencanakan penanganannya.

Saat mengevaluasi perubahan organisasi, pelajari kompensasi antara model, posisi tim individu Anda dalam model (sekarang dan setelah perubahan), bagaimana hubungan dan kemampuan tim akan berubah, dan apakah manfaat sesuai dengan dampak pada organisasi.

Anda bisa sukses menggunakan masing-masing dari empat model operasi berikut. Beberapa model lebih sesuai untuk kasus penggunaan tertentu atau pada titik tertentu dalam pengembangan. Beberapa model dapat memberikan manfaat lebih dari model yang digunakan dalam lingkungan.

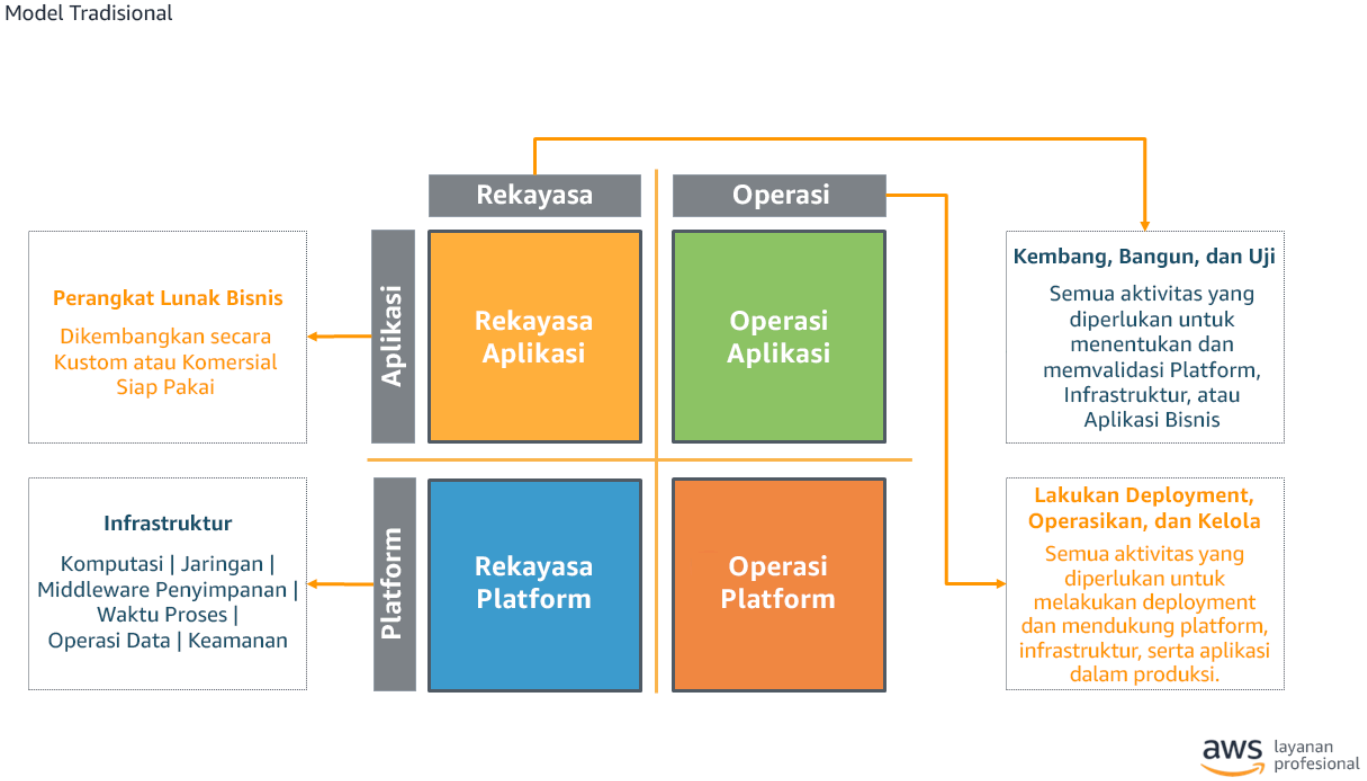
Topik

- Model operasi yang terpisah sepenuhnya
- Rekayasa dan Operasi Aplikasi Terpisah (AEO) serta Rekayasa dan Operasi Infrastruktur (IEO) dengan tata kelola terpusat
- AEO dan IEO Terpisah dengan tata kelola terpusat serta penyedia layanan
- AEO dan IEO Terpisah dengan tata kelola terpusat dan partner konsultan penyedia layanan internal
- AEO dan IEO Terpisah dengan tata kelola terdesentralisasi

Model operasi yang terpisah sepenuhnya

Dalam diagram berikut, aplikasi dan infrastruktur berada pada sumbu vertikal. Aplikasi merujuk pada beban kerja yang digunakan dalam hasil bisnis dan dapat dikembangkan secara kustom atau perangkat lunak yang dibeli. Infrastruktur merujuk pada infrastruktur fisik serta virtual dan perangkat lunak lainnya yang mendukung beban kerja tersebut.

Rekayasa dan Operasi berada pada sumbu horizontal. Rekayasa merujuk pada pengembangan, pembangunan, dan pengujian aplikasi serta infrastruktur. Operasi adalah deployment, pembaruan, dan dukungan yang masih berlangsung untuk aplikasi dan infrastruktur.



Dalam berbagai organisasi, model “terpisah sepenuhnya” ini digunakan. Aktivitas dalam setiap kuadran dilakukan oleh tim yang berbeda. Pekerjaan diteruskan antartim melalui mekanisme seperti permintaan pekerjaan, antrean pekerjaan, tiket, atau dengan menggunakan sistem manajemen layanan IT (ITSM).

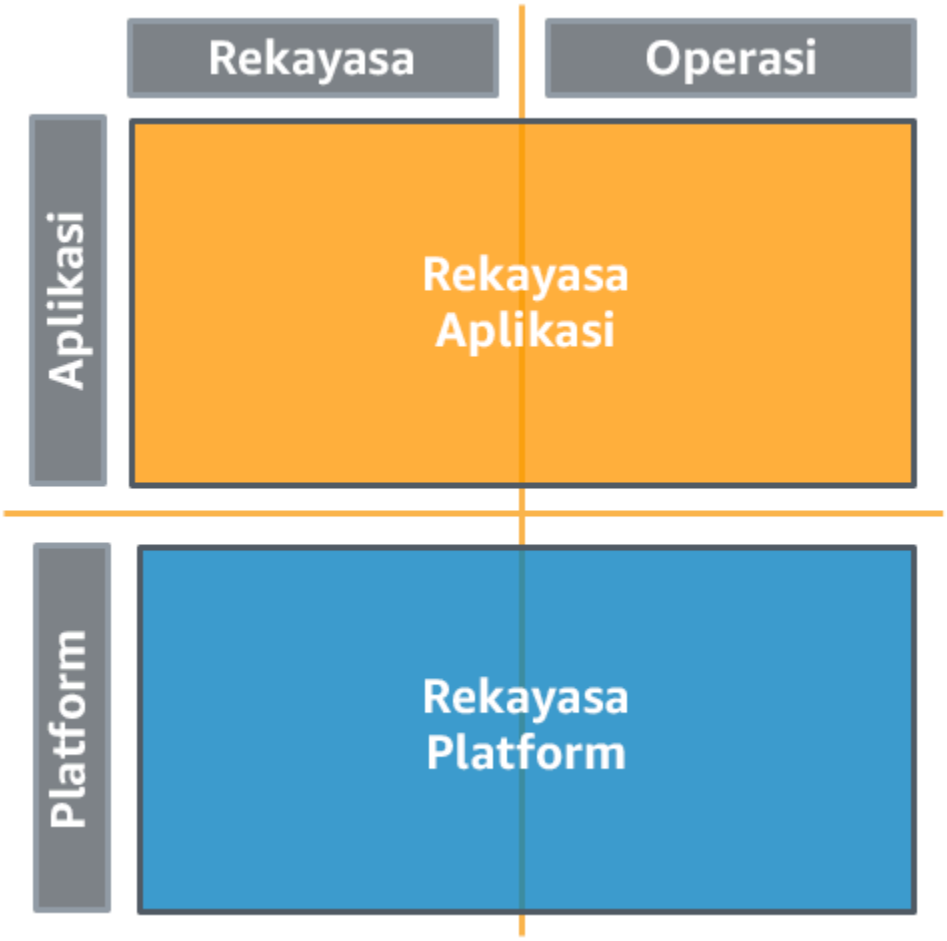
Transisi tugas kepada atau antartim dapat meningkatkan kompleksitas, serta menyebabkan bottleneck dan penundaan. Permintaan mungkin ditunda hingga menjadi prioritas. Kecacatan yang terlambat diidentifikasi dapat memerlukan banyak upaya untuk pengerjaan ulang dan mungkin harus melewati tim yang sama beserta fungsinya satu kali lagi. Jika ada insiden yang memerlukan tindakan tim rekayasawan, respons mereka akan ditunda oleh aktivitas transfer.

Ada risiko ketidaksesuaian yang lebih tinggi saat bisnis, pengembangan, dan tim operasi dikelola di sekitar aktivitas atau fungsi yang sedang dijalankan. Hal ini dapat membuat tim berfokus pada tanggung jawab spesifik mereka, bukan pada mencapai hasil bisnis. Tim dapat terspesialisasi secara sempit, terisolasi secara fisik, atau terisolasi secara logis, menghambat komunikasi dan kolaborasi.

Rekayasa dan Operasi Aplikasi Terpisah (AEO) serta Rekayasa dan Operasi Infrastruktur (IEO) dengan tata kelola terpusat

Model “AEO dan IEO Terpisah” mengikuti metodologi “you build it you run it” (Anda yang membangunnya, Anda pula yang menjalankannya).

Rekayasawan dan developer Anda melakukan rekayasa dan operasi beban kerja mereka. Dengan cara yang sama, rekayasawan infrastruktur melakukan rekayasa dan operasi pada platform yang digunakan untuk mendukung tim aplikasi.



Untuk contoh ini, kami akan membuat tata kelola menjadi terpusat. Standar didistribusikan, disediakan, atau dibagikan kepada tim aplikasi.

Anda harus menggunakan alat atau layanan yang memungkinkan Anda untuk mengelola lingkungan di seluruh akun Anda secara terpusat, seperti AWS Organizations. Layanan seperti AWS Control Tower memperluas kemampuan manajemen ini dengan memudahkan Anda menentukan cetak biru

(yang mendukung model operasi) untuk penyiapan akun, menerapkan tata kelola berkelanjutan menggunakan AWS Organizations, dan mengotomatiskan penyediaan akun baru.

“You build it you run it” bukan berarti tim aplikasi bertanggung jawab atas full stack, tool chain, dan platform.

Tim rekayasawan platform menyediakan set layanan yang terstandardisasi (misalnya, alat pengembangan, alat pemantauan, alat pencadangan dan pemulihan, serta jaringan) kepada tim aplikasi. Tim platform juga dapat menyediakan ke layanan penyedia cloud yang disetujui, konfigurasi tertentu, atau keduanya, kepada tim aplikasi.

Mekanisme yang memberikan kemampuan untuk melakukan deployment konfigurasi dan layanan yang disetujui, misalnya Service Catalog, yang dapat membantu membatasi penundaan terkait permintaan pemenuhan sekaligus menerapkan tata kelola.

Tim platform mengaktifkan visibilitas full stack agar tim aplikasi dapat membedakan antara masalah dengan komponen aplikasi dan layanan serta komponen infrastruktur yang digunakan aplikasi. Tim platform juga dapat menyediakan bantuan untuk mengonfigurasi layanan ini dan panduan tentang cara meningkatkan operasi tim aplikasi.

Seperti yang didiskusikan sebelumnya, tersedianya mekanisme sangat penting untuk meminta penambahan, perubahan, dan pengecualian terhadap standar guna mendukung aktivitas tim dan inovasi aplikasi.

Model AEO IEO Terpisah memberikan loop umpan balik kepada tim aplikasi. Operasi harian beban kerja meningkatkan kontak dengan pelanggan, baik melalui interaksi langsung atau tidak langsung lewat permintaan fitur dan dukungan. Visibilitas yang lebih tinggi ini memungkinkan tim aplikasi untuk menangani masalah dengan cepat. Keterlibatan yang mendalam dan hubungan yang lebih dekat memberikan wawasan atas kebutuhan pelanggan dan memungkinkan inovasi yang lebih cepat.

Semua ini juga berlaku untuk tim platform yang mendukung tim aplikasi.

Standar yang diadopsi mungkin belum disetujui untuk penggunaan, mengurangi jumlah peninjauan yang diperlukan untuk memasuki produksi. Menggunakan standar yang didukung dan diuji serta disediakan oleh tim platform dapat mengurangi frekuensi masalah dengan layanan tersebut. Pengadopsian standar memungkinkan tim aplikasi untuk fokus menjadikan beban kerja mereka kompetitif.

AEO dan IEO Terpisah dengan tata kelola terpusat serta penyedia layanan

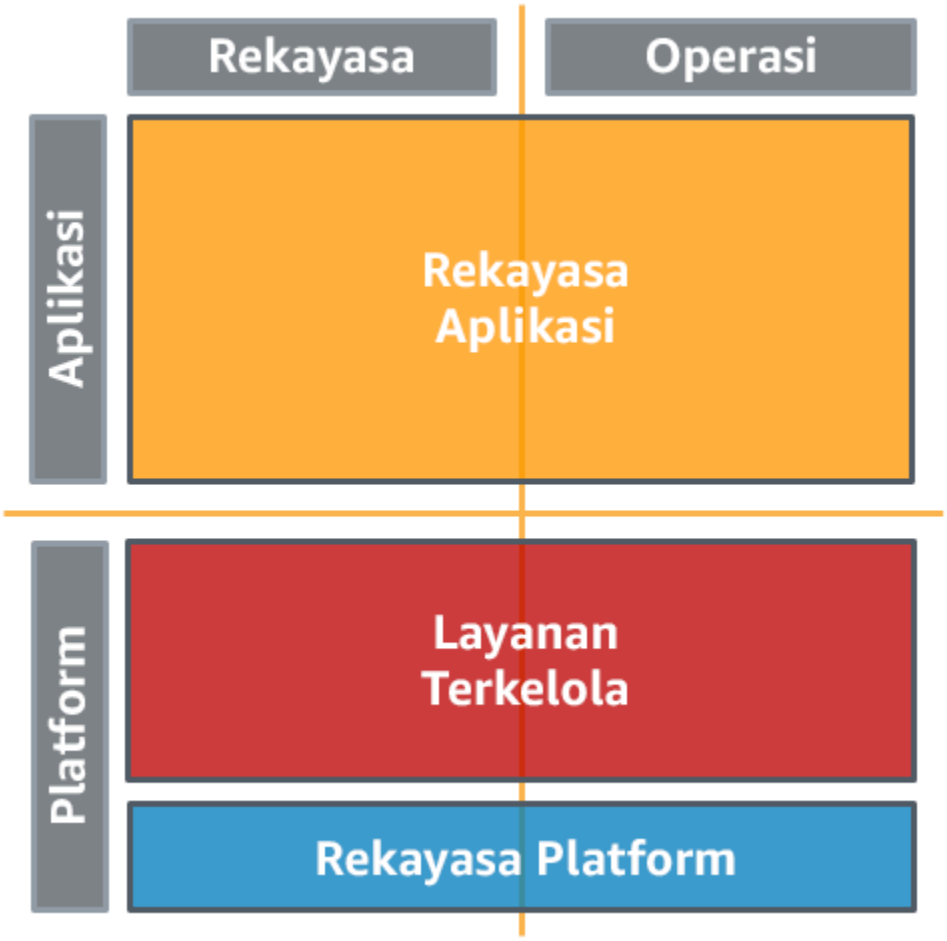
Model “AEO dan IEO Terpisah” mengikuti metodologi “you build it you run it” (Anda yang membangunnya, Anda pula yang menjalankannya).

Rekayasawan dan developer Anda melakukan rekayasa dan operasi beban kerja mereka.

Organisasi Anda mungkin belum memiliki keterampilan, atau anggota tim, untuk mendukung rekayasa platform khusus dan tim operasi, atau Anda mungkin tidak ingin meluangkan waktu dan tenaga untuk melakukannya.

Jika tidak, Anda mungkin ingin memiliki tim platform yang berfokus pada pengembangan kemampuan yang akan membedakan bisnis Anda, tetapi Anda ingin membongkar operasi harian yang tidak mendatangkan keuntungan kompetitif kepada pengalih daya.

Penyedia Layanan Terkelola seperti AWS Managed Services, Partner AWS Managed Services, atau Penyedia Layanan Terkelola di Jaringan Partner AWS, menyediakan keahlian dalam mengimplementasikan lingkungan cloud, dan mendukung persyaratan keamanan dan kepatuhan serta tujuan bisnis.



Untuk variasi ini, kami akan menetapkan agar tata kelola dipusatkan dan dikelola oleh tim platform, dengan pembuatan akun dan kebijakan yang dikelola melalui AWS Organizations dan AWS Control Tower.

Dalam model ini, mekanisme perlu diubah agar dapat dijalankan dengan penyedia layanan tersebut. Hal ini tidak mengatasi bottleneck dan penundaan yang disebabkan oleh transisi tugas antartim, termasuk penyedia layanan, atau kemungkinan pengerjaan ulang terkait dengan keterlambatan identifikasi kecacatan.

Anda mendapatkan manfaat dari standar penyedia, praktik terbaik, proses, dan keahlian. Anda juga mendapatkan manfaat dari pengembangan berkelanjutan penawaran layanan.

Menambahkan Layanan Terkelola ke model operasi Anda dapat menghemat waktu dan sumber daya, serta memungkinkan Anda menjaga tim internal Anda untuk tetap belajar dan fokus pada hasil strategis yang akan membedakan bisnis Anda, dan bukan mengembangkan keterampilan dan kemampuan baru.

AEO dan IEO Terpisah dengan tata kelola terpusat dan partner konsultan penyedia layanan internal

Model “AEO dan IEO Terpisah” ini berupaya menetapkan metodologi “you build it you run it”.

Anda ingin tim aplikasi melakukan aktivitas rekayasa dan operasi untuk beban kerja, dan mengadopsi kultur yang lebih mirip DevOps.

Tim aplikasi mungkin dalam proses migrasi, mengadopsi cloud, atau memodernisasi beban kerja, dan tidak memiliki kemampuan yang ada untuk mendukung cloud serta operasi cloud secara memadai. Tim aplikasi yang belum cukup familier dan kemampuannya belum memadai dapat menjadi penghambat usaha Anda.

Untuk menangani masalah ini, Anda menetapkan tim Pusat Pemberdayaan Cloud (CCoE) yang menyediakan forum untuk mengajukan pertanyaan, kebutuhan diskusi, dan mengidentifikasi solusi. Bergantung pada kebutuhan organisasi, CCoE dapat berisi tim ahli tim virtual khusus dengan peserta yang dipilih dari seluruh organisasi. CCoE memungkinkan transformasi cloud untuk tim, menetapkan tata kelola cloud terpusat, dan menentukan akun serta standar manajemen organisasi. Mereka juga mengidentifikasi pola dan arsitektur referensi yang berhasil untuk penggunaan perusahaan.

Kami merujuk CCoE sebagai Pusat Pemberdayaan Cloud, bukan istilah yang lebih umum yaitu Pusat Keunggulan Cloud, untuk memberikan penekanan pada pemberdayaan keberhasilan tim yang didukung dan pencapaian hasil bisnis.

Tim rekayasa platform membangun kemampuan platform bersama inti berdasarkan standar tersebut agar diadopsi oleh tim aplikasi. Mereka mengodekan arsitektur referensi dan pola yang disediakan untuk tim aplikasi melalui mekanisme layanan mandiri. Dengan menggunakan layanan seperti AWS Service Catalog, tim aplikasi dapat melakukan deployment arsitektur referensi, pola, layanan, dan konfigurasi yang disetujui, serta secara default mematuhi tata kelola terpusat dan standar keamanan.

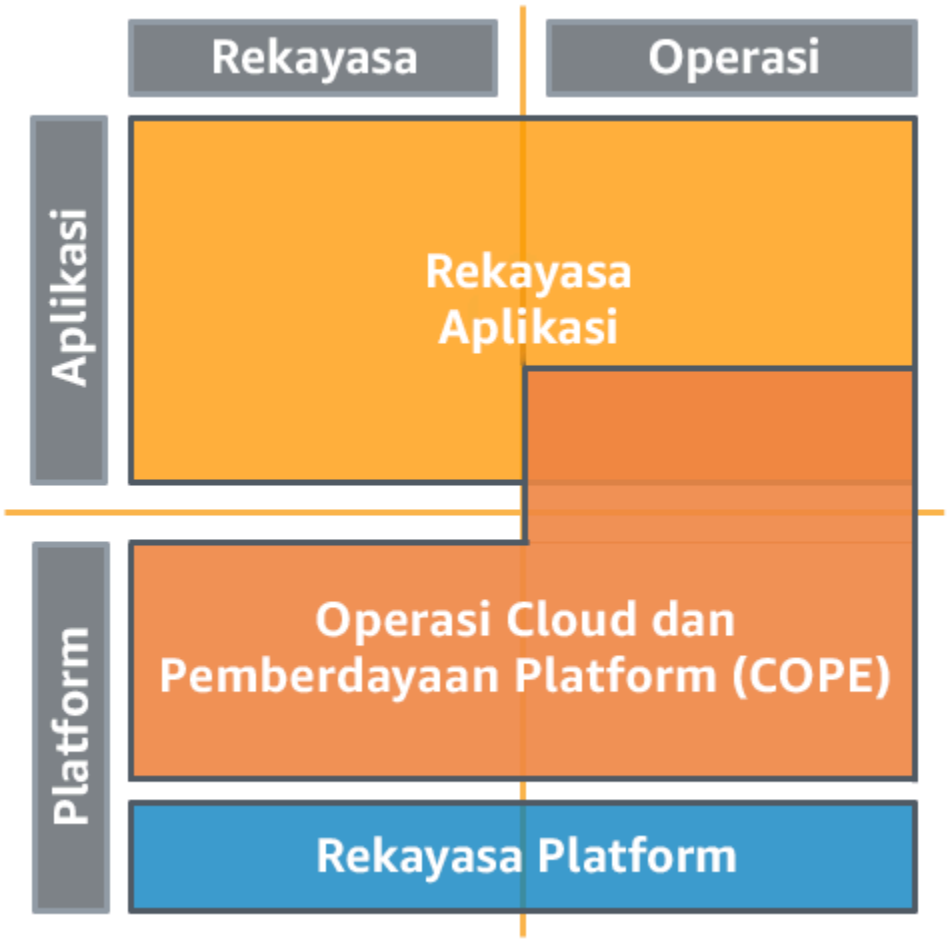
Tim rekayasawan platform juga menyediakan set layanan yang terstandardisasi (misalnya, alat pengembangan, alat pemantauan, alat pencadangan dan pemulihan, serta jaringan) kepada tim aplikasi.

Organisasi memiliki “Partner Konsultasi dan MSP Internal” yang mengelola dan mendukung layanan terstandardisasi serta memberikan bantuan kepada tim aplikasi untuk menetapkan kehadiran cloud berdasarkan arsitektur referensi dan pola. Tim “Pemberdayaan Platform dan Operasi Cloud (COPE)” bekerja dengan tim aplikasi untuk membantu mereka menetapkan operasi dasar dengan tim aplikasi secara progresif, dengan mengambil lebih banyak tanggung jawab untuk sistem dan sumber daya

mereka dari waktu ke waktu. Tim COPE mendorong peningkatan berkelanjutan bersama dengan tim CCoE dan Rekayasa Platform, serta bertindak sebagai pendukung untuk tim aplikasi.

Tim aplikasi mendapatkan bantuan dalam menyiapkan lingkungan, pelipir CICD, manajemen perubahan, observabilitas dan pemantauan, serta menetapkan insiden dan proses manajemen peristiwa dengan tim COPE yang terintegrasi dengan perusahaan sebagaimana yang diperlukan. Tim COPE bekerja sama dengan tim aplikasi dalam menjalankan aktivitas operasi ini, menghapus keterlibatan tim COPE dari waktu ke waktu saat tim aplikasi mengambil alih kepemilikan.

Tim aplikasi memperoleh manfaat dari keterampilan tim COPE dan pelajaran yang dipetik oleh organisasi. Mereka dilindungi oleh pagar pembatas yang didirikan melalui tata kelola terpusat. Tim aplikasi melakukan pembangunan berdasarkan kesuksesan yang diakui dan mendapatkan manfaat dari pengembangan berkelanjutan dari standar organisasi yang telah diadopsi. Mereka mendapatkan wawasan ke operasi beban kerja melalui proses penetapan observabilitas dan pemantauan, dan dapat memahami dampak perubahan yang dibuat untuk beban kerja dengan lebih baik.



Representasi model operasi 2 per 2

Tim COPE mempertahankan akses yang diperlukan untuk mendukung aktivitas operasi, memberikan tampilan operasi perusahaan yang mencakup tim aplikasi, dan untuk memberikan dukungan manajemen insiden yang sangat penting. Tim COPE mempertahankan tanggung jawab untuk aktivitas yang dianggap sebagai pekerjaan berat yang tidak mendatangkan keuntungan kompetitif, yang mereka penuhi melalui solusi standar yang dapat didukung dalam skala besar. Mereka juga terus mengelola aktivitas operasi terprogram dan otomatis yang dipahami dengan baik untuk tim aplikasi sehingga mereka dapat fokus menjadikan aplikasi mereka kompetitif.

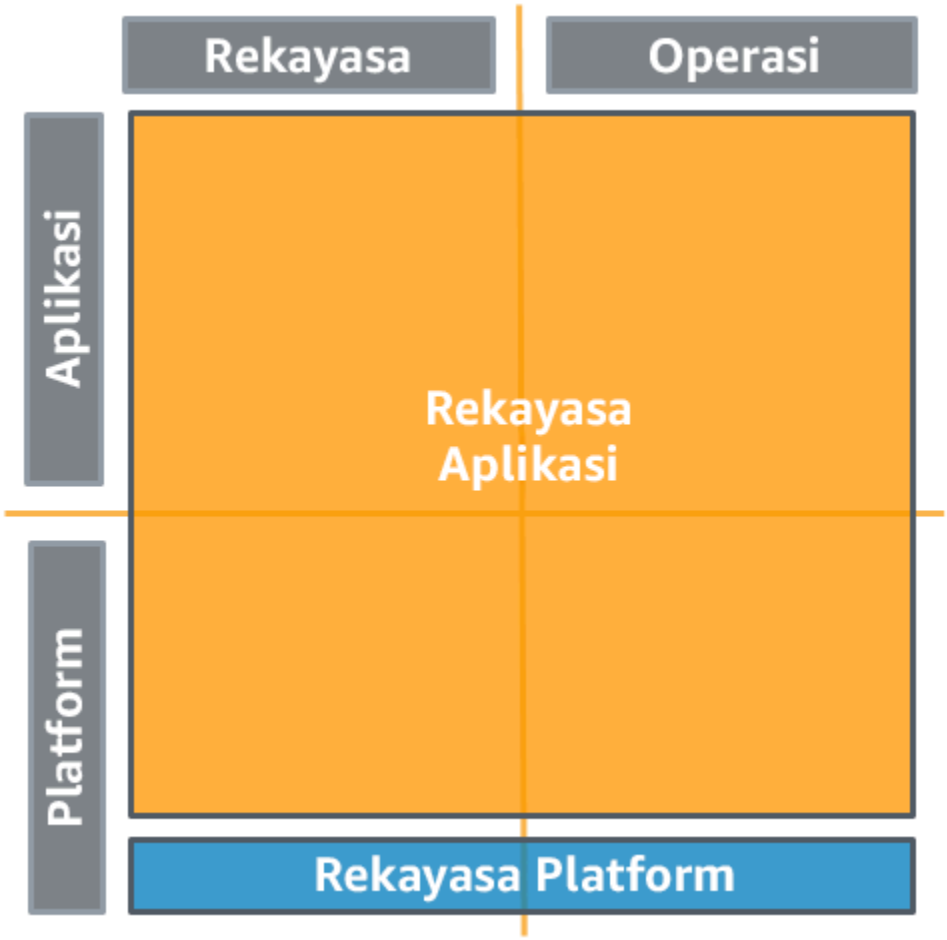
Anda mendapatkan keuntungan dari standar, praktik terbaik, proses, dan keahlian organisasi yang diperoleh dari keberhasilan tim Anda. Anda menetapkan mekanisme untuk mereplikasi pola yang berhasil ini untuk tim baru yang melakukan adopsi atau modernisasi di cloud. Model ini menekankan pada kemampuan tim COPE untuk membantu penetapan tim aplikasi, serta mentransisikan pengetahuan dan artefak. Ini mengurangi beban operasional tim aplikasi dengan risiko bahwa tim aplikasi akan gagal menjadi independen secara umum. Ini membangun hubungan antara CCoE, COPE, dan tim aplikasi dalam membuat loop umpan balik guna mendukung perkembangan dan inovasi lebih lanjut.

Menetapkan tim CCoE dan COPE sekaligus menentukan standar di seluruh organisasi, dapat memfasilitasi adopsi cloud dan mendukung upaya modernisasi. Dengan memberikan dukungan tambahan dari tim COPE yang bertindak sebagai konsultan dan partner untuk tim aplikasi, Anda dapat menghilangkan hambatan yang memperlambat adopsi tim aplikasi dari kemampuan cloud yang bermanfaat.

AEO dan IEO Terpisah dengan tata kelola terdesentralisasi

Model “AEO dan IEO Terpisah” mengikuti metodologi “you build it you run it” (Anda yang membangunnya, Anda pula yang menjalankannya).

Rekayasawan dan developer Anda melakukan rekayasa dan operasi beban kerja mereka. Dengan cara yang sama, rekayasawan infrastruktur melakukan rekayasa dan operasi pada platform yang digunakan untuk mendukung tim aplikasi.



Untuk contoh ini, kami akan membuat tata kelola menjadi terdesentralisasi.

Standar masih didistribusikan, disediakan, atau dibagikan kepada tim aplikasi oleh tim platform, tetapi tim aplikasi bebas untuk merekayasa dan mengoperasikan kemampuan platform baru guna mendukung beban kerja.

Dalam model ini, tim aplikasi mengalami lebih sedikit kendala, tetapi hal itu disertai dengan peningkatan tanggung jawab yang signifikan. Keterampilan tambahan, dan anggota tim potensial, harus ada untuk mendukung kemampuan platform tambahan. Risiko pengerjaan ulang yang signifikan akan meningkat jika keahlian tidak memadai dan kecacatan tidak teridentifikasi lebih awal.

Anda harus menerapkan kebijakan yang tidak didelegasikan secara khusus ke tim aplikasi. Gunakan alat atau layanan yang memungkinkan Anda untuk mengelola lingkungan di seluruh akun Anda secara terpusat, seperti AWS Organizations. Layanan seperti AWS Control Tower memperluas kemampuan manajemen ini dengan memudahkan Anda menentukan cetak biru (yang mendukung

model operasi) untuk penyiapan akun, menerapkan tata kelola berkelanjutan menggunakan AWS Organizations, dan mengotomatiskan penyediaan akun baru.

Tim aplikasi akan diuntungkan dengan memiliki mekanisme untuk meminta penambahan dan perubahan standar. Mereka dapat memberikan kontribusi berupa standar baru yang dapat memberikan manfaat bagi tim aplikasi lain. Tim platform dapat memutuskan bahwa memberikan dukungan langsung untuk kemampuan tambahan ini merupakan dukungan yang efektif untuk hasil bisnis.

Model ini meminimalkan kendala inovasi dengan keterampilan yang signifikan dan persyaratan anggota tim. Ini mengatasi banyak hambatan dan penundaan yang disebabkan oleh transisi tugas antartim sambil tetap mempromosikan pengembangan hubungan yang efektif antara tim dan pelanggan.

Hubungan dan kepemilikan

Model operasi menentukan hubungan antartim dan mendukung kepemilikan serta tanggung jawab yang dapat diidentifikasi.

Praktik terbaik

- OPS02-BP01 Sumber daya memiliki pemilik teridentifikasi
- OPS02-BP02 Proses dan Prosedur memiliki pemilik teridentifikasi
- OPS02-BP03 Aktivitas operasi memiliki pemilik teridentifikasi yang bertanggung jawab atas kinerjanya
- OPS02-BP04 Anggota tim tahu tanggung jawab mereka
- OPS02-BP05 Mekanisme tersedia untuk mengidentifikasi tanggung jawab dan kepemilikan
- OPS02-BP06 Mekanisme tersedia untuk meminta penambahan, perubahan, dan pengecualian
- OPS02-BP07 Tanggung jawab antara tim telah dinegosiasi atau ditetapkan sebelumnya

OPS02-BP01 Sumber daya memiliki pemilik teridentifikasi

Sumber daya untuk beban kerja Anda harus memiliki pemilik yang teridentifikasi untuk pengontrolan perubahan, penyelesaian masalah, dan fungsi-fungsi lainnya. Pemilik ditetapkan untuk beban kerja, akun, infrastruktur, platform, dan aplikasi. Kepemilikan dicatat menggunakan alat seperti daftar sentral atau metadata yang dilampirkan ke sumber daya. Nilai bisnis komponen menginformasikan proses dan prosedur yang diterapkan kepadanya.

Hasil yang diinginkan:

- Sumber daya telah mengidentifikasi pemilik menggunakan metadata atau daftar sentral.
- Anggota tim dapat mengidentifikasi siapa pemilik sumber daya.
- Akun memiliki satu pemilik apabila mungkin.

Antipola umum:

- Kontak alternatif untuk Akun AWS Anda tidak diisi.
- Sumber daya tidak memiliki tag yang mengidentifikasi tim mana yang memilikinya.
- Anda memiliki antrean ITSM tanpa pemetaan email.
- Dua tim sama-sama merupakan pemilik bagian penting dari infrastruktur.

Manfaat menjalankan praktik terbaik ini:

- Kontrol perubahan untuk sumber daya mudah dilakukan dengan ditetapkannya kepemilikan.
- Anda dapat melibatkan pemilik yang benar ketika menyelesaikan masalah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Tentukan pentingnya kepemilikan untuk kasus penggunaan sumber daya di lingkungan Anda. Kepemilikan dapat berarti siapa yang mengawasi perubahan pada sumber daya, mendukung sumber daya selama penyelesaian masalah, atau siapa yang bertanggung jawab secara finansial. Sebutkan dan catat pemilik untuk sumber daya, termasuk nama, informasi kontak, organisasi, serta tim.

Contoh pelanggan

AnyCompany Retail menetapkan kepemilikan sebagai tim atau individu yang memiliki perubahan dan dukungan untuk sumber daya. Mereka memanfaatkan AWS Organizations untuk mengelola Akun AWS mereka. Kontak akun alternatif dikonfigurasi menggunakan kotak masuk grup. Setiap antrean ITSM dipetakan ke alias email. Tag mengidentifikasi siapa yang memiliki sumber daya AWS. Untuk infrastruktur dan platform lainnya, mereka memiliki halaman wiki yang mengidentifikasi kepemilikan dan informasi kontak.

Langkah implementasi

1. Mulai dengan menetapkan kepemilikan untuk organisasi Anda. Kepemilikan dapat menyiratkan siapa yang memiliki risiko untuk sumber daya, siapa yang memiliki perubahan pada sumber daya, atau siapa yang mendukung sumber daya ketika menyelesaikan masalah. Kepemilikan juga dapat menyiratkan kepemilikan sumber daya secara finansial atau administratif.
2. Gunakan AWS Organizations untuk mengelola akun. Anda dapat mengelola kontak alternatif untuk akun Anda secara terpusat.
   a. Dengan menggunakan alamat email dan nomor telepon milik perusahaan untuk informasi kontak, Anda dapat tetap dapat mengaksesnya meskipun orang yang memilikinya tidak lagi bekerja di organisasi Anda. Misalnya, buat daftar distribusi email terpisah untuk penagihan, operasional, dan keamanan lalu konfigurasikan ketiganya sebagai kontak Penagihan, Keamanan, dan Operasional di setiap Akun AWS yang aktif. Banyak orang akan menerima notifikasi AWS dan dapat merespons, meskipun ada yang sedang berlibur, berganti posisi, atau meninggalkan perusahaan.
   b. Jika akun tidak dikelola oleh AWS Organizations, kontak akun alternatif dapat membantu AWS menghubungi personel yang tepat jika diperlukan. Konfigurasikan kontak alternatif akun sehingga menunjuk ke grup dan bukannya individu.
3. Gunakan tag untuk mengidentifikasi pemilik untuk sumber daya AWS. Anda dapat menentukan pemilik maupun informasi kontak mereka dalam tag terpisah.
   a. Anda dapat menggunakan aturan AWS Config untuk menegakkan bahwa sumber daya memiliki tag kepemilikan yang diperlukan.
   b. Untuk panduan mendalam tentang cara membangun strategi pemberian tag untuk organisasi Anda, lihat AWSlaporan resmi Praktik Terbaik Pemberian Tag.
4. Untuk sumber daya, platform, dan infrastruktur lainnya, buat dokumentasi yang mengidentifikasi kepemilikan. Dokumentasi ini harus dapat diakses oleh semua anggota tim.

Tingkat upaya untuk rencana implementasi: Rendah. Manfaatkan informasi kontak akun dan tag untuk menetapkan kepemilikan sumber daya AWS. Untuk sumber daya lainnya, Anda dapat menggunakan sesuatu yang sederhana seperti tabel di wiki hingga catatan kepemilikan dan informasi kontak, atau gunakan alat ITSM untuk memetakan kepemilikan.

Sumber daya

Praktik terbaik terkait:

- OPS02-BP02 Proses dan Prosedur memiliki pemilik teridentifikasi - Proses dan prosedur untuk mendukung sumber daya bergantung pada kepemilikan sumber daya.

- OPS02-BP04 Anggota tim tahu tanggung jawab mereka - Anggota tim harus memahami sumber daya apa yang mereka miliki.
- OPS02-BP05 Mekanisme tersedia untuk mengidentifikasi tanggung jawab dan kepemilikan - Kepemilikan harus dapat ditemukan menggunakan mekanisme seperti tag atau kontak akun.

Dokumen terkait:

- Manajemen Akun AWS - Memperbarui informasi kontak
- Aturan AWS Config - tag yang diperlukan
- AWS Organizations - Memperbarui kontak alternatif dalam organisasi Anda
- Laporan resmi Praktik Terbaik Pemberian Tag AWS

Contoh terkait:

- Aturan AWS Config - Amazon EC2 dengan tag yang diperlukan dan nilai yang valid

Layanan terkait:

- AWS Config
- AWS Organizations

OPS02-BP02 Proses dan Prosedur memiliki pemilik teridentifikasi

Pahami siapa pemegang kepemilikan atas definisi dari masing-masing proses dan prosedur, alasan prosedur dan proses tertentu digunakan, serta alasan adanya kepemilikan tersebut. Dengan memahami alasan untuk menggunakan proses dan prosedur tertentu, identifikasi peluang pengembangan yang dapat dilakukan.

Manfaat menerapkan praktik terbaik ini: Dengan memahami kepemilikan, Anda dapat mengidentifikasi siapa yang dapat menyetujui pengembangan, mengimplementasikan pengembangan tersebut, atau melakukan keduanya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Proses dan prosedur memiliki pemilik teridentifikasi yang bertanggung jawab atas definisinya: Dokumentasikan proses dan prosedur yang digunakan di lingkungan Anda, serta individu atau tim yang bertanggung jawab atas definisinya.
  - Identifikasikan proses dan prosedur: Identifikasi aktivitas operasi yang dijalankan untuk mendukung beban kerja Anda. Dokumentasikan aktivitas ini di lokasi yang mudah ditemukan.
  - Tentukan siapa yang memiliki definisi proses atau prosedur: Identifikasi secara khusus individu atau tim yang bertanggung jawab atas spesifikasi aktivitas. Mereka bertanggung jawab untuk memastikan aktivitas dapat dijalankan dengan sukses oleh anggota tim yang memiliki keterampilan memadai dengan izin, akses, serta alat yang sesuai. Jika terdapat masalah saat menjalankan aktivitas tersebut, anggota tim yang menjalankannya bertanggung jawab untuk memberikan tanggapan mendetail yang diperlukan agar aktivitas tersebut dapat ditingkatkan.
  - Dokumentasikan kepemilikan di metadata artefak aktivitas: Prosedur yang diotomatiskan dalam layanan seperti AWS Systems Manager, melalui dokumen, dan AWS Lambda, sebagai fungsi, mendukung dokumentasi informasi metadata sebagai tanda. Dokumentasikan kepemilikan sumber daya menggunakan grup sumber daya atau tanda, yang menentukan informasi kontak dan kepemilikan. Gunakan AWS Organizations untuk membuat kebijakan penandaan serta memastikan dokumentasi informasi kontak serta kepemilikan.

OPS02-BP03 Aktivitas operasi memiliki pemilik teridentifikasi yang bertanggung jawab atas kinerjanya

Pahami siapa yang bertanggung jawab untuk menjalankan aktivitas tertentu terhadap beban kerja yang ditentukan serta alasan adanya tanggung jawab tersebut. Memahami siapa yang bertanggung jawab untuk menjalankan aktivitas dapat memberikan informasi tentang siapa yang akan melakukan aktivitas tersebut, memvalidasi hasilnya, serta memberikan umpan balik kepada pemilik aktivitas.

Manfaat menerapkan praktik terbaik ini: Memahami siapa yang bertanggung jawab untuk menjalankan sebuah aktivitas dapat memberikan informasi tentang siapa yang harus diberi tahu saat diperlukan tindakan dan siapa yang akan melakukan tindakan, memvalidasi hasilnya, serta memberikan umpan balik kepada pemilik aktivitas tersebut.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Aktivitas operasi memiliki pemilik teridentifikasi yang bertanggung jawab atas kinerjanya: Dokumentasikan tanggung jawab untuk menjalankan proses dan prosedur yang digunakan di lingkungan Anda.
  - Identifikasikan proses dan prosedur: Identifikasi aktivitas operasi yang dijalankan untuk mendukung beban kerja Anda. Dokumentasikan aktivitas ini di lokasi yang mudah ditemukan.
  - Tentukan siapa yang bertanggung jawab untuk menjalankan setiap aktivitas: Identifikasikan tim yang bertanggung jawab atas aktivitas. Pastikan mereka memiliki detail aktivitas, keterampilan yang diperlukan dan izin yang tepat, akses, dan alat yang sesuai untuk menjalankan aktivitas. Mereka harus memahami kapan aktivitas tersebut harus dijalankan (misalnya, sesuai peristiwa atau jadwal). Buat informasi ini dapat ditemukan sehingga para anggota organisasi Anda dapat mengidentifikasi siapa yang perlu mereka hubungi, tim atau individu, untuk kebutuhan tertentu.

OPS02-BP04 Anggota tim tahu tanggung jawab mereka

Memahami tanggung jawab peran Anda dan bagaimana Anda berkontribusi terhadap hasil bisnis memberitahukan penentuan prioritas tugas Anda dan mengapa peran Anda itu penting. Ini memampukan anggota tim untuk mengenali kebutuhan dan merespons dengan tepat.

Manfaat dari menjalankan praktik terbaik ini: Memahami tanggung jawab Anda memberikan informasi untuk keputusan yang Anda ambil, tindakan yang Anda ambil, dan penyerahan aktivitas Anda ke pemiliknya yang benar.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

- Pastikan anggota tim memahami peran dan tanggung jawab mereka: Identifikasi peran dan tanggung jawab anggota tim dan pastikan mereka memahami yang diharapkan dari peran mereka. Buat informasi ini dapat ditemukan sehingga para anggota organisasi Anda dapat mengidentifikasi siapa yang perlu mereka hubungi, tim atau individu, untuk kebutuhan tertentu.

OPS02-BP05 Mekanisme tersedia untuk mengidentifikasi tanggung jawab dan kepemilikan

Apabila tidak ada individu atau tim yang diidentifikasi, terdapat jalur eskalasi yang ditetapkan ke seseorang yang memiliki wewenang untuk menetapkan kepemilikan atau rencana untuk penanganan kebutuhan tersebut.

Manfaat menjalankan praktik terbaik ini: Dengan memahami siapa yang memiliki tanggung jawab atau kepemilikan, Anda dapat menghubungi tim atau anggota tim yang tepat untuk melakukan permintaan atau mengalihkan tugas. Dengan adanya orang yang diidentifikasi yang memiliki wewenang untuk menetapkan tanggung jawab atau kepemilikan atau rencana untuk menangani kebutuhan, risiko tidak adanya aksi dan tidak tertanganinya kebutuhan dapat diminimalkan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

- Mekanisme tersedia untuk mengidentifikasi tanggung jawab dan kepemilikan: Sediakan mekanisme yang dapat diakses bagi anggota organisasi untuk menemukan dan mengidentifikasi kepemilikan dan tanggung jawab. Mekanisme ini memungkinkan mereka untuk mengidentifikasi siapa yang harus dihubungi, baik tim maupun individu, untuk kebutuhan tertentu.

OPS02-BP06 Mekanisme tersedia untuk meminta penambahan, perubahan, dan pengecualian

Anda dapat mengajukan permintaan kepada pemilik proses, prosedur, dan sumber daya. Permintaan mencakup penambahan, perubahan, dan pengecualian. Permintaan ini diajukan melalui proses manajemen perubahan. Buat keputusan yang matang untuk menyetujui permintaan apabila memungkinkan dan dianggap tepat setelah dilakukan evaluasi manfaat dan risiko.

Hasil yang diinginkan:

- Anda dapat mengajukan permintaan untuk mengubah proses, prosedur, dan sumber daya berdasarkan kepemilikan yang ditetapkan.
- Perubahan dibuat dengan penuh pertimbangan, dengan memikirkan manfaat dan risikonya.

Antipola umum:

- Anda harus memperbarui cara Anda melakukan deployment aplikasi Anda, tetapi perubahan proses deployment tidak dapat diminta dari tim operasi.
- Rencana pemulihan bencana harus diperbarui, tetapi tidak ada pemilik yang teridentifikasi untuk dimintai perubahan.

Manfaat menjalankan praktik terbaik ini:

- Proses, prosedur, dan sumber daya dapat berubah seiring perubahan persyaratan.
- Pemilik dapat mengambil keputusan yang bijaksana ketika harus membuat perubahan.
- Perubahan dibuat dengan cara yang penuh pertimbangan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Untuk mengimplementasikan praktik terbaik ini, Anda harus dapat meminta perubahan proses, prosedur, dan sumber daya. Proses manajemen perubahan bisa ringan. Dokumentasikan proses manajemen perubahan.

Contoh pelanggan

AnyCompany Retail menggunakan matriks penetapan tanggung jawab (RACI) untuk mengidentifikasi siapa yang memiliki perubahan untuk proses, prosedur, dan sumber daya. Mereka memiliki proses manajemen perubahan terdokumentasi yang ringan dan mudah diikuti. Menggunakan matriks RACI dan proses, siapa pun dapat menyampaikan permintaan perubahan.

Langkah implementasi

1. Identifikasi proses, prosedur, dan sumber daya untuk beban kerja Anda dan pemilik untuk masing-masing. Dokumentasikan dalam sistem manajemen pengetahuan Anda.
   a. Jika Anda belum mengimplementasikan OPS02-BP01 Sumber daya memiliki pemilik teridentifikasi, OPS02-BP02 Proses dan Prosedur memiliki pemilik teridentifikasi, atau OPS02-BP03 Aktivitas operasi memiliki pemilik teridentifikasi yang bertanggung jawab atas kinerjanya, mulai dengan itu terlebih dahulu.
2. Bekerjasamalah dengan pemangku kepentingan di organisasi Anda untuk mengembangkan proses manajemen perubahan. Proses harus meliputi penambahan, perubahan, dan pengecualian untuk sumber daya, proses, dan prosedur.

   a. Anda dapat menggunakan AWS Systems Manager Manajer Perubahan sebagai platform manajemen perubahan untuk sumber daya beban kerja.

3. Dokumentasikan proses manajemen perubahan dalam sistem manajemen pengetahuan Anda.

Tingkat upaya untuk rencana implementasi: Sedang. Mengembangkan proses manajemen perubahan memerlukan penyelarasan dengan beberapa pemangku kepentingan di seluruh organisasi Anda.

Sumber daya

Praktik terbaik terkait:

- OPS02-BP01 Sumber daya memiliki pemilik teridentifikasi - Sumber daya memerlukan pengidentifikasian pemilik sebelum Anda membangun proses manajemen perubahan.
- OPS02-BP02 Proses dan Prosedur memiliki pemilik teridentifikasi - Proses memerlukan pengidentifikasian pemilik sebelum Anda membangun proses manajemen perubahan.
- OPS02-BP03 Aktivitas operasi memiliki pemilik teridentifikasi yang bertanggung jawab atas kinerjanya - Aktivitas operasi memerlukan pengidentifikasian pemilik sebelum Anda membangun proses manajemen perubahan.

Dokumen terkait:

- Panduan Preskriptif AWS - Pedoman mendasar untuk migrasi besar AWS: Membuat matriks RACI
- Laporan Resmi Manajemen Perubahan di Cloud

Layanan terkait:

- Manajer Perubahan AWS Systems Manager

OPS02-BP07 Tanggung jawab antara tim telah dinegosiasi atau ditetapkan sebelumnya

Miliki perjanjian yang telah ditetapkan atau dinegosiasi antara tim yang menjelaskan bagaimana mereka akan bekerja sama dan saling mendukung satu sama lain (contohnya, waktu respons, tujuan tingkat layanan, atau perjanjian tingkat layanan). Saluran komunikasi antar-tim didokumentasi.

Memahami dampak dari pekerjaan tim atas hasil bisnis, dan hasil dari tim lain dan organisasi memberitahukan penentuan prioritas tugas mereka dan membantu mereka merespons dengan tepat.

Ketika tanggung jawab dan kepemilikan tidak ditetapkan atau tidak diketahui, Anda menanggung risiko tidak menangani aktivitas yang diperlukan secara tepat waktu serta risiko munculnya upaya yang berulang dan kemungkinan bertentangan untuk menangani kebutuhan-kebutuhan tersebut.

Hasil yang diinginkan:

- Perjanjian bekerja atau mendukung antar-tim disetujui dan didokumentasi.
- Tim yang mendukung atau bekerja dengan satu sama lain memiliki ekspektasi respons dan saluran komunikasi yang telah ditetapkan sebelumnya.

Antipola umum:

- Masalah terjadi dalam produksi dan dua tim terpisah mulai menyelesaikan masalahnya sendiri-sendiri. Upaya terpisah mereka memperpanjang masa penghentian produksi.
- Tim operasi membutuhkan bantuan dari tim pengembangan, tetapi tidak ada waktu respons yang disepakati. Permintaannya tetap tinggal di timbunan yang belum dikerjakan.

Manfaat menjalankan praktik terbaik ini:

- Tim mengetahui cara berinteraksi dan mendukung satu sama lain.
- Ekspektasi untuk tingkat responsivitas diketahui.
- Saluran komunikasi ditetapkan dengan jelas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Rendah

Panduan implementasi

Mengimplementasikan praktik terbaik ini berarti tidak ada ambiguitas tentang bagaimana tim bekerja dengan satu sama lain. Perjanjian resmi mengodekan bagaimana tim bekerja sama atau mendukung satu sama lain. Saluran komunikasi antar-tim didokumentasi.

Contoh pelanggan

Tim SRE AnyCompany Retail memiliki perjanjian tingkat layanan dengan tim pengembangan mereka. Setiap kali tim pengembangan mengajukan permintaan dalam sistem tiket mereka, mereka dapat

mengantisipasi bahwa respons akan diterima dalam waktu lima belas menit. Jika ada penghentian kerja di lokasi, tim SRE akan memimpin investigasinya dengan dukungan tim pengembangan.

Langkah implementasi

1. Melalui kerja sama dengan pemangku kepentingan di seluruh organisasi Anda, adakan perjanjian antara tim berdasarkan proses dan prosedur.
   a. Jika proses atau prosedur dibagi antara dua tim, kembangkan runbook tentang cara tim akan bekerja sama.
   b. Jika ada ketergantungan antara tim, setujui SLA respons untuk permintaan.
2. Dokumentasikan tanggung jawab dalam sistem manajemen pengetahuan Anda.

Tingkat upaya untuk rencana implementasi: Sedang. Jika belum ada perjanjian antara tim, mungkin akan diperlukan upaya agar para pemangku kepentingan di seluruh organisasi Anda bisa sepakat.

Sumber daya

Praktik terbaik terkait:

- OPS02-BP02 Proses dan Prosedur memiliki pemilik teridentifikasi - Kepemilikan proses harus diidentifikasi sebelum perjanjian diadakan antara tim.
- OPS02-BP03 Aktivitas operasi memiliki pemilik teridentifikasi yang bertanggung jawab atas kinerjanya - Kepemilikan aktivitas operasi harus diidentifikasi sebelum perjanjian diadakan antara tim.

Dokumen terkait:

- Wawasan Eksekutif AWS - Memberdayakan Inovasi dengan Tim Dua Piza
- Pengantar DevOps di AWS - Tim Dua Piza

Budaya organisasi

Berikan dukungan kepada anggota tim Anda sehingga mereka dapat menjadi lebih efektif dalam mengambil tindakan dan mendukung hasil bisnis Anda.

Praktik terbaik

- OPS03-BP01 Sponsor Eksekutif

- OPS03-BP02 Anggota tim diberdayakan untuk bertindak ketika terdapat risiko pada hasil
- OPS03-BP03 Imbauan eskalasi
- OPS03-BP04 Komunikasi yang tepat waktu, jelas, dan dapat ditindaklanjuti
- OPS03-BP05 Mendorong eksperimen
- OPS03-BP06 Mendorong dan mendukung anggota tim untuk mempertahankan dan mengembangkan tingkat keterampilan mereka
- OPS03-BP07 Bekali tim dengan sumber daya dengan sesuai
- OPS03-BP08 Pendapat yang beragam didukung dan dicari di dalam dan lintas tim

OPS03-BP01 Sponsor Eksekutif

Pimpinan senior dengan jelas menetapkan ekspektasi untuk organisasi dan mengevaluasi kesuksesan. Pimpinan senior adalah sponsor, pendukung, dan pendorong untuk pengadopsian praktik terbaik serta perkembangan organisasi.

Manfaat menerapkan praktik terbaik ini: Pimpinan yang terlibat, ekspektasi yang dikomunikasikan dengan jelas, serta tujuan bersama, dapat memastikan anggota tim mengetahui apa yang diharapkan dari mereka. Dengan mengevaluasi kesuksesan, penghalang kesuksesan dapat diidentifikasi, sehingga dapat diatasi melalui intervensi oleh pendukung sponsor atau delegasinya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Sponsor Eksekutif: Pimpinan senior dengan jelas menetapkan ekspektasi untuk organisasi dan mengevaluasi kesuksesan. Pimpinan senior adalah sponsor, pendukung, dan pendorong untuk pengadopsian praktik terbaik serta perkembangan organisasi.
  - Tetapkan ekspektasi: Tentukan dan publikasikan tujuan untuk organisasi Anda, termasuk cara mengukur tujuan tersebut.
  - Lacak capaian tujuan: Ukur capaian bertahap dari tujuan secara rutin serta bagikan hasilnya, agar tindakan yang sesuai dapat segera dilakukan jika hasil sedang dipertaruhkan.
  - Sediakan sumber daya yang diperlukan untuk mencapai target Anda: Lakukan peninjauan secara rutin apakah sumber daya masih sesuai, atau berikan sumber daya tambahan jika diperlukan, berdasarkan pada: informasi baru, perubahan target, tanggung jawab, atau lingkungan bisnis Anda.

- Dukung tim Anda: Tetap berinteraksi dengan tim Anda sehingga Anda memahami bagaimana kondisi mereka dan mengetahui jika ada faktor eksternal yang memengaruhi mereka. Ketika ada faktor eksternal yang memengaruhi kinerja mereka, evaluasi kembali tujuan dan sesuaikan target sebagaimana mestinya. Identifikasikan masalah yang menghambat kemajuan tim Anda. Bertindaklah atas nama tim Anda untuk membantu mengatasi masalah dan menghilangkan beban yang tidak perlu.
- Jadilah penggerak untuk pengadopsian praktik terbaik: Identifikasikan praktik terbaik yang terbukti memberikan manfaat terukur serta beri pengakuan kepada pencipta dan penggunanya. Dukung adopsi lebih lanjut untuk memperbesar manfaat yang dapat dicapai.
- Jadilah penggerak perkembangan tim Anda: Ciptakan budaya peningkatan berkelanjutan. Dukung peningkatan dan perkembangan yang dicapai baik oleh perorangan maupun organisasi. Berikan target jangka panjang yang harus dikejar dan mengharuskan pencapaian bertahap dari waktu ke waktu. Sesuaikan visi ini untuk menyempurnakan kebutuhan, tujuan bisnis, serta lingkungan bisnis Anda seiring dengan perubahannya.

OPS03-BP02 Anggota tim diberdayakan untuk bertindak ketika terdapat risiko pada hasil

Pemilik beban kerja telah menetapkan panduan dan cakupan yang memberdayakan anggota tim untuk merespons ketika terdapat risiko pada hasil. Mekanisme eskalasi digunakan untuk mendapatkan petunjuk ketika peristiwa berada di luar cakupan yang ditetapkan.

Manfaat menerapkan praktik terbaik ini: Dengan menguji dan memvalidasi perubahan sejak dini, Anda dapat mengatasi masalah dengan biaya minim dan membatasi dampak terhadap pelanggan. Dengan menguji sebelum deployment, Anda meminimalkan munculnya kesalahan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

- Anggota tim diberdayakan untuk bertindak ketika terdapat risiko pada hasil: Bekali anggota tim Anda dengan izin, alat, dan peluang untuk mempraktikkan keterampilan yang diperlukan untuk merespons secara efektif.
  - Beri anggota tim Anda peluang untuk mempraktikkan keterampilan yang diperlukan untuk merespons: Sediakan alternatif lingkungan aman di mana proses dan prosedur dapat diuji dan digunakan untuk latihan dengan aman. Lakukan aktivitas permainan untuk memberi kesempatan

pada anggota tim untuk mendapatkan pengalaman merespons insiden dunia nyata dalam lingkungan simulasi yang aman.

- Tetapkan dan kenali wewenang anggota tim untuk bertindak: Tetapkan secara khusus wewenang anggota tim untuk bertindak dengan memberikan izin dan akses ke beban kerja dan komponen yang mereka dukung. Ketahui bahwa mereka diberdayakan untuk bertindak ketika terdapat risiko pada hasil.

OPS03-BP03 Imbauan eskalasi

Anggota tim memiliki mekanisme dan diimbau untuk mengeskalasikan masalah ke pengambil keputusan dan pemangku kepentingan jika mereka yakin terdapat risiko pada hasil. Eskalasi harus dilakukan sejak dini dan secara sering agar risiko dapat diidentifikasi, dan dicegah sebelum menyebabkan insiden.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

- Dorong eskalasi sejak dini dan secara sering: Akui pada tingkat organisasi bahwa eskalasi yang dilakukan sejak dini dan secara sering merupakan praktik terbaik. Akui dan terima pada tingkat organisasi bahwa eskalasi mungkin saja terbukti tidak berdasar, tetapi lebih baik mengambil kesempatan untuk mencegah insiden daripada melewatkan kesempatan untuk melakukan eskalasi.
  - Miliki mekanisme untuk eskalasi: Miliki prosedur terdokumentasi yang menetapkan kapan dan bagaimana eskalasi harus dilakukan. Dokumentasikan sekelompok personel dengan wewenang berjenjang untuk mengambil tindakan atau menyetujui tindakan beserta informasi kontak mereka. Eskalasi harus berlanjut sampai anggota tim yakin bahwa risiko telah dialihkan ke personel yang mampu mengatasinya, atau mereka telah menghubungi personel yang memiliki hak atas risiko dan tanggung jawab atas operasi beban kerja. Personel tersebutlah yang memiliki semua keputusan akhir terkait beban kerja mereka. Eskalasi harus menyertakan sifat risiko, tingkat kekritisan beban kerja, orang yang terkena dampak, apa dampaknya, dan urgensinya, yakni kapan dampak diperkirakan akan dialami.
- Lindungi karyawan yang melakukan eskalasi: Miliki kebijakan yang melindungi anggota tim dari tindakan pembalasan jika mereka melakukan eskalasi di sekitar pengambil keputusan atau pemangku kepentingan yang tidak responsif. Terapkan mekanisme untuk mengidentifikasi apakah hal ini terjadi dan beri respons yang tepat.

OPS03-BP04 Komunikasi yang tepat waktu, jelas, dan dapat ditindaklanjuti

Mekanisme dihadirkan dan digunakan untuk memberikan pengingat secara tepat waktu kepada anggota tim tentang risiko yang diketahui dan peristiwa yang direncanakan. Konteks, detail, dan waktu (ketika memungkinkan) yang diperlukan diberikan untuk membantu menentukan apakah memerlukan tindakan, tindakan apa yang diperlukan, serta untuk melakukan tindakan tepat waktu. Misalnya, memberikan peringatan kerentanan perangkat lunak agar patching dapat dipercepat, atau memberikan peringatan tentang promosi penjualan yang direncanakan sehingga pemberhentian perubahan dapat diimplementasikan untuk menghindari gangguan layanan. Peristiwa yang direncanakan dapat dicatat dalam kalender perubahan atau jadwal pemeliharaan sehingga anggota tim dapat mengidentifikasi aktivitas yang tertunda.

Hasil yang diinginkan:

- Komunikasi memberikan ekspektasi konteks, detail, dan waktu.
- Anggota tim memiliki pemahaman yang jelas tentang kapan dan bagaimana mereka harus bertindak untuk merespons komunikasi.
- Manfaatkan kalender perubahan untuk memberikan pemberitahuan ekspektasi perubahan.

Antipola umum:

- Peringatan positif palsu muncul beberapa kali seminggu. Anda mematikan suara pemberitahuan setiap kali peringatan muncul.
- Anda diminta untuk membuat perubahan pada grup keamanan Anda, tetapi tidak diberi ekspektasi kapan itu harus terjadi.
- Anda terus-menerus menerima pemberitahuan dalam obrolan ketika sistem menaikkan skala tetapi tidak diperlukan tindakan. Anda menghindari saluran obrolan dan melewatkan pemberitahuan penting.
- Perubahan dibuat pada produksi tanpa pemberitahuan kepada tim operasi. Perubahan tersebut memicu peringatan dan tim yang jaga diaktifkan.

Manfaat menjalankan praktik terbaik ini:

- Organisasi Anda menghindari kejemuan karena peringatan.
- Anggota tim dapat bertindak dengan ekspektasi dan konteks yang diperlukan.
- Perubahan dapat dibuat selama jendela perubahan, yang mengurangi risiko.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Untuk mengimplementasikan praktik terbaik ini, Anda harus bekerja sama dengan pemangku kepentingan di seluruh organisasi untuk menyepakati standar komunikasi. Publikasikan standar tersebut ke organisasi Anda. Identifikasi dan singkirkan peringatan yang positif palsu atau selalu muncul. Gunakan kalender perubahan sehingga anggota tim tahu kapan tindakan dapat diambil dan aktivitas apa yang masih perlu dikerjakan. Verifikasi apakah komunikasi menghasilkan tindakan yang jelas dengan konteks yang perlu.

Contoh pelanggan

AnyCompany Retail menggunakan obrolan sebagai media komunikasi utama mereka. Peringatan dan informasi lainnya memenuhi saluran tertentu. Ketika seseorang harus bertindak, hasil yang diinginkan dinyatakan dengan jelas, dan dalam banyak kasus, mereka diberi runbook atau playbook untuk digunakan. Mereka menggunakan kalender perubahan untuk menjadwalkan perubahan besar pada sistem produksi.

Langkah implementasi

1. Analisis peringatan Anda untuk mengetahui adanya peringatan positif palsu atau peringatan yang terus-menerus terpicu. Singkirkan atau ubah peringatan tersebut sehingga akan muncul apabila intervensi manusia diperlukan. Jika peringatan muncul, berikan runbook atau playbook.
    a. Anda dapat menggunakan Dokumen AWS Systems Manager guna membangun playbook dan runbook untuk peringatan.
2. 2Mekanisme diterapkan untuk memberikan pemberitahuan risiko atau peristiwa terencana dengan cara yang jelas dan dapat ditindaklanjuti, melalui peringatan yang memadai untuk memberi respons yang sesuai. Gunakan daftar email atau saluran obrolan untuk mengirimkan pemberitahuan sebelum acara yang sudah direncanakan.
    a. AWS Chatbot dapat digunakan untuk mengirimkan peringatan dan respons terhadap acara dalam platform pengiriman pesan organisasi Anda.
3. Berikan sumber informasi yang dapat diakses di mana acara yang sudah direncanakan dapat ditemukan. Beri pemberitahuan tentang peristiwa yang direncanakan dari sistem yang sama.
    a. Kalender Perubahan AWS Systems Manager dapat digunakan untuk membuat jendela perubahan ketika perubahan dapat terjadi. Hal ini memberi anggota tim pemberitahuan kapan mereka dapat membuat perubahan dengan aman.

4. Pantau pemberitahuan kerentanan dan informasi patch untuk memahami kerentanan risiko tinggi dan potensial yang berkaitan dengan komponen beban kerja Anda. Berikan pemberitahuan kepada anggota tim agar mereka dapat bertindak.
   a. Anda dapat berlangganan Buletin Keamanan AWS untuk menerima pemberitahuan tentang kerentanan di AWS.

Sumber daya

Praktik terbaik terkait:

- OPS07-BP03 Menggunakan runbook untuk menjalankan prosedur - Buat komunikasi dapat ditindaklanjuti dengan memberikan runbook ketika hasilnya diketahui.
- OPS07-BP04 Menggunakan buku panduan untuk menyelidiki masalah - Jika hasilnya tidak diketahui, playbook dapat membuat komunikasi dapat ditindaklanjuti.

Dokumen terkait:

- Buletin Keamanan AWS
- CVE Terbuka

Contoh terkait:

- Well-Architected Labs: Manajemen Inventaris dan Patch (Tingkat 100)

Layanan terkait:

- AWS Chatbot
- Kalender Perubahan AWS Systems Manager
- Dokumen AWS Systems Manager

OPS03-BP05 Mendorong eksperimen

Eksperimen adalah katalis untuk mengubah ide baru menjadi produk dan fitur. Eksperimen mempercepat proses pembelajaran dan membuat anggota tim terus tertarik dan terlibat. Anggota tim didorong untuk sering bereksperimen guna mendorong inovasi. Meskipun hasil yang tidak diinginkan

terjadi, ada nilai dalam memiliki pengetahuan tentang apa yang sebaiknya tidak dilakukan. Anggota tim tidak dihukum untuk eksperimen yang berhasil dengan hasil yang tidak diinginkan.

Hasil yang diinginkan:

- Organisasi Anda mendorong eksperimen untuk mendukung inovasi.
- Eksperimen digunakan sebagai peluang untuk belajar.

Antipola umum:

- Anda ingin menjalankan pengujian A/B tetapi tidak ada mekanisme untuk menjalankan eksperimen tersebut. Anda melakukan deployment perubahan UI tanpa kemampuan untuk mengujinya. Tindakan tersebut mengakibatkan pengalaman pelanggan yang negatif.
- Perusahaan Anda hanya memiliki lingkungan produksi dan staging. Tidak ada lingkungan sandbox untuk bereksperimen dengan fitur atau produk baru sehingga Anda harus bereksperimen di dalam lingkungan produksi.

Manfaat menjalankan praktik terbaik ini:

- Eksperimen mendorong inovasi.
- Anda dapat bereaksi lebih cepat terhadap umpan balik dari pengguna melalui eksperimen.
- Organisasi Anda mengembangkan budaya belajar.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Eksperimen harus dijalankan dengan cara yang aman. Manfaatkan beberapa lingkungan untuk bereksperimen tanpa membahayakan sumber daya produksi. Gunakan pengujian A/B dan tanda fitur untuk menguji eksperimen. Berikan kepada anggota tim kemampuan untuk melakukan eksperimen dalam lingkungan sandbox.

Contoh pelanggan

AnyCompany Retail mendorong eksperimen. Anggota tim dapat menggunakan 20% dari minggu kerja mereka untuk bereksperimen atau mempelajari teknologi baru. Mereka memiliki lingkungan sandbox di mana mereka dapat berinovasi. Pengujian A/B digunakan untuk fitur baru guna memvalidasinya dengan umpan balik nyata pengguna.

Langkah implementasi

1. Bekerjasamalah dengan pimpinan di seluruh organisasi Anda untuk mendukung eksperimen. Anggota tim harus didorong untuk melakukan eksperimen dengan cara yang aman.
2. Berikan kepada anggota tim Anda lingkungan di mana mereka dapat bereksperimen dengan aman. Mereka harus memiliki akses ke lingkungan yang seperti produksi.
   a. Anda dapat menggunakan Akun AWS terpisah untuk membuat lingkungan sandbox untuk eksperimen. AWS Control Tower dapat digunakan untuk menyediakan akun-akun ini.
3. Gunakan tanda fitur dan pengujian A/B untuk bereksperimen dengan aman dan kumpulkan umpan balik pengguna.
   a. Tanda Fitur AWS AppConfig memberikan kemampuan untuk membuat tanda fitur.
   b. Evidently Amazon CloudWatch dapat digunakan untuk menjalankan pengujian A/B selama deployment terbatas.
   c. Anda dapat menggunakan versi AWS Lambda untuk melakukan deployment versi baru fungsi untuk pengujian beta.

Tingkat upaya untuk rencana implementasi: Tinggi. Memberikan kepada anggota tim lingkungan untuk bereksperimen dan cara yang aman untuk melakukan eksperimen dapat memerlukan investasi besar. Anda mungkin juga harus memodifikasi kode aplikasi untuk menggunakan tanda fitur atau mendukung pengujian A/B.

Sumber daya

Praktik terbaik terkait:

- OPS11-BP02 Menjalankan analisis setelah insiden - Belajar dari insiden merupakan pendorong penting untuk inovasi bersama dengan eksperimen.
- OPS11-BP03 Mengimplementasikan loop umpan balik - Siklus umpan balik merupakan bagian penting dari eksperimen.

Dokumen terkait:

- Gambaran Mendalam tentang Budaya Amazon: Eksperimen, Kegagalan, dan Obsesi
- Praktik terbaik untuk membuat dan mengelola akun sandbox di AWS
- Buat Budaya Eksperimen yang Dimampukan oleh Cloud
- Memampukan eksperimen dan inovasi di cloud di SulAmérica Seguros

- Bereksperimen Lebih Sering, Gagal Lebih Jarang
- Mengatur Lingkungan AWS Anda Menggunakan Beberapa Akun - OU Sandbox
- Menggunakan Tanda Fitur AWS AppConfig

Video terkait:

- AWS On Air dengan Evidently Amazon CloudWatch | Acara AWS
- On Air San Fran Summit 2022 AWS dengan Integrasi Tanda Fitur AWS AppConfig dengan Jira
- AWS re:Invent 2022 - Deployment bukan rilis: Kontrol peluncuran Anda dengan tanda fitur (BOA305-R)
- Secara Terprogram Buat Akun AWS dengan AWS Control Tower
- Menyiapkan Lingkungan AWS Multiakun yang Menggunakan Praktik Terbaik untuk AWS Organizations

Contoh terkait:

- Sandbox Inovasi AWS
- Personalisasi Menyeluruh 101 untuk E-Commerce

Layanan terkait:

- Amazon CloudWatch Evidently
- AWS AppConfig
- AWS Control Tower

OPS03-BP06 Mendorong dan mendukung anggota tim untuk mempertahankan dan mengembangkan tingkat keterampilan mereka

Tim harus mengembangkan tingkat keterampilan mereka untuk mengadopsi perkembangan teknologi, serta untuk mengimbangi perubahan permintaan dan tanggung jawab dalam mendukung beban kerja Anda. Perkembangan keterampilan menggunakan teknologi dapat menjadi sumber kepuasan tim dan mendorong inovasi. Dukung anggota tim Anda untuk mendapatkan dan mempertahankan sertifikasi industri yang memvalidasi dan mengakui perkembangan keterampilan mereka. Terapkan pelatihan silang untuk mendorong transfer pengetahuan dan meminimalkan

dampak signifikan yang terjadi karena kehilangan anggota tim berpengalaman yang memiliki keterampilan dan pengetahuan terkait lembaga. Berikan waktu khusus yang terstruktur untuk pembelajaran.

AWS menyediakan sumber daya, termasuk Pusat Sumber Daya untuk Memulai AWS, Blog AWS, AWS Online Tech Talks, Acara dan Webinar AWS, serta Lab AWS Well-Architected, yang menyediakan panduan, contoh, dan ringkasan mendetail untuk mengedukasi tim Anda.

AWS juga membagikan pola dan praktik terbaik yang telah kami pelajari melalui operasi AWS di Amazon Builders' Library serta berbagai macam materi edukasi bermanfaat lainnya dari Blog AWS dan Official AWS Podcast.

Anda harus memanfaatkan sumber daya edukasi yang disediakan oleh AWS seperti lab Well-Architected, AWS Support (Pusat Pengetahuan AWS, Forum Diskusi AWS, dan Pusat AWS Support) dan Dokumentasi AWS untuk mengedukasi tim Anda. Hubungi AWS Support melalui Pusat AWS Support jika Anda memiliki pertanyaan seputar AWS.

AWS Training and Certification menyediakan beberapa pelatihan gratis melalui kursus digital mandiri tentang dasar-dasar AWS. Anda juga dapat mengikuti pelatihan yang dipandu instruktur untuk mendukung perkembangan keterampilan AWS tim Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Dorong dan dukung anggota tim untuk mempertahankan dan mengembangkan tingkat keterampilan mereka: Diperlukan edukasi yang berkelanjutan untuk mengadopsi teknologi baru, mendorong inovasi, dan mengimbangi perubahan permintaan serta tanggung jawab dalam mendukung beban kerja Anda.
  - Sediakan sumber daya untuk kepentingan edukasi: Sediakan waktu khusus yang terstruktur, akses ke materi pelatihan, sumber daya lab, dan dukung partisipasi untuk mengikuti konferensi dan organisasi profesional yang memberikan kesempatan untuk belajar dari pendidik dan rekan. Berikan akses bagi anggota tim junior untuk belajar dari anggota tim senior atau biarkan tim junior meniru pekerjaan tim senior serta melihat metode dan keterampilan mereka. Dorong pembelajaran tentang konten yang tidak terkait langsung dengan pekerjaan agar mereka memiliki pandangan yang lebih luas.
  - Edukasi tim dan interaksi antartim: Buat rencana untuk kebutuhan anggota tim terkait pembelajaran berkelanjutan. Berikan kesempatan kepada anggota tim untuk bergabung dengan

tim lain (sementara atau seterusnya) guna berbagi keterampilan dan praktik terbaik yang bermanfaat bagi organisasi Anda.

- Dukung untuk mendapatkan dan mempertahankan sertifikasi industri: Dukung anggota tim Anda untuk mendapatkan dan mempertahankan sertifikasi industri yang memvalidasi kemampuan yang telah mereka pelajari, serta akui pencapaian mereka.

Sumber daya

Dokumen terkait:

- Pusat Sumber Daya untuk Memulai AWS
- Blog AWS
- Kepatuhan AWS Cloud
- Forum Diskusi AWS
- Dokumentasi AWS
- AWS Online Tech Talks
- Acara dan Webinar AWS
- Pusat Pengetahuan AWS
- AWS Support
- AWS Training and Certification
- Lab AWS Well-Architected,
- Amazon Builders' Library
- Official AWS Podcast.

OPS03-BP07 Bekali tim dengan sumber daya dengan sesuai

Pertahankan kapasitas anggota tim, serta sediakan alat dan sumber daya untuk mendukung kebutuhan beban kerja Anda. Pemberian tugas yang terlalu banyak kepada anggota tim meningkatkan risiko insiden yang diakibatkan oleh kesalahan manusia. Investasi alat dan sumber daya (misalnya, menyediakan otomatisasi untuk aktivitas yang sering dilakukan) dapat meningkatkan efektivitas tim, serta memungkinkan mereka untuk mendukung aktivitas tambahan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Bekali tim dengan sumber daya yang sesuai: Pastikan Anda memiliki pemahaman tentang keberhasilan tim Anda serta faktor yang berkontribusi dalam keberhasilan atau ketidakberhasilan mereka. Dukung tim dengan sumber daya yang sesuai.
  - Pahami kinerja tim: Ukur pencapaian hasil operasional dan pengembangan aset oleh tim Anda. Lacak perubahan pada output dan tingkat kesalahan dari waktu ke waktu. Berinteraksilah dengan tim untuk memahami tantangan terkait pekerjaan yang memengaruhi mereka (misalnya, meningkatnya tanggung jawab, perubahan teknologi, kehilangan personel, atau peningkatan pelanggan yang didukung).
  - Pahami dampak pada kinerja mereka: Tetap berinteraksi dengan tim Anda sehingga Anda memahami bagaimana keadaan mereka dan apakah ada faktor eksternal yang memengaruhi mereka. Ketika tim Anda terdampak oleh faktor eksternal, evaluasi kembali tujuan dan sesuaikan target sebagaimana mestinya. Identifikasi rintangan yang menghambat kemajuan tim Anda. Bertindaklah sebagai perwakilan tim Anda untuk membantu mengatasi rintangan dan menghapus beban yang tidak perlu.
  - Sediakan sumber daya yang diperlukan tim untuk meraih keberhasilan: Tinjau secara teratur apakah sumber daya masih layak, apakah diperlukan sumber daya tambahan, dan buat penyesuaian yang tepat untuk mendukung tim.

OPS03-BP08 Pendapat yang beragam didukung dan dicari di dalam dan lintas tim

Manfaatkan keragaman lintas organisasi untuk mencari berbagai perspektif unik. Gunakan perspektif ini untuk meningkatkan inovasi, menantang asumsi Anda, dan mengurangi risiko bias konfirmasi. Kembangkan inklusi, keragaman, dan kemudahan akses dalam tim Anda untuk mendapatkan perspektif yang menguntungkan.

Budaya organisasi berdampak langsung pada retensi dan kepuasan kerja anggota tim. Dukung keterlibatan dan kemampuan anggota tim Anda untuk mendukung keberhasilan bisnis Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Rendah

Panduan implementasi

- Cari pendapat dan perspektif yang beragam: Dorong kontribusi dari semua orang. Beri suara untuk kelompok yang kurang terwakili. Rotasikan peran dan tanggung jawab dalam rapat.

- Perluas peran dan tanggung jawab: Sediakan kesempatan bagi anggota tim untuk mengambil peran yang mungkin jarang bisa mereka ambil. Mereka akan mendapatkan pengalaman dan perspektif dari peran tersebut, serta dari interaksi dengan anggota tim baru yang mungkin tidak akan berinteraksi dengan mereka di luar peran tersebut. Mereka akan membawa pengalaman dan perspektif mereka ke peran baru serta anggota tim yang berinteraksi dengan mereka. Begitu perspektif meningkat, kesempatan bisnis tambahan bisa muncul, atau kesempatan baru untuk peningkatan bisa teridentifikasi. Buat anggota tim bergantian dalam melakukan tugas umum yang biasanya dilakukan anggota lain untuk memahami tuntutan dan dampak melakukan tugas tersebut.
- Sediakan lingkungan yang aman dan ramah: Miliki kebijakan dan kontrol yang melindungi mental dan keselamatan fisik anggota tim dalam organisasi Anda. Anggota tim harus bisa berinteraksi tanpa rasa takut akan pembalasan. Ketika anggota tim merasa aman dan diterima, mereka mungkin menjadi lebih terlibat dan produktif. Makin beragam organisasi Anda, makin baik pemahaman Anda tentang orang-orang yang Anda dukung termasuk pelanggan Anda. Ketika anggota tim Anda merasa nyaman, merasa bebas untuk berbicara, dan yakin bahwa suara mereka akan didengar, mereka lebih berpeluang untuk membagikan wawasan berharga (misalnya, peluang pemasaran, kebutuhan aksesibilitas, segmen pasar yang belum terlayani, risiko yang tidak diketahui di lingkungan Anda).
- Dukung anggota tim untuk berpartisipasi penuh: Sediakan sumber daya yang diperlukan bagi karyawan Anda untuk berpartisipasi penuh pada semua aktivitas yang berkaitan dengan pekerjaan. Anggota tim yang menghadapi tantangan harian telah mengembangkan keterampilan untuk pekerjaan di sekitar mereka. Keterampilan yang dikembangkan secara khusus ini bisa memberi keuntungan yang signifikan bagi organisasi Anda. Mendukung anggota tim dengan akomodasi yang diperlukan akan meningkatkan keuntungan yang bisa Anda terima dari kontribusi mereka.

Persiapan

Untuk menyiapkan keunggulan operasional, Anda harus memahami beban kerja Anda serta perkiraan perilakunya. Dengan begitu Anda akan mampu merancangnya agar dapat menyediakan wawasan tentang statusnya dan membangun prosedur untuk mendukungnya.

Untuk menyiapkan keunggulan operasional, Anda perlu melakukan hal-hal berikut ini:

Topik

Menerapkan observabilitas

Terapkan observabilitas dalam beban kerja Anda sehingga Anda dapat memahami statusnya dan membuat keputusan berbasis data berdasarkan persyaratan bisnis.

Observabilitas lebih dari sekadar pemantauan sederhana, memberikan pemahaman yang komprehensif tentang cara kerja internal sistem berdasarkan output eksternalnya. Berakar pada metrik, log, dan jejak, observabilitas menawarkan wawasan mendalam tentang perilaku dan dinamika sistem. Dengan observabilitas yang efektif, tim dapat membedakan pola, anomali, dan tren, memungkinkan mereka untuk secara proaktif mengatasi masalah potensial dan menjaga kesehatan sistem yang optimal.

Mengidentifikasi indikator kinerja utama (KPI) sangat penting untuk memastikan keselarasan antara kegiatan pemantauan dan tujuan bisnis. Penyelarasan ini memastikan bahwa tim membuat keputusan berbasis data menggunakan metrik yang benar-benar penting, mengoptimalkan kinerja sistem dan hasil bisnis.

Selain itu, observabilitas memberdayakan bisnis untuk menjadi proaktif, bukan reaktif. Tim dapat memahami hubungan sebab-akibat dalam sistem mereka, memprediksi dan mencegah masalah, bukan hanya bereaksi terhadapnya. Seiring berkembangnya beban kerja, penting untuk meninjau kembali dan menyempurnakan strategi observabilitas, memastikannya tetap relevan dan efektif.

Praktik terbaik

- OPS04-BP01 Identifikasikan indikator performa utama
- OPS04-BP02 Mengimplementasikan telemetri aplikasi
- OPS04-BP03 Mengimplementasikan telemetri pengalaman pengguna
- OPS04-BP04 Mengimplementasikan telemetri dependensi
- OPS04-BP05 Mengimplementasikan penelusuran terdistribusi

OPS04-BP01 Identifikasikan indikator performa utama

Untuk mengimplementasikan observabilitas dalam beban kerja, Anda memulainya dengan memahami statusnya dan mengambil keputusan berbasis data berdasarkan persyaratan bisnis. Salah satu cara paling efektif untuk memastikan keselarasan antara kegiatan pemantauan dan tujuan bisnis adalah dengan menentukan serta memantau indikator kinerja utama (KPI).

Hasil yang diinginkan: Praktik observabilitas yang efisien yang sangat selaras dengan tujuan bisnis, sehingga memastikan upaya pemantauan selalu memenuhi hasil bisnis yang nyata.

Antipola umum:

- KPI yang tidak ditentukan: Bekerja tanpa KPI yang jelas dapat menyebabkan terlalu banyak atau terlalu sedikit pemantauan, sehingga sinyal-sinyal vital menjadi terlewatkan.
- KPI statis: Tidak meninjau atau menyempurnakan KPI seiring perkembangan beban kerja atau tujuan bisnis.
- Ketidakselarasan: Berfokus pada metrik teknis yang tidak berkorelasi langsung dengan hasil bisnis atau yang lebih sulit untuk berkorelasi dengan masalah dunia nyata.

Manfaat menjalankan praktik terbaik ini:

- Kemudahan identifikasi masalah: KPI bisnis sering memunculkan masalah secara lebih jelas daripada metrik teknis. Pengamatan pada KPI bisnis dapat mengenali masalah dengan lebih efektif daripada memilah-milah banyak metrik teknis.
- Keselarasan bisnis: Memastikan bahwa kegiatan pemantauan secara langsung mendukung tujuan bisnis.
- Efisiensi: Prioritaskan pemantauan sumber daya dan perhatian pada metrik yang penting.
- Proaktif: Kenali dan atasi masalah sebelum memunculkan dampak bisnis yang lebih luas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Untuk menentukan KPI beban kerja secara efektif:

1. Mulailah dengan hasil bisnis: Sebelum menyelami metrik, pahami dahulu hasil bisnis yang diinginkan. Apakah peningkatan penjualan, keterlibatan pengguna yang lebih tinggi, atau waktu respons yang lebih cepat?
2. Korelasikan metrik teknis dengan tujuan bisnis: Tidak semua metrik teknis memiliki dampak langsung terhadap hasil bisnis. Identifikasikan metrik yang berdampak langsung terhadap hasil bisnis, tetapi sering kali lebih mudah mengidentifikasi masalah menggunakan KPI bisnis.
3. Gunakan Amazon CloudWatch: Gunakan CloudWatch untuk menentukan dan memantau metrik yang mewakili KPI Anda.
4. Tinjau dan perbarui KPI secara rutin: Saat beban kerja dan bisnis Anda berkembang, jaga agar KPI Anda tetap relevan.
5. Libatkan pemangku kepentingan: Libatkan tim teknis dan bisnis dalam menentukan dan meninjau KPI.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik terbaik terkait:

- the section called “OPS04-BP02 Mengimplementasikan telemetri aplikasi”
- the section called “OPS04-BP03 Mengimplementasikan telemetri pengalaman pengguna”
- the section called “OPS04-BP04 Mengimplementasikan telemetri dependensi”
- the section called “OPS04-BP05 Mengimplementasikan penelusuran terdistribusi”

Dokumen terkait:

- Praktik Terbaik Observabilitas AWS
- Panduan Pengguna CloudWatch
- Kursus Skill Builder Observabilitas AWS

Video terkait:

- Mengembangkan strategi observabilitas

Contoh terkait:

- Lokakarya One Observability

OPS04-BP02 Mengimplementasikan telemetri aplikasi

Telemetri aplikasi berfungsi sebagai fondasi observabilitas beban kerja Anda. Sangat penting menghadirkan telemetri yang menawarkan wawasan yang dapat ditindaklanjuti tentang keadaan aplikasi Anda serta pencapaian hasil teknis dan bisnis. Dari pemecahan masalah hingga pengukuran dampak fitur baru atau memastikan keselarasan dengan indikator kinerja utama (KPI) bisnis, telemetri aplikasi menjadi patokan bagi cara Anda membangun, mengoperasikan, dan mengembangkan beban kerja Anda.

Metrik, log, dan jejak merupakan tiga pilar utama observabilitas. Ketiganya berfungsi sebagai alat diagnostik yang menggambarkan keadaan aplikasi Anda. Seiring waktu, tiga hal ini membantu menciptakan garis acuan dan mengidentifikasi anomali. Namun, untuk memastikan keselarasan antara aktivitas pemantauan dan tujuan bisnis, KPI harus ditentukan dan dipantau. KPI bisnis sering kali mempermudah identifikasi masalah dibandingkan dengan metrik teknis saja.

Jenis telemetri lainnya, seperti pemantauan pengguna nyata (RUM) dan transaksi sintetis, melengkapi sumber-sumber data primer ini. RUM menawarkan wawasan tentang interaksi pengguna waktu nyata, sedangkan transaksi sintetis menyimulasikan perilaku pengguna potensial, sehingga membantu mendeteksi kemacetan sebelum pengguna nyata mengalaminya.

Hasil yang diinginkan: Dapatkan wawasan yang dapat ditindaklanjuti tentang performa beban kerja Anda. Wawasan ini memungkinkan Anda mengambil keputusan proaktif tentang pengoptimalan performa, mencapai peningkatan stabilitas beban kerja, merampingkan proses CI/CD, dan memanfaatkan sumber daya secara efektif.

Antipola umum:

- Observabilitas yang tidak lengkap: Mengabaikan penggunaan observabilitas di setiap lapisan beban kerja, sehingga mengakibatkan titik buta yang dapat mengaburkan performa sistem vital dan wawasan perilaku.

- Tampilan data terfragmentasi: Ketika data tersebar di beberapa alat dan sistem, mempertahankan pandangan yang menyeluruh tentang kondisi dan performa beban kerja Anda menjadi sulit dilakukan.
- Masalah yang dilaporkan pengguna: Tanda kurangnya deteksi masalah yang proaktif melalui telemetri dan pemantauan KPI bisnis.

Manfaat menjalankan praktik terbaik ini:

- Pengambilan keputusan berbasis informasi: Dengan wawasan dari telemetri dan KPI bisnis, Anda dapat mengambil keputusan berbasis data.
- Peningkatan efisiensi operasional: Pemanfaatan sumber daya berbasis data menghasilkan efektivitas biaya.
- Penyempurnaan stabilitas beban kerja: Deteksi dan penyelesaian masalah yang lebih cepat yang menghasilkan peningkatan waktu aktif.
- Perampingan proses CI/CD: Wawasan dari data telemetri memfasilitasi penyempurnaan proses dan pengiriman kode yang andal.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Untuk mengimplementasikan telemetri aplikasi untuk beban kerja Anda, gunakan layanan AWS seperti Amazon CloudWatch dan AWS X-Ray. Amazon CloudWatch menyediakan rangkaian alat pemantauan yang komprehensif, sehingga Anda dapat mengamati sumber daya dan aplikasi Anda di lingkungan AWS dan on-premise. Layanan ini mengumpulkan, melacak, dan menganalisis metrik, menggabungkan dan memantau data log, dan merespons perubahan dalam sumber daya Anda, menyempurnakan pemahaman Anda tentang bagaimana beban kerja Anda beroperasi. Secara bersamaan, AWS X-Ray memungkinkan Anda melacak, menganalisis, dan men-debug aplikasi Anda, sehingga memberi Anda pemahaman yang mendalam tentang perilaku beban kerja Anda. Dengan fitur seperti peta layanan, distribusi latensi, dan lini waktu penelusuran, X-Ray memberikan wawasan tentang performa beban kerja Anda dan hambatan yang memengaruhinya.

Langkah implementasi

1. Identifikasikan data apa yang akan dikumpulkan: Pastikan metrik, log, dan jejak penting yang akan menawarkan wawasan substansial tentang kondisi, performa, dan perilaku beban kerja Anda.

2. Deploy agen CloudWatch : Agen CloudWatch berperan penting dalam penyediaan metrik dan log sistem serta aplikasi dari beban kerja Anda dan infrastruktur yang mendasarinya. Agen CloudWatch juga dapat digunakan untuk mengumpulkan OpenTelemetry atau jejak X-Ray dan mengirimkannya ke X-Ray.
3. Tentukan dan pantau KPI bisnis: Tetapkan metrik kustom yang selaras dengan hasil bisnis.
4. Lengkapi aplikasi Anda dengan AWS X-Ray: Selain men-deploy agen CloudWatch, sangat penting untuk melengkapi aplikasi Anda untuk menghasilkan data jejak. Proses ini dapat memberikan wawasan lebih lanjut tentang perilaku dan performa beban kerja Anda.
5. Lakukan standardisasi pengumpulan data di seluruh aplikasi Anda: Lakukan standardisasi praktik pengumpulan data di seluruh aplikasi Anda. Keseragaman bermanfaat dalam mengorelasikan dan menganalisis data, sehingga memberikan pandangan yang komprehensif tentang perilaku aplikasi Anda.
6. Analisis dan bertindak berdasarkan data: Setelah pengumpulan dan normalisasi data dilakukan, gunakan Amazon CloudWatch untuk analisis metrik dan log, dan AWS X-Ray untuk analisis jejak. Analisis tersebut dapat menghasilkan wawasan penting tentang kondisi, performa, dan perilaku beban kerja Anda, sehingga memandu proses pengambilan keputusan Anda.

Tingkat upaya untuk rencana implementasi: Tinggi

Sumber daya

Praktik terbaik terkait:

- OPS04-BP01 Identifikasikan indikator performa utama
- OPS04-BP03 Mengimplementasikan telemetri pengalaman pengguna
- OPS04-BP04 Mengimplementasikan telemetri dependensi
- OPS04-BP05 Mengimplementasikan penelusuran terdistribusi

Dokumen terkait:

- Praktik Terbaik Observabilitas AWS
- Panduan Pengguna CloudWatch
- Panduan AWS X-Ray untuk Pengembang
- Menginstrumentasikan sistem terdistribusi untuk visibilitas operasional
- Kursus Skill Builder Observabilitas AWS

- Apa yang Baru dengan Amazon CloudWatch
- Apa yang Baru dengan AWS X-Ray

Video terkait:

- AWS re:Invent 2022 - Praktik terbaik observabilitas di Amazon
- AWS re:Invent 2022 - Mengembangkan strategi observabilitas

Contoh terkait:

- Lokakarya One Observability
- Pustaka Solusi AWS: Pemantauan Aplikasi dengan Amazon CloudWatch

OPS04-BP03 Mengimplementasikan telemetri pengalaman pengguna

Memperoleh wawasan yang mendalam tentang pengalaman dan interaksi pelanggan dengan aplikasi Anda adalah hal krusial. Pemantauan pengguna nyata (RUM) dan transaksi sintetis menjadi alat yang ampuh untuk tujuan ini. RUM menyediakan data tentang interaksi pengguna nyata yang memberikan perspektif kepuasan pengguna tanpa filter, sementara transaksi sintetis mensimulasikan interaksi pengguna, sehingga membantu mendeteksi potensi masalah bahkan sebelum berdampak pada pengguna nyata.

Hasil yang diinginkan: Pandangan yang menyeluruh tentang pengalaman pelanggan, deteksi masalah yang proaktif, dan optimalisasi interaksi pengguna untuk memberikan pengalaman digital yang mulus.

Antipola umum:

- Aplikasi tanpa pemantauan pengguna nyata (RUM):
  - Deteksi masalah yang tertunda: Tanpa RUM, Anda mungkin tidak menyadari kemacetan atau masalah performa sampai pengguna mengeluh. Pendekatan reaktif ini dapat menyebabkan ketidakpuasan pelanggan.
  - Tidak adanya wawasan pengalaman pengguna: Tanpa menggunakan RUM, Anda kehilangan data penting yang menunjukkan bagaimana pengguna nyata berinteraksi dengan aplikasi Anda, sehingga membatasi kemampuan Anda untuk mengoptimalkan pengalaman pengguna.
- Aplikasi tanpa transaksi sintetis:

- Kasus edge yang terlewatkan: Transaksi sintetis membantu Anda menguji jalur dan fungsi yang mungkin tidak sering digunakan oleh pengguna biasa tetapi sangat penting untuk fungsi bisnis tertentu. Tanpanya, jalur-jalur tersebut bisa mengalami kesalahan fungsi dan luput dari perhatian.
- Memeriksa masalah saat aplikasi tidak digunakan: Pengujian sintetis rutin dapat mensimulasikan saat-saat ketika pengguna nyata tidak berinteraksi secara aktif dengan aplikasi Anda, sehingga memastikan sistem selalu berfungsi dengan benar.

Manfaat menjalankan praktik terbaik ini:

- Deteksi masalah proaktif: Identifikasikan dan atasi potensi masalah sebelum berdampak pada pengguna nyata.
- Pengalaman pengguna yang dioptimalkan: Umpan balik yang berkelanjutan dari RUM membantu menyempurnakan dan meningkatkan pengalaman pengguna secara keseluruhan.
- Wawasan tentang performa perangkat dan browser: Memahami performa aplikasi Anda di berbagai perangkat dan browser, sehingga memungkinkan pengoptimalan lebih lanjut.
- Alur kerja bisnis yang divalidasi: Transaksi sintetis yang rutin memastikan fungsionalitas inti dan jalur-jalur kritis tetap berjalan dan efisien.
- Performa aplikasi yang ditingkatkan: Manfaatkan wawasan yang dikumpulkan dari data pengguna nyata untuk meningkatkan responsivitas dan keandalan aplikasi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Untuk memanfaatkan RUM dan transaksi sintetis untuk telemetri aktivitas pengguna, AWS menawarkan layanan seperti Amazon CloudWatch RUM dan Amazon CloudWatch Synthetics. Metrik, log, dan jejak, ditambah dengan data aktivitas pengguna, memberikan pandangan yang komprehensif tentang status operasional aplikasi dan pengalaman pengguna.

Langkah implementasi

1. Lakukan deployment Amazon CloudWatch RUM: Integrasikan aplikasi Anda dengan CloudWatch RUM untuk mengumpulkan, menganalisis, dan menyajikan data pengguna nyata.
   a. Gunakan perpustakaan JavaScript CloudWatch RUM untuk mengintegrasikan RUM dengan aplikasi Anda.

   b. Siapkan dasbor untuk memvisualisasikan dan memantau data pengguna nyata.

2. Konfigurasikan CloudWatch Synthetics: Buat canary, atau rutinitas terprogram, yang mensimulasikan interaksi pengguna dengan aplikasi Anda.
   a. Tentukan alur kerja dan jalur aplikasi kritis.
   b. Rancang canary menggunakan skrip CloudWatch Synthetics untuk mensimulasikan interaksi pengguna untuk jalur-jalur tersebut.
   c. Jadwalkan dan pantau canary agar berjalan pada interval tertentu, sehingga memastikan pemeriksaan performa yang konsisten.
3. Analisis dan tindak lanjuti data: Manfaatkan data dari RUM dan transaksi sintetis untuk mendapatkan wawasan dan mengambil tindakan korektif ketika anomali terdeteksi. Gunakan dasbor dan alarm CloudWatch untuk tetap memutakhirkan informasi.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik terbaik terkait:

- OPS04-BP01 Identifikasikan indikator performa utama
- OPS04-BP02 Mengimplementasikan telemetri aplikasi
- OPS04-BP04 Mengimplementasikan telemetri dependensi
- OPS04-BP05 Mengimplementasikan penelusuran terdistribusi

Dokumen terkait:

- Panduan Amazon CloudWatch RUM
- Panduan Amazon CloudWatch Synthetics

Video terkait:

- Mengoptimalkan aplikasi melalui wawasan pengguna akhir dengan RUM
- AWS on Air ft. Pemantauan Pengguna Nyata untuk Amazon CloudWatch

Contoh terkait:

- Lokakarya One Observability
- Repositori Git untuk Klien Web Amazon CloudWatch RUM
- Menggunakan Amazon CloudWatch Synthetics untuk mengukur waktu pemuatan halaman

OPS04-BP04 Mengimplementasikan telemetri dependensi

Telemetri dependensi sangat penting untuk memantau kondisi dan performa layanan dan komponen eksternal yang diandalkan oleh beban kerja Anda. Hal ini memberikan wawasan berharga tentang keterjangkauan, batas waktu, dan peristiwa penting lainnya yang terkait dengan dependensi seperti DNS, basis data, atau API pihak ketiga. Dengan menginstrumentasi aplikasi Anda agar menghasilkan metrik, log, dan jejak tentang dependensi ini, Anda mendapatkan pemahaman yang lebih jelas tentang potensi kemacetan, masalah performa, atau kegagalan yang dapat memengaruhi beban kerja Anda.

Hasil yang diinginkan: Dependensi yang diandalkan beban kerja Anda menunjukkan performa sesuai harapan, sehingga Anda dapat secara proaktif mengatasi masalah dan memastikan performa beban kerja yang optimal.

Antipola umum:

- Mengabaikan dependensi eksternal: Hanya berfokus pada metrik aplikasi internal sambil mengabaikan metrik yang berkaitan dengan dependensi eksternal.
- Kurangnya pemantauan proaktif: Menunggu masalah muncul alih-alih terus memantau kondisi dan performa dependensi.
- Pemantauan model silo: Menggunakan beberapa alat pemantauan yang berbeda-beda sehingga wawasan tentang kondisi dependensi menjadi terfragmentasi dan tidak konsisten.

Manfaat menjalankan praktik terbaik ini:

- Peningkatan keandalan beban kerja: Dengan memastikan bahwa dependensi eksternal terus-menerus tersedia dan berkinerja optimal.
- Deteksi dan penyelesaian masalah yang lebih cepat: Secara proaktif mengidentifikasi dan menangani masalah pada dependensi sebelum berdampak pada beban kerja.
- Pandangan menyeluruh: Mendapatkan pandangan yang menyeluruh tentang komponen internal dan eksternal yang memengaruhi kondisi beban kerja.

- Peningkatan skalabilitas beban kerja: Dengan memahami batas skalabilitas dan karakteristik performa dependensi eksternal.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Implementasikan telemetri dependensi dengan memulai dari identifikasi layanan, infrastruktur, dan proses yang digunakan oleh beban kerja Anda. Ukur seperti apa kondisi yang baik ketika dependensi berfungsi sesuai harapan, kemudian tentukan data apa yang diperlukan untuk mengukurnya. Dengan informasi tersebut, Anda dapat membuat dasbor dan peringatan yang memberikan wawasan kepada tim operasi Anda tentang status dependensi tersebut. Gunakan alat AWS untuk menemukan dan mengukur dampak ketika dependensi tidak dapat menunjukkan hasil sesuai kebutuhan. Selalu tinjau ulang strategi Anda agar memperhitungkan perubahan prioritas, sasaran, dan wawasan yang diperoleh.

Langkah implementasi

Untuk mengimplementasikan telemetri dependensi secara efektif:

1. Identifikasikan dependensi eksternal: Lakukan kolaborasi dengan pemangku kepentingan untuk menentukan dependensi eksternal yang diandalkan oleh beban kerja Anda. Dependensi eksternal dapat mencakup layanan seperti basis data eksternal, API pihak ketiga, rute konektivitas jaringan ke lingkungan lain, dan layanan DNS. Langkah pertama menuju telemetri dependensi yang efektif adalah memiliki pemahaman yang menyeluruh tentang apa saja dependensi tersebut.
2. Kembangkan strategi pemantauan: Setelah Anda memiliki gambaran yang jelas tentang dependensi eksternal Anda, rancanglah strategi pemantauan yang disesuaikan dengan dependensi tersebut. Ini melibatkan pemahaman tingkat kekritisan setiap dependensi, perilaku yang diharapkan, dan perjanjian atau target tingkat layanan (SLA atau SLT) terkait. Siapkan peringatan proaktif untuk memberi tahu Anda tentang perubahan status atau penyimpangan performa.
3. Manfaatkan Amazon CloudWatch Internet Monitor: Layanan ini menawarkan wawasan tentang internet global, sehingga membantu memahami pemadaman atau gangguan yang mungkin memengaruhi dependensi eksternal Anda.
4. Mutakhirkan informasi dengan AWS Health Dashboard: Layanan ini memberikan peringatan dan panduan remediasi ketika AWS mengalami peristiwa yang dapat memengaruhi layanan Anda.

5. Lengkapi aplikasi Anda dengan AWS X-Ray: AWS X-Ray memberikan wawasan tentang bagaimana performa aplikasi dan dependensi yang mendasarinya. Dengan melacak permintaan dari awal hingga akhir, Anda dapat mengidentifikasi kemacetan atau kegagalan dalam layanan eksternal atau komponen yang diandalkan oleh aplikasi Anda.
6. Gunakan Amazon DevOps Guru: Layanan berbasis pembelajaran mesin ini mengidentifikasi masalah operasional, memprediksi kapan masalah kritis mungkin terjadi, dan merekomendasikan tindakan spesifik yang harus diambil. Layanan ini sangat bermanfaat untuk mendapatkan wawasan tentang dependensi dan menentukan bahwa dependensi bukan sumber masalah operasional.
7. Pantau secara teratur: Terus pantau metrik dan log yang berkaitan dengan dependensi eksternal. Siapkan peringatan untuk perilaku tak terduga atau performa yang menurun.
8. Lakukan validasi setelah perubahan: Setiap kali ada pembaruan atau perubahan pada salah satu dependensi eksternal, lakukan validasi performa dan periksa keselarasannya dengan persyaratan aplikasi Anda.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik terbaik terkait:

- OPS04-BP01 Identifikasikan indikator performa utama
- OPS04-BP02 Mengimplementasikan telemetri aplikasi
- OPS04-BP03 Mengimplementasikan telemetri pengalaman pengguna
- OPS04-BP05 Mengimplementasikan penelusuran terdistribusi

Dokumen terkait:

- Apa itu AWS Health?
- Menggunakan Amazon CloudWatch Internet Monitor
- Panduan AWS X-Ray untuk Pengembang
- Panduan Pengguna Amazon DevOps Guru

Video terkait:

- Visibilitas tentang bagaimana masalah internet memengaruhi performa aplikasi
- Pengantar Amazon DevOps Guru

Contoh terkait:

- Mendapatkan wawasan operasional dengan AIOps menggunakan Amazon DevOps Guru
- AWS Health Aware

OPS04-BP05 Mengimplementasikan penelusuran terdistribusi

Penelusuran terdistribusi menawarkan cara untuk memantau dan memvisualisasikan permintaan yang melintasi berbagai komponen sistem terdistribusi. Dengan menangkap data jejak dari berbagai sumber dan menganalisisnya dalam tampilan terpadu, tim dapat lebih memahami bagaimana permintaan mengalir, di mana kemacetan terjadi, dan di mana upaya pengoptimalan harus difokuskan.

Hasil yang diinginkan: Dapatkan tampilan menyeluruh permintaan yang mengalir melewati sistem terdistribusi Anda, sehingga memungkinkan debugging yang presisi, performa yang dioptimalkan, dan pengalaman pengguna yang lebih baik.

Antipola umum:

- Instrumentasi yang tidak konsisten: Tidak semua layanan dalam sistem terdistribusi diinstrumentasi untuk penelusuran.
- Mengabaikan latensi: Hanya berfokus pada kesalahan dan tidak mempertimbangkan latensi atau penurunan performa bertahap.

Manfaat menjalankan praktik terbaik ini:

- Gambaran umum sistem yang komprehensif: Memvisualisasikan seluruh jalur permintaan, dari masuk hingga keluar.
- Debugging yang disempurnakan: Mengidentifikasi dengan cepat di mana kegagalan atau masalah performa terjadi.
- Pengalaman pengguna yang ditingkatkan: Memantau dan mengoptimalkan berdasarkan data pengguna aktual, memastikan sistem memenuhi tuntutan dunia nyata.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Mulailah dengan mengidentifikasi semua elemen beban kerja Anda yang memerlukan instrumentasi. Setelah semua komponen diperhitungkan, manfaatkan alat seperti AWS X-Ray dan OpenTelemetry untuk mengumpulkan data jejak untuk dianalisis dengan alat seperti X-Ray dan Amazon CloudWatch ServiceLens Map. Lakukan peninjauan rutin dengan developer, dan lengkapi diskusi tersebut dengan alat seperti Amazon DevOps Guru, Analitik X-Ray, dan Wawasan X-Ray untuk membantu mengungkap temuan yang lebih mendalam. Buat peringatan dari data jejak untuk memberi tahu kapan hasil, sebagaimana didefinisikan dalam rencana pemantauan beban kerja, mengandung risiko.

Langkah implementasi

Untuk mengimplementasikan penelusuran terdistribusi secara efektif:

1. Adopsi AWS X-Ray: Integrasikan X-Ray ke dalam aplikasi Anda untuk mendapatkan wawasan tentang perilakunya, memahami performanya, dan mengenali kemacetan. Manfaatkan Wawasan X-Ray untuk analisis jejak otomatis.
2. Lengkapi layanan Anda: Verifikasi bahwa setiap layanan, dari fungsi AWS Lambda hingga instans EC2, mengirimkan data jejak. Makin banyak layanan yang Anda lengkapi, maka makin jelas tampilan yang menyeluruh.
3. Sertakan Pemantauan Pengguna Nyata CloudWatch dan pemantauan sintetis: Integrasikan Pemantauan Pengguna Nyata (RUM) dan pemantauan sintetis dengan X-Ray. Hal ini memungkinkan perekaman pengalaman pengguna dunia nyata dan simulasi interaksi pengguna untuk mengidentifikasi potensi masalah.
4. Gunakan agen CloudWatch: Agen ini dapat mengirimkan jejak dari X-Ray atau OpenTelemetry, sehingga meningkatkan kedalaman wawasan yang diperoleh.
5. Gunakan Amazon DevOps Guru: DevOps Guru menggunakan data dari X-Ray, CloudWatch, AWS Config, dan AWS CloudTrail untuk memberikan rekomendasi yang dapat ditindaklanjuti.
6. Lakukan analisis jejak: Tinjau data jejak secara rutin untuk membedakan pola, anomali, atau kemacetan yang dapat memengaruhi performa aplikasi Anda.
7. Siapkan peringatan: Konfigurasikan alarm di CloudWatch untuk pola yang tidak biasa atau latensi yang meluas, sehingga memungkinkan penanganan masalah secara proaktif.
8. Peningkatan berkelanjutan: Tinjau ulang strategi penelusuran Anda saat layanan ditambahkan atau dimodifikasi untuk menangkap semua titik data yang relevan.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik terbaik terkait:

- OPS04-BP01 Identifikasikan indikator performa utama
- OPS04-BP02 Mengimplementasikan telemetri aplikasi
- OPS04-BP03 Mengimplementasikan telemetri pengalaman pengguna
- OPS04-BP04 Mengimplementasikan telemetri dependensi

Dokumen terkait:

- Panduan AWS X-Ray untuk Pengembang
- Panduan Pengguna agen Amazon CloudWatch
- Panduan Pengguna Amazon DevOps Guru

Video terkait:

- Gunakan Wawasan AWS X-Ray
- AWS on Air ft. Observabilitas: Amazon CloudWatch dan AWS X-Ray

Contoh terkait:

- Menginstrumentasi Aplikasi Anda dengan AWS X-Ray

Desain operasi

Adopsi pendekatan yang meningkatkan aliran perubahan ke dalam produksi dan yang membantu pemfaktoran ulang, umpan balik cepat atas kualitas, dan perbaikan bug. Ini mempercepat perubahan yang bermanfaat memasuki produksi, membatasi masalah yang di-deploy, dan menyediakan identifikasi cepat serta perbaikan masalah akibat aktivitas deployment.

Di AWS, Anda dapat menampilkan keseluruhan beban kerja Anda (aplikasi, infrastruktur, kebijakan, tata kelola, dan operasi) dalam bentuk kode. Beban kerja Anda dapat ditetapkan dalam kode dan

diperbarui menggunakan kode. Ini artinya Anda dapat menerapkan disiplin teknik yang sama yang Anda gunakan untuk kode aplikasi pada setiap elemen tumpukan Anda.

Praktik terbaik

- OPS05-BP01 Menggunakan kontrol versi
- OPS05-BP02 Menguji dan memvalidasi perubahan
- OPS05-BP03 Menggunakan sistem manajemen konfigurasi
- OPS05-BP04 Menggunakan sistem manajemen build dan deployment
- OPS05-BP05 Melakukan manajemen patch
- OPS05-BP06 Membagikan standar desain
- OPS05-BP07 Mengimplementasikan praktik untuk meningkatkan kualitas kode
- OPS05-BP08 Menggunakan beberapa lingkungan
- OPS05-BP09 Membuat perubahan yang sering, kecil, dan dapat dikembalikan
- OPS05-BP10 Mengotomatiskan integrasi dan deployment sepenuhnya

OPS05-BP01 Menggunakan kontrol versi

Gunakan kontrol versi untuk memungkinkan pelacakan perubahan dan rilis.

Banyak layanan AWS menawarkan kemampuan kontrol versi. Gunakan sistem kontrol revisi atau sumber seperti AWS CodeCommit untuk mengelola kode dan artefak lain, seperti templat AWS CloudFormation yang dikontrol versi dari infrastruktur Anda.

Hasil yang diinginkan: Tim Anda berkolaborasi mengerjakan kode. Saat digabungkan, kode tersebut konsisten dan tidak ada perubahan yang hilang. Kesalahan mudah dibatalkan melalui versioning yang benar.

Antipola umum:

- Anda telah mengembangkan dan menyimpan kode di stasiun kerja Anda. Anda mengalami kegagalan penyimpanan yang tidak dapat dipulihkan di stasiun kerja lalu kode Anda hilang.
- Setelah menimpa kode yang ada dengan perubahan Anda, Anda memulai ulang aplikasi namun sudah tidak dapat beroperasi lagi. Anda tidak bisa membatalkan perubahan.
- Anda memiliki write lock pada file laporan yang perlu diedit orang lain. Mereka meminta Anda untuk berhenti mengerjakannya agar mereka bisa menyelesaikan tugas mereka.

- Tim penelitian Anda telah mengerjakan analisis mendetail yang membentuk pekerjaan mendatang Anda. Seseorang secara tidak sengaja menyimpan daftar belanjanya dan menimpa laporan akhir. Anda tidak bisa membatalkan perubahan dan harus membuat ulang laporan tersebut.

Manfaat menjalankan praktik terbaik ini: Dengan menggunakan kemampuan kontrol versi, Anda dapat secara mudah kembali ke versi sebelumnya dengan status baik, dan membatasi risiko kehilangan aset.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Pelihara aset di repositori dengan kontrol versi. Tindakan ini mendukung pelacakan perubahan, deployment versi baru, deteksi perubahan pada versi yang ada, dan pengembalian ke versi sebelumnya (misalnya, kembali ke versi dengan status baik apabila terjadi kegagalan). Integrasikan kemampuan kontrol versi sistem manajemen konfigurasi Anda ke dalam prosedur Anda.

Sumber daya

Praktik terbaik terkait:

- OPS05-BP04 Menggunakan sistem manajemen build dan deployment

Dokumen terkait:

- Apa Itu AWS CodeCommit?

Video terkait:

- Pengantar AWS CodeCommit

OPS05-BP02 Menguji dan memvalidasi perubahan

Setiap perubahan yang di-deploy harus diuji untuk menghindari kesalahan dalam produksi. Praktik terbaik ini difokuskan untuk menguji perubahan dari kontrol versi hingga build artefak. Di samping perubahan kode aplikasi, pengujian harus menyertakan infrastruktur, konfigurasi, kontrol keamanan, dan prosedur operasi. Ada banyak bentuk pengujian, dari uji unit hingga analisis komponen

perangkat lunak (SCA). Makin ke kiri pengujian dalam proses integrasi dan pengiriman perangkat lunak menghasilkan tingkat kepastian kualitas artefak yang lebih tinggi.

Organisasi Anda harus mengembangkan standar pengujian untuk semua artefak perangkat lunak. Pengujian otomatis dapat mengurangi kerja yang melelahkan dan mencegah kesalahan pengujian manual. Uji manual mungkin diperlukan dalam beberapa kasus. Developer harus memiliki akses ke hasil uji otomatis untuk menciptakan loop umpan balik yang meningkatkan kualitas perangkat lunak.

Hasil yang diinginkan: Perubahan perangkat lunak Anda diuji sebelum dikirim. Pengembang memiliki akses ke hasil pengujian dan validasi. Organisasi memiliki standar pengujian yang berlaku untuk semua perubahan perangkat lunak.

Antipola umum:

- Anda men-deploy perubahan perangkat lunak baru tanpa pengujian apa pun. Perangkat lunak gagal berjalan dalam produksi, dan mengakibatkan matinya sistem.
- Grup keamanan baru di-deploy dengan AWS CloudFormation tanpa diuji dalam lingkungan pra-produksi. Grup keamanan tersebut menjadikan aplikasi Anda tidak terjangkau oleh pelanggan Anda.
- Sebuah metode diubah tanpa pengujian unit. Perangkat lunak gagal saat di-deploy ke produksi.

Manfaat menjalankan praktik terbaik ini: Tingkat kegagalan perubahan deployment perangkat lunak menjadi berkurang. Kualitas perangkat lunak meningkat. Developer memiliki kesadaran yang lebih tinggi tentang kelayakan kode mereka. Kebijakan keamanan dapat diluncurkan dengan penuh keyakinan untuk mendukung kepatuhan organisasi. Perubahan infrastruktur seperti pembaruan kebijakan penskalaan otomatis diuji di awal untuk memenuhi kebutuhan lalu lintas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Pengujian dilakukan pada semua perubahan, dari kode aplikasi hingga infrastruktur, sebagai bagian dari praktik integrasi berkelanjutan Anda. Hasil pengujian dipublikasikan sehingga developer memiliki umpan balik yang cepat. Organisasi memiliki standar pengujian bahwa semua perubahan harus lulus.

Contoh pelanggan

Sebagai bagian dari pipeline integrasi berkelanjutan mereka, AnyCompany Retail melakukan beberapa jenis pengujian pada semua artefak perangkat lunak. Mereka mempraktikkan

pengembangan yang didorong pengujian sehingga semua perangkat lunak memiliki pengujian-pengujian unit. Begitu artefak dibangun, mereka menjalankan pengujian menyeluruh. Setelah pengujian putaran pertama selesai, mereka menjalankan pemindaian keamanan aplikasi statis, yang mencari kerentanan yang dikenali. Developer menerima pesan setelah setiap gerbang pengujian dilalui. Setelah semua pengujian selesai, artefak perangkat lunak disimpan di dalam repositori artefak.

Langkah implementasi

1. Bekerjalah dengan pemangku kepentingan di organisasi Anda untuk mengembangkan standar pengujian untuk artefak perangkat lunak. Pengujian standar apa yang harus dilalui oleh semua artefak? Apakah ada persyaratan kepatuhan atau tata kelola yang harus disertakan di dalam cakupan pengujian? Apakah Anda perlu melakukan pengujian kualitas kode? Setelah pengujian selesai, siapa yang perlu mengetahuinya?
   a. Perintah Rujukan Pipeline Deployment AWS berisi daftar tepercaya untuk jenis-jenis pengujian yang dapat dilakukan pada artefak perangkat lunak sebagai bagian dari pipeline integrasi.
2. Instrumentasikan aplikasi Anda dengan pengujian yang diperlukan berdasarkan standar pengujian perangkat lunak Anda. Setiap set pengujian harus selesai dalam waktu kurang dari sepuluh menit. Pengujian harus berjalan sebagai bagian dari pipeline integrasi.
   a. Amazon CodeGuru Reviewer dapat menguji kode aplikasi Anda untuk mendeteksi kecacatan.
   b. Anda dapat menggunakan AWS CodeBuild untuk melakukan pengujian pada artefak perangkat lunak.
   c. AWS CodePipeline dapat mengorkestrasi pengujian perangkat lunak Anda ke dalam pipeline.

Sumber daya

Praktik terbaik terkait:

- OPS05-BP01 Menggunakan kontrol versi
- OPS05-BP06 Membagikan standar desain
- OPS05-BP10 Mengotomatiskan integrasi dan deployment sepenuhnya

Dokumen terkait:

- Adopsi pendekatan pengembangan yang didorong pengujian
- Pipeline Pengujian AWS CloudFormation Otomatis dengan TaskCat dan CodePipeline

- Membangun pipeline CI/CD DevSecOps AWS yang menyeluruh dengan alat-alat SCA, SAST, dan DAST sumber terbuka
- Memulai pengujian aplikasi nirserver
- Pipeline CI/CD adalah pemandu utama rilisan saya
- Laporan resmi Mempraktikkan Integrasi Berkelanjutan dan Pengiriman Berkelanjutan di AWS

Video terkait:

- AWS re:Invent 2020: Infrastruktur yang dapat diuji: Pengujian integrasi di AWS
- AWS Summit ANZ 2021 - Mendorong strategi yang mengutamakan pengujian dengan CDK dan pengembangan yang didorong pengujian
- Menguji Infrastruktur sebagai Kode dengan AWS CDK

Sumber daya terkait:

- Arsitektur Rujukan Pipeline Deployment AWS - Aplikasi
- Pipeline DevSecOps Kubernetes AWS
- Lokakarya Kebijakan sebagai Kode – Pengembangan yang Didorong Pengujian
- Menjalankan pengujian unit untuk aplikasi Node.js dari GitHub dengan menggunakan AWS CodeBuild
- Menggunakan Serverspec untuk pengembangan kode infrastruktur yang didorong pengujian

Layanan terkait:

- Amazon CodeGuru Reviewer
- AWS CodeBuild
- AWS CodePipeline

OPS05-BP03 Menggunakan sistem manajemen konfigurasi

Gunakan sistem manajemen konfigurasi untuk membuat dan melacak perubahan konfigurasi. Sistem ini mengurangi kesalahan yang disebabkan oleh proses manual dan meminimalkan tingkat upaya untuk melakukan deployment perubahan.

Manajemen konfigurasi statis menetapkan nilai saat menginisialisasi sumber daya yang diharapkan tetap konsisten selama masa pakai sumber daya. Beberapa contoh menyertakan pengaturan konfigurasi untuk web atau server aplikasi pada instans, atau menentukan konfigurasi layanan AWS dalam AWS Management Console atau melalui AWS CLI.

Manajemen konfigurasi dinamis menetapkan nilai saat inisialisasi. Nilai ini dapat atau diharapkan berubah selama masa pakai sumber daya. Misalnya, Anda dapat menetapkan toggle fitur untuk mengaktifkan fungsionalitas dalam kode melalui perubahan konfigurasi, atau mengubah tingkat detail log selama insiden untuk memperoleh lebih banyak data, lalu mengubahnya kembali setelah insiden menghilangkan log yang saat ini tidak dibutuhkan dan pengeluaran yang terkait dengannya.

Di AWS, Anda dapat menggunakan AWS Config untuk terus mengawasi konfigurasi sumber daya AWS Anda di seluruh akun dan Wilayah. Dengan demikian, Anda dapat melacak riwayat konfigurasi mereka, memahami bagaimana perubahan konfigurasi akan memengaruhi sumber daya lainnya, dan mengauditnya terhadap konfigurasi yang diharapkan atau diinginkan dengan menggunakan Aturan AWS Config dan Paket Konformasi AWS Config.

Jika Anda memiliki konfigurasi dinamis di aplikasi Anda yang berjalan di instans Amazon EC2, AWS Lambda, kontainer, perangkat seluler, atau perangkat IoT, Anda dapat menggunakan AWS AppConfig untuk mengonfigurasi, memvalidasi, men-deploy, dan memantaunya di seluruh lingkungan Anda.

Di AWS, Anda dapat membuat pipeline integrasi berkelanjutan/deployment berkelanjutan (CI/CD) menggunakan layanan seperti Alat Developer AWS (misalnya, AWS CodeCommit, AWS CodeBuild, AWS CodePipeline, AWS CodeDeploy, dan AWS CodeStar).

Hasil yang diinginkan: Anda mengonfigurasi, memvalidasi, dan melakukan deployment sebagai bagian dari pipeline integrasi berkelanjutan, pengiriman berkelanjutan (CI/CD) Anda. Anda memantau untuk memvalidasi bahwa konfigurasi sudah benar. Hal ini meminimalkan dampak apa pun terhadap pelanggan dan pengguna akhir.

Antipola umum:

- Anda memperbarui konfigurasi server web secara manual di seluruh armada dan beberapa server menjadi tidak responsif karena kesalahan pembaruan.
- Anda memperbarui armada server aplikasi selama berjam-jam. Inkonsistensi dalam konfigurasi selama perubahan menyebabkan perilaku tak terduga.

- Seseorang telah memperbarui grup keamanan Anda dan server web Anda tidak lagi dapat diakses. Tanpa mengetahui apa yang telah diubah, Anda menghabiskan banyak waktu untuk menyelidiki masalah tersebut sehingga memperpanjang waktu pemulihan.
- Anda mendorong konfigurasi pra-produksi ke dalam produksi melalui CI/CD tanpa validasi. Anda mengekspos pengguna dan pelanggan ke data dan layanan yang salah.

Manfaat menjalankan praktik terbaik ini: Mengadopsi sistem manajemen konfigurasi meminimalkan tingkat upaya untuk membuat dan melacak perubahan, serta mengurangi frekuensi kesalahan yang disebabkan prosedur manual. Sistem manajemen konfigurasi memberikan jaminan sehubungan dengan persyaratan tata kelola, kepatuhan, dan peraturan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Sistem manajemen konfigurasi digunakan untuk melacak dan mengimplementasikan perubahan pada konfigurasi aplikasi dan lingkungan. Sistem manajemen konfigurasi juga digunakan untuk mengurangi kesalahan yang disebabkan oleh proses manual, membuat perubahan konfigurasi berulang dan dapat diaudit, dan mengurangi tingkat upaya.

Langkah implementasi

1. Identifikasikan pemilik konfigurasi.
    a. Buat agar pemilik konfigurasi menyadari kebutuhan kepatuhan, tata kelola, atau peraturan apa pun.
2. Identifikasikan item konfigurasi dan hasil kerja.
    a. Item konfigurasi adalah semua konfigurasi aplikasi dan lingkungan yang dipengaruhi oleh deployment dalam pipeline CI/CD Anda.
    b. Hasil kerja mencakup kriteria keberhasilan, validasi, dan hal-hal yang harus dipantau.
3. Pilih alat untuk manajemen konfigurasi berdasarkan kebutuhan bisnis dan pipeline pengiriman Anda.
4. Pertimbangkan deployment tertimbang seperti deployment canary untuk perubahan konfigurasi yang signifikan guna meminimalkan dampak konfigurasi yang salah.
5. Integrasikan manajemen konfigurasi Anda ke dalam pipeline CI/CD Anda.
6. Validasikan semua perubahan yang didorong.

Sumber daya

Praktik terbaik terkait:

- OPS06-BP01 Antisipasikan perubahan yang tidak berhasil
- OPS06-BP02 Menguji deployment
- OPS06-BP03 Menggunakan strategi deployment yang aman
- OPS06-BP04 Mengotomatiskan pengujian dan pengembalian (rollback)

Dokumen terkait:

- AWS Control Tower
- Akselerator Zona Pendaratan AWS
- AWS Config
- Apa itu AWS Config?
- AWS AppConfig
- Apa itu AWS CloudFormation?
- Alat Developer AWS

Video terkait:

- AWS re:Invent 2022 - Tata kelola dan kepatuhan proaktif untuk beban kerja AWS
- AWS re:Invent 2020: Capai kepatuhan sebagai kode menggunakan AWS Config
- Kelola dan Deploy Konfigurasi Aplikasi dengan AWS AppConfig

OPS05-BP04 Menggunakan sistem manajemen build dan deployment

Gunakan sistem manajemen build dan deployment. Sistem ini mengurangi kesalahan yang disebabkan oleh proses manual dan meminimalkan tingkat upaya untuk melakukan deployment perubahan.

Di AWS, Anda dapat membangun pipeline integrasi berkelanjutan/deployment berkelanjutan (CI/CD) menggunakan layanan seperti Alat Pengembang AWS (misalnya, AWS CodeCommit, AWS CodeBuild, AWS CodePipeline, AWS CodeDeploy, dan AWS CodeStar).

Hasil yang diinginkan: Sistem manajemen build dan deployment Anda mendukung sistem integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) organisasi Anda yang menyediakan kemampuan untuk mengotomatisasi peluncuran aman dengan konfigurasi yang benar.

Antipola umum:

- Setelah menyusun kode pada sistem pengembangan, Anda menyalin file yang dapat dieksekusi ke sistem produksi namun file tersebut gagal untuk memulai. File log lokal mengindikasikan bahwa kegagalan tersebut dikarenakan hilangnya dependensi.
- Anda berhasil membangun aplikasi Anda dengan fitur baru pada lingkungan pengembangan dan memberikan kodenya ke tim jaminan kualitas (QA). Kode tersebut gagal dalam QA karena ada aset statis yang hilang.
- Pada hari Jumat, setelah berupaya keras, Anda berhasil membangun aplikasi Anda secara manual di lingkungan pengembangan Anda termasuk fitur yang baru dikodekan. Pada hari Senin, Anda tidak dapat mengulangi langkah-langkah yang membuat Anda berhasil membangun aplikasi.
- Anda melakukan pengujian yang telah Anda buat untuk rilisan baru Anda. Kemudian Anda menghabiskan minggu selanjutnya untuk mempersiapkan lingkungan pengujian dan melakukan seluruh pengujian integrasi yang ada disusul dengan pengujian kinerja. Kode baru tersebut memiliki dampak kinerja yang tidak dapat diterima dan harus dikembangkan ulang dan kemudian diuji ulang.

Manfaat menerapkan praktik terbaik ini: Dengan menyediakan mekanisme untuk mengatasi aktivitas build dan deployment, Anda mengurangi upaya yang diperlukan untuk melakukan tugas berulang, membebaskan anggota tim Anda untuk fokus pada tugas kreatif mereka yang berharga, serta mengurangi terjadinya kesalahan akibat prosedur manual.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Sistem manajemen build dan deployment digunakan untuk melacak dan mengimplementasikan perubahan, mengurangi kesalahan yang disebabkan oleh proses manual, dan mengurangi upaya yang diperlukan untuk deployment yang aman. Otomatiskan sepenuhnya pipeline integrasi dan deployment dari check-in kode hingga build, pengujian, deployment, dan validasi. Hal ini mempersingkat waktu tunggu, mengurangi biaya, mendorong peningkatan frekuensi perubahan, mengurangi tingkat upaya, dan meningkatkan kolaborasi.

Langkah implementasi

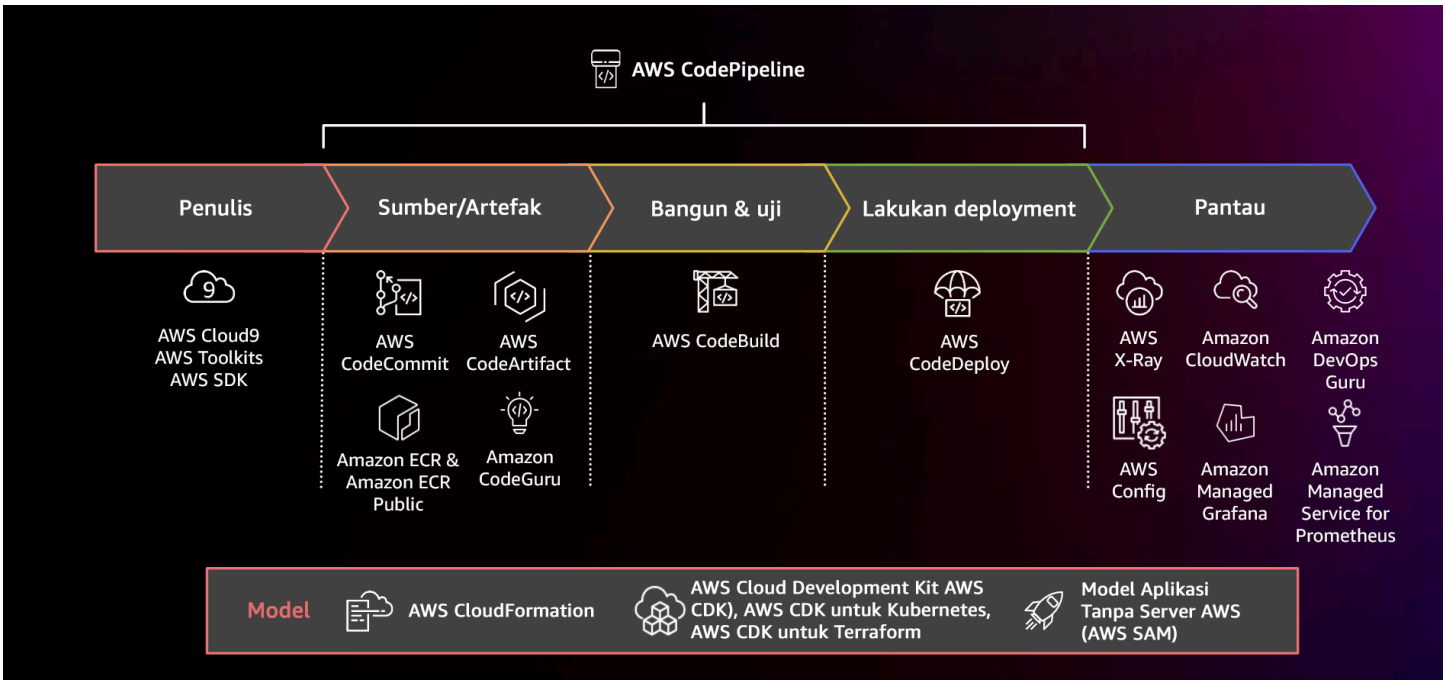


Diagram yang menunjukkan pipeline CI/CD menggunakan AWS CodePipeline dan layanan terkait

1. Gunakan AWS CodeCommit untuk mengontrol versi, menyimpan, dan mengelola aset (seperti dokumen, kode sumber, dan file biner).
2. Gunakan CodeBuild untuk mengompilasi kode sumber Anda, menjalankan pengujian unit, dan memproduksi artefak yang siap untuk deployment.
3. Gunakan CodeDeploy sebagai layanan deployment yang mengotomatiskan deployment aplikasi ke instans Amazon EC2 , instans on-premise, fungsi AWS Lambda nirserver, atau Amazon ECS.
4. Pantau deployment Anda.

Sumber daya

Praktik terbaik terkait:

- OPS06-BP04 Mengotomatiskan pengujian dan pengembalian (rollback)

Dokumen terkait:

- Alat Pengembang AWS
- Apa Itu AWS CodeCommit?

- Apa itu AWS CodeBuild?
- AWS CodeBuild
- Apa itu AWS CodeDeploy?

Video terkait:

- AWS re:Invent 2022 - Praktik terbaik AWS Well-Architected untuk DevOps di AWS

OPS05-BP05 Melakukan manajemen patch

Lakukan manajemen patch untuk mendapatkan fitur, menangani permasalahan, dan menjaga kepatuhan terhadap tata kelola. Otomatiskan manajemen patch untuk mengurangi kesalahan yang disebabkan oleh proses manual, menskalakan, dan mengurangi upaya untuk melakukan patch.

Manajemen patch dan kerentanan adalah bagian dari aktivitas manajemen manfaat dan risiko Anda. Lebih baik miliki infrastruktur tetap dan deploy beban kerja pada status yang diketahui baik dan terverifikasi. Jika tidak memungkinkan, opsi yang tersisa ialah menerapkan patching.

Amazon EC2 Image Builder menyediakan pipeline untuk memperbarui image mesin. Sebagai bagian dari manajemen patch, pertimbangkan Amazon Machine Images (AMI) menggunakan Pipeline image AMI atau image kontainer dengan Pipeline image Docker, sementara AWS Lambda memberikan pola bagi runtime kustom dan pustaka tambahan untuk menghapus kerentanan.

Anda harus mengelola pembaruan pada Amazon Machine Images untuk image Linux atau Windows Server menggunakan Amazon EC2 Image Builder. Anda dapat menggunakan Amazon Elastic Container Registry (Amazon ECR) dengan pipeline yang sudah ada untuk mengelola image Amazon ECS dan mengelola image Amazon EKS. Lambda mencakup fitur manajemen versi.

Patching tidak boleh dilakukan pada sistem produksi tanpa mengujinya terlebih dahulu di lingkungan yang aman. Patch hanya bisa diterapkan jika mendukung hasil operasi atau bisnis. Di AWS, Anda dapat menggunakan AWS Systems Manager Patch Manager untuk mengotomatiskan proses patching sistem terkelola dan menjadwalkan aktivitas menggunakan Periode Pemeliharaan Systems Manager.

Hasil yang diinginkan: Image AMI dan kontainer Anda diberikan patch, diperbarui, dan siap diluncurkan. Anda dapat melacak status semua image yang di-deploy dan mengetahui kepatuhan patch. Anda dapat melaporkan status saat ini dan memiliki proses untuk memenuhi kebutuhan kepatuhan Anda.

Antipola umum:

- Anda diberi tugas untuk menerapkan semua patch keamanan baru dalam waktu dua jam yang menyebabkan beberapa pemadaman akibat ketidaksesuaian aplikasi dengan patch.
- Pustaka yang tidak di-patch menimbulkan konsekuensi yang tidak dinginkan karena pihak yang tidak diketahui memanfaatkan kerentanan di dalamnya untuk mengakses beban kerja Anda.
- Anda melakukan patch pada lingkungan pengembangan secara otomatis tanpa memberi tahu pengembang. Anda menerima beberapa keluhan dari pengembang bahwa lingkungan mereka berhenti beroperasi sesuai dengan yang diharapkan.
- Anda belum menerapkan patch pada perangkat lunak komersial siap pakai di instans tetap. Ketika Anda mengalami masalah pada perangkat lunak dan menghubungi vendornya, Anda diberi tahu bahwa versi tersebut tidak didukung dan Anda harus melakukan patch pada tingkat tertentu untuk menerima bantuan.
- Patch yang baru-baru ini dirilis untuk perangkat lunak enkripsi yang Anda gunakan memiliki peningkatan kinerja yang signifikan. Sistem Anda yang tidak di-patch tetap memiliki masalah kinerja akibat tidak dilakukannya patching.
- Anda diberi tahu tentang kerentanan zero-day yang memerlukan perbaikan darurat dan Anda harus menerapkan patch pada semua lingkungan Anda secara manual.

Manfaat menjalankan praktik terbaik ini: Dengan menjalankan proses manajemen patch, termasuk kriteria Anda untuk patching dan metodologi untuk distribusi ke seluruh lingkungan Anda, Anda dapat menskalakan dan melaporkan tingkat patch. Ini memberikan jaminan seputar patching keamanan dan memastikan visibilitas yang jelas tentang status perbaikan yang diketahui sedang dilakukan. Hal ini mendorong adopsi fitur dan kemampuan yang diinginkan, penyingkiran masalah secara cepat, dan kepatuhan yang berkelanjutan terhadap tata kelola. Implementasikan sistem manajemen dan otomatisasi untuk mengurangi tingkat upaya untuk men-deploy patch dan mengurangi kesalahan yang disebabkan oleh proses manual.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

Lakukan patch pada sistem untuk menyelesaikan masalah, untuk mendapatkan fitur atau kemampuan yang diinginkan, dan untuk tetap patuh terhadap kebijakan tata kelola serta persyaratan dukungan vendor. Pada sistem tetap, deploy dengan rangkaian patch yang sesuai untuk mencapai hasil yang diinginkan. Otomatiskan mekanisme manajemen patch untuk mengurangi waktu yang

telah berlalu untuk melakukan patch, untuk mencegah kesalahan yang disebabkan oleh proses manual, dan mengurangi upaya dalam melakukan patch.

Langkah implementasi

Untuk Amazon EC2 Image Builder:

1. Menggunakan Amazon EC2 Image Builder, tentukan detail pipeline:
   a. Buat pipeline image dan beri nama
   b. Tentukan jadwal pipeline dan zona waktu
   c. Konfigurasikan dependensi apa pun
2. Pilih resep:
   a. Pilih resep yang sudah ada atau buat resep baru
   b. Pilih jenis image
   c. Beri nama dan versi resep Anda
   d. Pilih image dasar Anda
   e. Tambahkan komponen build dan tambahkan ke registri target
3. Opsional - tentukan konfigurasi infrastruktur Anda.
4. Opsional - tentukan pengaturan konfigurasi.
5. Tinjau pengaturan.
6. Pertahankan kebersihan resep secara teratur.

Untuk Systems Manager Patch Manager:

1. Buat dasar patch.
2. Pilih metode operasi patching.
3. Aktifkan pelaporan dan pemindaian kepatuhan.

Sumber daya

Praktik terbaik terkait:

- OPS06-BP04 Mengotomatiskan pengujian dan pengembalian (rollback)

Dokumen terkait:

- Apa itu Amazon EC2 Image Builder
- Buat pipeline image menggunakan Amazon EC2 Image Builder
- Buat pipeline image kontainer
- AWS Systems Manager Patch Manager
- Bekerja dengan Patch Manager
- Bekerja dengan laporan kepatuhan patch
- Alat Developer AWS

Video terkait:

- CI/CD untuk Aplikasi Nirserver di AWS
- Mendesain dengan Mempertimbangkan Operasional

Contoh terkait:

- Well-Architected Labs - Manajemen Inventaris dan Patch
- Tutorial AWS Systems Manager Patch Manager

OPS05-BP06 Membagikan standar desain

Bagikan praktik terbaik kepada seluruh tim untuk meningkatkan kesadaran dan memaksimalkan manfaat dari upaya pengembangan. Dokumentasikan dan jaga agar hal ini selalu mutakhir seiring evolusi arsitektur Anda. Jika standar bersama telah diterapkan di dalam organisasi Anda, tersedianya mekanisme sangat penting untuk meminta penambahan, perubahan, dan pengecualian terhadap standar. Tanpa opsi ini, standar akan menjadi penghambat inovasi.

Hasil yang diinginkan: Standar desain dibagikan ke semua tim dalam organisasi Anda. Standar ini didokumentasi dan dijaga agar selalu mutakhir seiring perkembangan praktik terbaik.

Antipola umum:

- Dua tim pengembangan masing-masing telah membuat layanan autentikasi pengguna. Pengguna Anda harus mempertahankan rangkaian kredensial terpisah untuk setiap bagian sistem yang ingin diakses.
- Setiap tim mengelola infrastruktur mereka sendiri. Persyaratan kepatuhan baru memaksakan perubahan pada infrastruktur Anda dan setiap tim mengimplementasikannya dengan cara yang berbeda.

Manfaat menjalankan praktik terbaik ini: Penggunaan standar bersama mendukung adopsi praktik terbaik dan memaksimalkan manfaat upaya pengembangan. Pendokumentasian dan pembaruan standar desain membuat organisasi Anda selalu mengikuti praktik terbaik dan persyaratan kepatuhan serta keamanan yang mutakhir.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Berbagi praktik terbaik yang ada, standar desain, daftar periksa, prosedur operasi, panduan, dan persyaratan tata kelola dengan semua tim. Miliki prosedur untuk meminta perubahan, penambahan, dan pengecualian standar desain untuk mendukung peningkatan dan inovasi. Buat tim mengetahui tentang konten yang dipublikasikan. Miliki mekanisme untuk menjaga agar standar desain selalu mutakhir seiring kemunculan praktik terbaik baru.

Contoh pelanggan

AnyCompany Retail memiliki tim arsitektur lintas fungsi yang membuat pola arsitektur perangkat lunak. Tim ini membangun arsitektur dengan kepatuhan dan tata kelola bawaan. Tim yang mengadopsi standar bersama ini mendapatkan manfaat dari memiliki kepatuhan dan tata kelola bawaan. Mereka dapat membangun di atas standar desain dengan cepat. Tim arsitektur mengadakan rapat setiap kuartal untuk mengevaluasi pola arsitektur dan memperbaruinya jika perlu.

Langkah implementasi

1. Identifikasikan tim lintas fungsi yang memegang kepemilikan atas pengembangan dan pembaruan standar desain. Tim ini harus bekerja sama dengan pemangku kepentingan di seluruh organisasi Anda untuk mengembangkan standar desain, standar operasi, daftar periksa, panduan, dan persyaratan tata kelola. Dokumentasikan standar desain dan bagikan dalam organisasi Anda.
   a. AWS Service Catalog dapat digunakan untuk membuat portofolio yang mewakili standar desain menggunakan infrastruktur sebagai kode. Anda dapat berbagi portofolio dengan semua akun.
2. Miliki mekanisme untuk menjaga agar standar desain selalu mutakhir seiring teridentifikasinya praktik terbaik baru.
3. Jika standar desain diterapkan secara terpusat, miliki proses untuk meminta perubahan, pembaruan, dan pengecualian.

Tingkat upaya untuk rencana implementasi: Sedang. Untuk mengembangkan proses untuk membuat dan berbagi standar desain mungkin diperlukan kerja sama dan koordinasi dengan para pemangku kepentingan di seluruh organisasi Anda.

Sumber daya

Praktik terbaik terkait:

- OPS01-BP03 Evaluasi persyaratan tata kelola - Persyaratan tata kelola memengaruhi standar desain.
- OPS01-BP04 Evaluasi persyaratan kepatuhan - Kepatuhan adalah input penting dalam membuat standar desain.
- OPS07-BP02 Memastikan peninjauan yang konsisten terkait kesiapan operasional - Daftar periksa kesiapan operasional merupakan mekanisme untuk mengimplementasikan standar desain ketika mendesain beban kerja Anda.
- OPS11-BP01 Miliki proses untuk peningkatan berkelanjutan - Memperbarui standar desain merupakan bagian dari peningkatan berkelanjutan.
- OPS11-BP04 Menjalankan manajemen pengetahuan - Sebagai bagian dari praktik manajemen pengetahuan Anda, dokumentasikan dan bagikan standar desain.

Dokumen terkait:

- Otomatiskan AWS Backup dengan AWS Service Catalog
- Akun AWS Service Catalog yang Ditingkatkan Pabrik
- Bagaimana Expedia Group membangun penawaran Basis Data sebagai Layanan (DBaaS) menggunakan AWS Service Catalog
- Mempertahankan visibilitas tentang penggunaan pola arsitektur cloud
- Menyederhanakan pembagian portofolio AWS Service Catalog Anda dalam pengaturan AWS Organizations

Video terkait:

- AWS Service Catalog – Memulai
- AWS re:Invent 2020: Mengelola portofolio AWS Service Catalog Anda layaknya ahli

Contoh terkait:

- Arsitektur Referensi AWS Service Catalog
- Lokakarya AWS Service Catalog

Layanan terkait:

- AWS Service Catalog

OPS05-BP07 Mengimplementasikan praktik untuk meningkatkan kualitas kode

Implementasikan praktik untuk meningkatkan kualitas kode dan meminimalkan kecacatan. Beberapa contohnya termasuk, pengembangan yang didorong pengujian, peninjauan kode, pengadopsian standar, dan pemrograman berpasangan. Sertakan praktik-praktik ini ke dalam integrasi berkelanjutan dan proses penyampaian hasil Anda.

Hasil yang diinginkan: Organisasi Anda menggunakan praktik terbaik seperti peninjauan kode atau pemrograman berpasangan untuk meningkatkan kualitas kode. Developer dan operator mengadopsi praktik terbaik dalam kualitas kode sebagai bagian dari siklus hidup pengembangan perangkat lunak.

Antipola umum:

- Anda mempercayakan kode ke cabang utama aplikasi tanpa peninjauan kode. Perubahan otomatis melakukan deployment ke produksi dan menyebabkan penghentian produksi.
- Aplikasi baru dikembangkan tanpa pengujian integrasi, unit, atau menyeluruh. Tidak ada cara untuk menguji aplikasi sebelum deployment.
- Tim Anda membuat perubahan manual pada produksi untuk mengatasi kecacatan. Perubahan tidak melalui proses pengujian atau peninjauan kode dan tidak ditangkap atau dicatat melalui proses penyampaian hasil dan integrasi berkelanjutan.

Manfaat menjalankan praktik terbaik ini: Dengan mengadopsi praktik untuk meningkatkan kualitas kode, Anda dapat membantu meminimalkan masalah yang terjadi di produksi. Kualitas kode akan meningkat dengan penggunaan praktik terbaik seperti pemrograman berpasangan dan peninjauan kode.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Implementasikan praktik untuk meningkatkan kualitas kode guna meminimalkan kecacatan sebelum dilakukan deployment terhadapnya. Gunakan praktik seperti pengembangan yang

didorong pengujian, peninjauan kode, dan pemrograman berpasangan untuk meningkatkan kualitas pengembangan Anda.

Contoh pelanggan

AnyCompany Retail mengadopsi beberapa praktik untuk meningkatkan kualitas kode. Mereka telah mengadopsi pengembangan yang didorong pengujian sebagai standar untuk menulis aplikasi. Untuk beberapa fitur baru, developer mereka memasangkan program menjadi satu selama sprint. Setiap permintaan penarikan akan melewati peninjauan kode oleh developer senior sebelum diintegrasikan dan dilakukan deployment.

Langkah implementasi

1. Adopsi praktik kualitas kode seperti pengembangan yang didorong pengujian, peninjauan kode, dan pemrograman berpasangan ke dalam proses penyampaian hasil dan integrasi berkelanjutan Anda. Gunakan teknik-teknik ini untuk meningkatkan kualitas perangkat lunak.
   a. Amazon CodeGuru Reviewer dapat memberikan rekomendasi pemrograman untuk kode Python dan Java menggunakan machine learning.
   b. Anda dapat membuat lingkungan pengembangan bersama dengan AWS Cloud9 di mana Anda dapat berkolaborasi dalam mengembangkan kode.

Tingkat upaya untuk rencana implementasi: Sedang. Ada banyak cara untuk mengimplementasikan praktik terbaik ini, tetapi membuat organisasi mau mengadopsinya mungkin merupakan hal yang sulit.

Sumber daya

Praktik terbaik terkait:

- OPS05-BP06 Membagikan standar desain - Anda dapat berbagi standar desain sebagai bagian dari praktik kualitas kode Anda.

Dokumen terkait:

- Panduan Perangkat Lunak Tangkas
- Pipeline CI/CD adalah pemandu utama rilisan saya
- Otomatiskan peninjauan kode dengan Amazon CodeGuru Reviewer
- Adopsi pendekatan pengembangan yang didorong pengujian

- Bagaimana DevFactory membangun aplikasi yang lebih baik dengan Amazon CodeGuru
- Tentang Pemrograman Berpasangan
- RENGA Inc. mengotomatiskan peninjauan kode dengan Amazon CodeGuru
- Seni Pengembangan Tangkas: Pengembangan yang Didorong Pengujian
- Mengapa peninjauan kode itu penting (dan sesungguhnya menghemat waktu!)

Video terkait:

- AWS re:Invent 2020: Peningkatan berkelanjutan kualitas kode dengan Amazon CodeGuru
- AWS Summit ANZ 2021 - Mendorong strategi yang mengutamakan pengujian dengan CDK dan pengembangan yang didorong pengujian

Layanan terkait:

- Amazon CodeGuru Reviewer
- Amazon CodeGuru Profiler
- AWS Cloud9

OPS05-BP08 Menggunakan beberapa lingkungan

Gunakan beberapa lingkungan untuk bereksperimen, mengembangkan, dan menguji beban kerja Anda. Gunakan tingkat kontrol berjenjang seiring lingkungan mendekati tahap produksi untuk mendapatkan keyakinan bahwa beban kerja Anda beroperasi sesuai keinginan ketika di-deploy.

Hasil yang diinginkan: Anda memiliki beberapa lingkungan yang mencerminkan kebutuhan kepatuhan dan tata kelola Anda. Anda menguji dan mempromosikan kode melalui lingkungan di jalur Anda menuju produksi.

Antipola umum:

- Anda sedang melakukan pengembangan di sebuah lingkungan pengembangan bersama dan developer lain menimpa perubahan kode Anda.
- Kontrol keamanan terbatas di lingkungan pengembangan bersama Anda melarang Anda melakukan eksperimen dengan layanan dan fitur baru.
- Anda melakukan pengujian beban pada sistem produksi Anda dan menyebabkan pemadaman untuk pengguna Anda.

- Kesalahan fatal yang menyebabkan hilangnya data terjadi di produksi. Di lingkungan produksi, Anda mencoba membuat ulang kondisi yang menyebabkan data hilang tersebut sehingga Anda dapat mengidentifikasi bagaimana hal tersebut terjadi dan mencegahnya agar tidak terjadi lagi. Untuk mencegah kejadian hilang data lainnya selama pengujian, Anda terpaksa menjadikan aplikasi tidak tersedia untuk pengguna.
- Anda mengoperasikan layanan multi-tenant dan tidak dapat mendukung permintaan lingkungan khusus yang diajukan pelanggan.
- Anda mungkin tidak selalu melakukan pengujian, tetapi ketika Anda menguji, Anda melakukannya di lingkungan produksi.
- Anda percaya bahwa dengan satu lingkungan tunggal, cakupan dampak perubahan hanya terjadi di dalam lingkungan tersebut.

Manfaat menjalankan praktik terbaik ini: Anda dapat mendukung beberapa lingkungan pengembangan, pengujian, dan produksi secara serentak tanpa menciptakan konflik antar developer atau komunitas pengguna.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Gunakan beberapa lingkungan dan sediakan lingkungan sandbox developer dengan kontrol minim untuk membantu eksperimen. Sediakan lingkungan pengembangan individu untuk membantu kerja secara paralel, sehingga ketangkasan pengembangan meningkat. Implementasikan kontrol yang lebih kuat di lingkungan ketika mendekati produksi agar developer dapat berinovasi. Gunakan infrastruktur sebagai kode dan sistem manajemen konfigurasi untuk men-deploy lingkungan yang dikonfigurasi sesuai dengan kontrol yang ada di dalam produksi guna memastikan sistem beroperasi sesuai keinginan saat di-deploy. Saat lingkungan tidak digunakan, nonaktifkan untuk menghindari biaya terkait sumber daya tidak terpakai (misalnya sistem pengembangan di malam hari dan di akhir pekan). Deploy lingkungan setara produksi saat melakukan pengujian beban untuk meningkatkan hasil yang valid.

Sumber daya

Dokumen terkait:

- Penjadwal Instans di AWS
- Apa itu AWS CloudFormation?

OPS05-BP09 Membuat perubahan yang sering, kecil, dan dapat dikembalikan

Gunakan perubahan yang sering, kecil, dan dapat dikembalikan untuk mengurangi cakupan perubahan. Ketika digunakan bersamaan dengan sistem manajemen perubahan, sistem manajemen konfigurasi, dan sistem build serta pengiriman, perubahan yang sering, kecil, dan dapat dikembalikan dapat mengurangi cakupan dan dampak perubahan. Hal ini menghasilkan pemecahan masalah yang lebih efektif dan remediasi yang lebih cepat dengan opsi untuk membatalkan perubahan.

Antipola umum:

- Anda men-deploy versi baru aplikasi Anda setiap tiga bulan sekali dengan periode perubahan yang mengharuskan layanan inti dinonaktifkan.
- Anda sering membuat perubahan pada skema basis data Anda tanpa melacak perubahan dalam sistem manajemen Anda.
- Anda melakukan pembaruan manual di tempat, menimpa instalasi dan konfigurasi yang ada, dan tidak memiliki rencana roll-back yang jelas.

Manfaat menjalankan praktik terbaik ini: Upaya pengembangan menjadi lebih cepat dengan sering men-deploy perubahan kecil. Ketika berukuran kecil, perubahan jauh lebih mudah diidentifikasi jika terdapat konsekuensi yang tidak diinginkan, serta lebih mudah untuk dikembalikan. Ketika perubahan dapat dikembalikan, risiko implementasi perubahan menjadi lebih kecil karena pemulihannya lebih mudah. Proses perubahan memiliki risiko yang lebih kecil dan dampak kegagalan perubahan menjadi berkurang.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Rendah

Panduan implementasi

Gunakan perubahan yang sering, kecil, dan dapat dikembalikan untuk mengurangi cakupan dan dampak perubahan. Hal ini memudahkan pemecahan masalah, membantu remediasi yang lebih cepat, dan menyediakan opsi untuk membatalkan perubahan. Hal ini juga meningkatkan rasio nilai yang dapat Anda berikan ke bisnis.

Sumber daya

Praktik terbaik terkait:

- OPS05-BP03 Menggunakan sistem manajemen konfigurasi

- OPS05-BP04 Menggunakan sistem manajemen build dan deployment
- OPS06-BP04 Mengotomatiskan pengujian dan pengembalian (rollback)

Dokumen terkait:

- Mengimplementasikan Layanan Mikro di AWS
- Layanan Mikro - Observabilitas

OPS05-BP10 Mengotomatiskan integrasi dan deployment sepenuhnya

Otomatiskan build, deployment, dan pengujian beban kerja. Hal ini mengurangi kesalahan yang disebabkan oleh proses manual, dan mengurangi upaya untuk melakukan deployment perubahan.

Terapkan metadata menggunakan Tag Sumber Daya dan AWS Resource Groups mengikuti strategi pemberian tag yang konsisten untuk membantu identifikasi sumber daya Anda. Berikan tag pada sumber daya Anda untuk pengaturan, akuntansi biaya, kontrol akses, dan penargetan pelaksanaan aktivitas operasi yang diotomatiskan.

Hasil yang diinginkan: Developer menggunakan alat untuk mengirimkan kode dan mencapai produksi. Developer tidak harus masuk ke dalam AWS Management Console untuk memberikan pembaruan. Terdapat jejak audit penuh untuk perubahan dan konfigurasi, sehingga memenuhi kebutuhan tata kelola dan kepatuhan. Proses dapat diulang dan distandardisasi di seluruh tim. Developer bebas memusatkan perhatian pada pengembangan dan pendorongan kode, sehingga meningkatkan produktivitas.

Antipola umum:

- Pada hari Jumat, Anda selesai menulis kode baru untuk cabang fitur Anda. Pada hari Senin, setelah menjalankan skrip pengujian kualitas kode dan setiap skrip pengujian unit, Anda mendaftarkan kode untuk rilisan terjadwal berikutnya.
- Anda ditugaskan untuk membuat kode perbaikan untuk sebuah masalah besar yang memengaruhi banyak pelanggan di tahap produksi. Setelah menguji perbaikan tersebut, Anda melakukan commit kode Anda dan mengirimkan manajemen perubahan melalui email untuk meminta persetujuan deployment ke produksi.
- Sebagai developer, Anda masuk ke AWS Management Console untuk membuat lingkungan pengembangan baru menggunakan metode dan sistem non-standar.

Manfaat menjalankan praktik terbaik ini: Dengan mengimplementasikan sistem manajemen build dan deployment otomatis, Anda mengurangi kesalahan yang disebabkan proses manual dan mengurangi upaya untuk melakukan deployment perubahan yang membantu anggota tim Anda berkonsentrasi menghadirkan nilai bisnis. Anda meningkatkan kecepatan pengiriman selama proses menuju produksi.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Rendah

Panduan implementasi

Anda menggunakan sistem manajemen build dan deployment untuk melacak dan mengimplementasikan perubahan, mengurangi kesalahan yang disebabkan oleh proses manual, dan mengurangi upaya. Otomatiskan sepenuhnya pipeline integrasi dan deployment dari check-in kode hingga build, pengujian, deployment, dan validasi. Hal ini mengurangi waktu tunggu, mendorong peningkatan frekuensi perubahan, mengurangi tingkat upaya, meningkatkan kecepatan masuk pasar, menghasilkan peningkatan produktivitas, dan meningkatkan keamanan kode Anda selama proses Anda menuju produksi.

Sumber daya

Praktik terbaik terkait:

- OPS05-BP03 Menggunakan sistem manajemen konfigurasi
- OPS05-BP04 Menggunakan sistem manajemen build dan deployment

Dokumen terkait:

- Apa itu AWS CodeBuild?
- Apa itu AWS CodeDeploy?

Video terkait:

- AWS re\:Invent 2022 - AWS Praktik terbaik Well-Architected untuk DevOps di AWS

Memitigasi risiko deployment

Adopsi pendekatan yang memberikan umpan balik cepat atas kualitas dan menyediakan pemulihan cepat dari perubahan yang tidak memiliki hasil yang tidak diinginkan. Menggunakan praktik-praktik ini memitigasi dampak masalah akibat deployment perubahan.

Desain beban kerja Anda harus menjelaskan bagaimana beban kerja akan diterapkan, diperbarui, dan dioperasikan. Anda akan mengimplementasikan praktik-praktik rekayasa yang selaras dengan pengurangan cacat serta perbaikan yang cepat dan aman.

Praktik terbaik

- OPS06-BP01 Antisipasikan perubahan yang tidak berhasil
- OPS06-BP02 Menguji deployment
- OPS06-BP03 Menggunakan strategi deployment yang aman
- OPS06-BP04 Mengotomatiskan pengujian dan pengembalian (rollback)

OPS06-BP01 Antisipasikan perubahan yang tidak berhasil

Rencanakan untuk kembali ke keadaan yang diketahui pasti baik, atau perbaiki di lingkungan produksi jika deployment menyebabkan hasil yang tidak diinginkan. Adanya kebijakan untuk menetapkan rencana semacam ini bermanfaat bagi semua tim dalam mengembangkan strategi untuk pulih dari perubahan yang gagal. Beberapa contoh strategi adalah langkah deployment dan rollback, kebijakan perubahan, penanda fitur, pemisahan lalu lintas, dan pergeseran lalu lintas. Rilis tunggal dapat mencakup beberapa perubahan komponen yang terkait. Strategi harus memberikan kemampuan untuk bertahan atau pulih dari kegagalan perubahan komponen apa pun.

Hasil yang diinginkan: Anda telah menyiapkan rencana pemulihan yang mendetail untuk perubahan Anda apabila perubahan tersebut tidak berhasil. Selain itu, Anda telah mengurangi ukuran rilis untuk meminimalkan dampak potensial pada komponen beban kerja lainnya. Hasilnya, Anda telah mengurangi dampak bisnis Anda dengan mempersingkat potensi waktu henti yang diakibatkan oleh perubahan yang gagal dan meningkatkan fleksibilitas serta efisiensi waktu pemulihan.

Antipola umum:

- Anda melakukan deployment dan aplikasi Anda menjadi tidak stabil tetapi tampaknya ada pengguna aktif di sistem. Anda harus memutuskan apakah akan mengembalikan perubahan yang akan berdampak pada pengguna aktif atau menunggu untuk mengembalikan perubahan karena tahu bagaimana pun juga pengguna dapat terkena dampaknya.

- Setelah membuat perubahan rutin, lingkungan baru Anda dapat diakses tetapi salah satu subnet Anda menjadi tidak dapat dijangkau. Anda harus memutuskan apakah akan mengembalikan semuanya atau mencoba memperbaiki subnet yang tidak dapat diakses tersebut. Sementara Anda sedang memutuskan hal ini, subnet tersebut tetap tidak dapat dijangkau.
- Sistem Anda tidak dirancang dapat diperbarui dengan rilis-rilis yang lebih kecil. Akibatnya, Anda mengalami kesulitan dalam membatalkan perubahan massal tersebut selama deployment yang gagal.
- Anda tidak menggunakan infrastruktur sebagai kode (IaC) dan Anda melakukan pembaruan manual pada infrastruktur Anda sehingga mengakibatkan konfigurasi yang tidak diinginkan. Anda tidak dapat melacak dan membatalkan perubahan manual secara efektif.
- Karena Anda belum mengukur peningkatan frekuensi deployment Anda, tim Anda kesulitan mengurangi ukuran perubahan mereka dan meningkatkan rencana rollback mereka untuk setiap perubahan, yang berimbas pada risiko yang lebih besar dan tingkat kegagalan yang meningkat.
- Anda tidak mengukur total durasi pemadaman yang disebabkan oleh perubahan yang tidak berhasil. Tim Anda tidak dapat memprioritaskan dan meningkatkan proses deployment serta efektivitas rencana pemulihannya.

Manfaat menjalankan praktik terbaik ini: Memiliki rencana untuk pulih dari perubahan yang gagal meminimalkan rata-rata waktu untuk pulih (MTTR) dan mengurangi dampak bisnis Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Kebijakan dan praktik yang konsisten serta terdokumentasi yang diadopsi oleh tim rilis memungkinkan organisasi untuk merencanakan apa yang harus terjadi apabila terjadi kegagalan perubahan. Kebijakan harus memungkinkan perbaikan maju (fix forward) dalam keadaan tertentu. Dalam situasi apa pun, rencana perbaikan maju atau rollback harus didokumentasikan dan diuji dengan baik sebelum deployment ke produksi langsung sehingga waktu yang diperlukan untuk mengembalikan perubahan dapat diminimalkan.

Langkah implementasi

1. Dokumentasikan kebijakan yang mengharuskan tim memiliki rencana efektif untuk mengembalikan perubahan dalam periode tertentu.
   a. Kebijakan harus menentukan kapan situasi perbaikan maju diperbolehkan.
   b. Rencana rollback yang terdokumentasi harus dapat diakses oleh semua pihak yang terlibat.

   c. Tentukan persyaratan untuk rollback (misalnya, ketika ternyata ada perubahan tidak sah yang telah di-deploy).
2. Analisis tingkat dampak semua perubahan yang berkaitan dengan setiap komponen beban kerja.
   a. Biarkan perubahan berulang untuk distandardisasi, dijadikan templat, dan diotorisasi di awal jika perubahan tersebut mengikuti alur kerja konsisten yang memberlakukan kebijakan perubahan.
   b. Kurangi potensi dampak perubahan apa pun dengan menjadikan ukuran perubahan lebih kecil sehingga pemulihan membutuhkan waktu yang lebih singkat dan menyebabkan lebih sedikit dampak bisnis.
   c. Pastikan prosedur rollback mengembalikan kode ke keadaan yang pasti baik untuk menghindari insiden jika memungkinkan.
3. Integrasikan alat dan alur kerja untuk menegakkan kebijakan Anda secara terprogram.
4. Buat agar data tentang perubahan dapat dilihat oleh pemilik beban kerja lain untuk meningkatkan kecepatan diagnosis perubahan yang gagal yang tidak dapat dibatalkan.
   a. Ukur keberhasilan praktik ini menggunakan data perubahan yang terlihat dan identifikasi peningkatan iteratif.
5. Gunakan alat pemantauan untuk memverifikasi keberhasilan atau kegagalan deployment untuk mempercepat pengambilan keputusan saat melakukan rollback.
6. Ukur durasi pemadaman Anda selama perubahan yang gagal untuk terus meningkatkan kualitas rencana pemulihan Anda.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik terbaik terkait:

- OPS06-BP04 Mengotomatiskan pengujian dan pengembalian (rollback)

Dokumen terkait:

- AWS Builders Library | Memastikan Keamanan Rollback Selama Deployment
- AWS Laporan Resmi | Manajemen Perubahan di Cloud

Video terkait:

- re:Invent 2019 | Pendekatan Amazon untuk deployment ketersediaan tinggi

OPS06-BP02 Menguji deployment

Uji prosedur rilis dalam tahap praproduksi dengan menggunakan konfigurasi deployment, kontrol keamanan, langkah, dan prosedur yang sama seperti dalam tahap produksi. Lakukan validasi bahwa semua langkah yang di-deploy selesai sesuai harapan, seperti dengan memeriksa file, konfigurasi, dan layanan. Uji lebih lanjut semua perubahan dengan pengujian fungsional, integrasi, dan beban, beserta pemantauan apa pun seperti pemeriksaan kondisi. Dengan melakukan pengujian ini, Anda dapat mengidentifikasi masalah deployment lebih awal dengan peluang untuk merencanakan dan menanggulanginya sebelum produksi.

Anda dapat membuat lingkungan paralel sementara untuk menguji setiap perubahan. Otomatiskan deployment lingkungan pengujian menggunakan infrastruktur sebagai kode (IaC) untuk membantu mengurangi jumlah pekerjaan yang terlibat dan memastikan stabilitas, konsistensi, dan pengiriman fitur yang lebih cepat.

Hasil yang diinginkan: Organisasi Anda mengadopsi budaya pengembangan berbasis pengujian yang mencakup pengujian deployment. Ini memastikan tim fokus menghadirkan nilai bisnis, bukan mengelola rilis. Tim terlibat sejak dini setelah identifikasi risiko deployment untuk menentukan arah mitigasi yang tepat.

Antipola umum:

- Selama rilis produksi, deployment yang belum teruji sering menyebabkan masalah yang memerlukan penyelesaian dan eskalasi.
- Rilis Anda berisi infrastruktur sebagai kode (IaC) yang memperbarui sumber daya yang ada. Anda tidak yakin apakah IaC berjalan dengan sukses atau menyebabkan dampak pada sumber daya.
- Anda men-deploy sebuah fitur baru ke aplikasi Anda. Fitur tersebut tidak berfungsi sesuai keinginan dan masalah ini baru dapat diketahui setelah dilaporkan oleh pengguna yang terdampak.
- Anda memperbarui sertifikat Anda. Anda tidak sengaja menginstal sertifikat ke komponen yang salah, yang akhirnya tidak terdeteksi dan berdampak pada pengunjung situs web karena koneksi yang aman ke situs web tidak dapat dibuat.

Manfaat menjalankan praktik terbaik ini: Pengujian ekstensif selama tahap praproduksi dalam prosedur deployment serta perubahan yang dimunculkannya dapat meminimalkan potensi dampak terhadap produksi yang disebabkan oleh langkah-langkah deployment. Hal ini meningkatkan

kepercayaan diri selama rilis produksi dan meminimalkan dukungan operasional tanpa memperlambat penyampaian perubahan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Menguji proses deployment sama pentingnya dengan menguji perubahan yang dihasilkan dari deployment Anda. Hal ini dapat dicapai dengan menguji langkah-langkah deployment Anda di lingkungan praproduksi yang semaksimal mungkin mencerminkan produksi. Masalah umum, seperti langkah deployment yang tidak lengkap atau salah, atau kesalahan konfigurasi, dapat terdeteksi sebelum masuk ke tahap produksi. Selain itu, Anda dapat menguji langkah-langkah pemulihan Anda.

Contoh pelanggan

Sebagai bagian dari pipeline integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD), AnyCompany Retail melakukan langkah-langkah yang ditentukan yang diperlukan untuk merilis pembaruan infrastruktur dan perangkat lunak bagi pelanggannya di dalam lingkungan mirip produksi. Pipeline tersebut terdiri dari prapemeriksaan untuk mendeteksi penyimpangan (mendeteksi perubahan pada sumber daya yang dilakukan di luar IaC Anda) di dalam sumber daya sebelum deployment, serta memvalidasi tindakan yang dilakukan IaC setelah inisiasi. Tahap ini memvalidasi langkah-langkah deployment, seperti memverifikasi bahwa file dan konfigurasi tertentu sudah siap dan layanan sudah dalam status berjalan serta merespons dengan benar pemeriksaan kondisi pada host lokal sebelum didaftarkan ulang dengan penyeimbang beban. Selain itu, semua perubahan menandai sejumlah pengujian otomatis, seperti pengujian fungsional, keamanan, regresi, integrasi, dan beban.

Langkah implementasi

1. Lakukan pemeriksaan prainstal pada produksi untuk mencerminkan lingkungan praproduksi.
   a. Gunakan deteksi penyimpangan untuk mendeteksi apabila sumber daya telah diubah di luar AWS CloudFormation.
   b. Gunakan set perubahan untuk memvalidasi bahwa maksud pembaruan tumpukan sesuai dengan tindakan yang dilakukan oleh AWS CloudFormation saat rangkaian perubahan dimulai.
2. Ini memicu langkah persetujuan manual di AWS CodePipeline untuk mengotorisasi deployment ke lingkungan praproduksi.
3. Gunakan konfigurasi deployment seperti file AWS CodeDeploy AppSpec untuk menentukan langkah deployment dan validasi.

4. Jika perlu, integrasikan AWS CodeDeploy dengan layanan AWS lain atau integrasikan AWS CodeDeploy dengan produk dan layanan partner.
5. Pantau deployment menggunakan Amazon CloudWatch, AWS CloudTrail, dan notifikasi peristiwa Amazon SNS.
6. Lakukan pengujian otomatis pasca-deployment, termasuk pengujian fungsional, keamanan, regresi, integrasi, dan beban.
7. Pecahkan masalah deployment.
8. Validasi yang berhasil dari langkah-langkah sebelumnya seharusnya menginisiasi alur kerja persetujuan manual untuk memberikan otorisasi deployment ke produksi.

Tingkat upaya untuk rencana implementasi: Tinggi

Sumber daya

Praktik terbaik terkait:

- OPS05-BP02 Menguji dan memvalidasi perubahan

Dokumen terkait:

- AWS Builders' Library | Mengotomatiskan deployment hands-off yang aman | Menguji Deployment
- AWS Laporan Resmi | Mempraktikkan Integrasi Berkelanjutan dan Pengiriman Berkelanjutan di AWS
- Kisah Apollo - Mesin Deployment Amazon
- Cara menguji dan melakukan debug AWS CodeDeploy secara lokal sebelum mengirimkan kode Anda
- Mengintegrasikan Pengujian Konektivitas Jaringan dengan Deployment Infrastruktur

Video terkait:

- re:Invent 2020 | Menguji perangkat lunak dan sistem di Amazon

Contoh terkait:

- Tutorial | Melakukan Deployment dan layanan Amazon ECS dengan uji validasi

OPS06-BP03 Menggunakan strategi deployment yang aman

Peluncuran produksi yang aman mengontrol aliran perubahan yang bermanfaat dengan tujuan untuk meminimalkan dampak yang dirasakan oleh pelanggan dari perubahan tersebut. Kontrol keselamatan menyediakan mekanisme inspeksi untuk memvalidasi hasil yang diinginkan dan membatasi ruang lingkup dampak dari cacat apa pun yang disebabkan oleh perubahan atau dari kegagalan deployment. Peluncuran yang aman dapat mencakup strategi seperti feature-flag, one-box, bergulir (rilis canary), immutable, pemisahan lalu lintas, dan deployment blue/green.

Hasil yang diinginkan: Organisasi Anda menggunakan sistem integrasi berkelanjutan pengiriman berkelanjutan (CI/CD) yang menyediakan kemampuan untuk mengotomatiskan peluncuran yang aman. Tim diharuskan menggunakan strategi peluncuran aman yang tepat.

Antipola umum:

- Anda melakukan deployment perubahan yang tidak berhasil ke seluruh produksi sekaligus. Akibatnya, semua pelanggan merasakan dampaknya secara bersamaan.
- Cacat akibat deployment serentak ke semua sistem memerlukan rilis darurat. Diperlukan waktu beberapa hari untuk memperbaikinya untuk semua pelanggan.
- Untuk mengelola rilis produksi diperlukan perencanaan dan partisipasi beberapa tim. Hal ini menghambat kemampuan Anda untuk sering memperbarui fitur bagi pelanggan Anda.
- Anda melakukan deployment yang dapat diubah dengan memodifikasi sistem yang sudah ada. Setelah mengetahui bahwa perubahan tidak berhasil, Anda terpaksa memodifikasi sistem lagi untuk memulihkan versi yang lama sehingga memperpanjang waktu pemulihan Anda.

Manfaat menjalankan praktik terbaik ini: Deployment otomatis menyeimbangkan kecepatan peluncuran dengan menghadirkan perubahan yang bermanfaat secara konsisten kepada pelanggan. Pembatasan dampak dapat mencegah kegagalan deployment yang mahal dan memaksimalkan kemampuan tim untuk merespons kegagalan secara efisien.

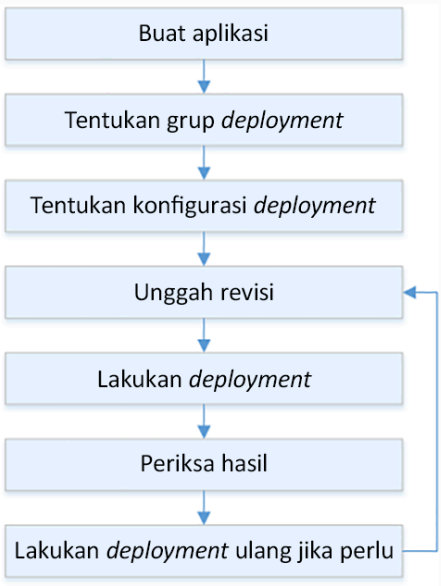
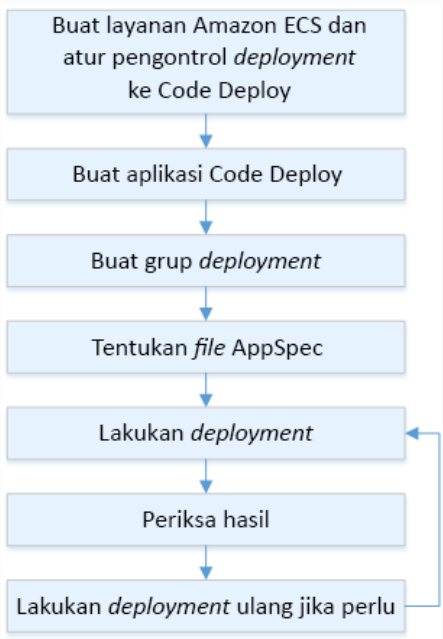
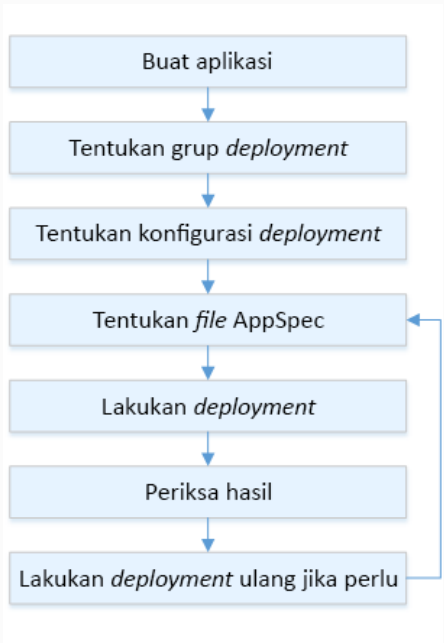
Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Kegagalan pengiriman berkelanjutan dapat menyebabkan berkurangnya ketersediaan layanan dan buruknya pengalaman pelanggan. Untuk memaksimalkan tingkat keberhasilan deployment, terapkan kontrol keamanan dalam proses rilis menyeluruh (end-to-end) untuk meminimalkan kesalahan deployment, dengan tujuan mencapai nol kegagalan deployment.

Contoh pelanggan

AnyCompany Retail memiliki misi untuk mencapai deployment dengan waktu henti yang minim hingga nol, yang berarti tidak ada dampak yang dirasakan oleh penggunanya selama deployment. Untuk mencapainya, perusahaan telah membuat pola deployment (lihat diagram alur kerja berikut), seperti deployment blue/green dan bergulir (rolling). Semua tim mengadopsi satu atau beberapa pola tersebut di dalam pipeline CI/CD mereka.

Alur kerja CodeDeploy untuk Amazon EC2	Alur kerja CodeDeploy untuk Amazon ECS	Alur kerja CodeDeploy untuk Lambda
		

Langkah implementasi

1. Gunakan alur kerja persetujuan untuk memulai urutan langkah peluncuran produksi setelah promosi ke produksi.
2. Gunakan sistem deployment otomatis seperti AWS CodeDeploy. Opsi deployment AWS CodeDeploy meliputi deployment pengganti untuk EC2/On-Premise dan deployment blue/green untuk EC2/On-Premise, AWS Lambda, dan Amazon ECS (lihat diagram alur kerja sebelumnya).
   a. Jika perlu, integrasikan AWS CodeDeploy dengan layanan AWS lain atau integrasikan AWS CodeDeploy dengan produk dan layanan partner.
3. Gunakan deployment blue/green untuk basis data seperti Amazon Aurora dan Amazon RDS.

4. Pantau deployment menggunakan Amazon CloudWatch, AWS CloudTrail, dan notifikasi peristiwa Amazon SNS.
5. Lakukan pengujian otomatis pasca-deployment termasuk pengujian fungsional, keamanan, regresi, integrasi, dan beban apa pun.
6. Pecahkan masalah deployment.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik terbaik terkait:

- OPS05-BP02 Menguji dan memvalidasi perubahan
- OPS05-BP09 Membuat perubahan yang sering, kecil, dan dapat dikembalikan
- OPS05-BP10 Mengotomatiskan integrasi dan deployment sepenuhnya

Dokumen terkait:

- AWS Builders Library | Mengotomatiskan deployment hands-off yang aman | Deployment produksi
- AWS Builders Library | Pipeline CI/CD saya adalah kapten rilis saya | Rilis produksi otomatis yang aman
- Laporan Resmi AWS | Mempraktikkan Integrasi Berkelanjutan dan Pengiriman Berkelanjutan di AWS | Metode deployment
- AWS CodeDeploy Panduan Pengguna
- Mulai konfigurasi deployment di AWS CodeDeploy
- Konfigurasikan canary API Gateway untuk meluncurkan deployment
- Jenis Deployment Amazon ECS
- Deployment Blue/Green Terkelola Penuh di Amazon Aurora dan Amazon RDS
- Deployment Blue/Green dengan AWS Elastic Beanstalk

Video terkait:

- re:Invent 2020 | Hands-off: Mengotomatiskan pipeline pengiriman berkelanjutan di Amazon
- re:Invent 2019 | Pendekatan deployment ketersediaan tinggi Amazon

Contoh terkait:

- Coba Sampel Deployment Blue/Green di AWS CodeDeploy
- Lokakarya | Membangun pipeline CI/CD untuk deployment canary Lambda menggunakan AWS CDK
- Lokakarya | Deployment Blue/Green dan Canary untuk EKS dan ECS
- Lokakarya | Membangun Pipeline CI/CD Lintas Akun

OPS06-BP04 Mengotomatiskan pengujian dan pengembalian (rollback)

Untuk meningkatkan kecepatan, keandalan, dan keyakinan pada proses deployment Anda, miliki strategi untuk kemampuan pengujian dan rollback otomatis di lingkungan praproduksi dan produksi. Otomatiskan pengujian saat melakukan deployment ke produksi untuk menyimulasikan interaksi manusia dan sistem yang memverifikasi perubahan yang sedang di-deploy. Otomatiskan rollback untuk kembali ke keadaan pasti baik sebelumnya dengan cepat. Rollback harus dimulai secara otomatis pada kondisi yang telah ditentukan di awal seperti ketika hasil perubahan yang Anda inginkan tidak tercapai atau ketika pengujian otomatis mengalami kegagalan. Mengotomatiskan kedua aktivitas ini dapat memperbaiki tingkat keberhasilan untuk deployment Anda, meminimalkan waktu pemulihan, dan mengurangi potensi dampak terhadap bisnis.

Hasil yang diinginkan: Strategi pengujian dan rollback otomatis Anda diintegrasikan ke dalam pipeline integrasi berkelanjutan dan pengiriman berkelanjutan (CI/CD) Anda. Pemantauan Anda dapat memvalidasi berdasarkan kriteria keberhasilan Anda dan memulai rollback otomatis setelah kegagalan. Hal ini meminimalkan dampak apa pun terhadap pelanggan dan pengguna akhir. Misalnya, ketika semua hasil pengujian telah terpenuhi, Anda meneruskan kode Anda ke lingkungan produksi tempat pengujian regresi otomatis dimulai, dengan memanfaatkan kasus pengujian yang sama. Jika hasil pengujian regresi tidak sesuai dengan harapan, maka rollback otomatis akan dimulai dalam alur kerja pipeline.

Antipola umum:

- Sistem Anda tidak dirancang dapat diperbarui dengan rilis-rilis yang lebih kecil. Akibatnya, Anda mengalami kesulitan dalam membatalkan perubahan massal tersebut selama deployment yang gagal.
- Proses deployment Anda terdiri dari serangkaian langkah manual. Setelah melakukan deployment perubahan ke beban kerja, Anda memulai pengujian pasca-deployment. Setelah pengujian, Anda menyadari bahwa beban kerja Anda tidak dapat dioperasikan dan koneksi pelanggan

terputus. Kemudian Anda mulai melakukan rollback ke versi sebelumnya. Semua langkah manual ini menghambat pemulihan sistem secara keseluruhan dan menyebabkan dampak yang berkepanjangan terhadap pelanggan Anda.

- Anda menghabiskan waktu mengembangkan kasus pengujian otomatis untuk fungsionalitas yang tidak sering digunakan dalam aplikasi Anda, sehingga memperkecil laba atas investasi dalam kemampuan pengujian otomatis Anda.
- Rilis Anda terdiri dari aplikasi, infrastruktur, patch, dan pembaruan konfigurasi yang tidak bergantung satu sama lain. Namun, Anda memiliki satu pipeline CI/CD yang mengirimkan semua perubahan sekaligus. Kegagalan pada satu komponen memaksa Anda untuk mengembalikan semua perubahan, menjadikan rollback Anda kompleks dan tidak efisien.
- Tim Anda menyelesaikan tugas pengodean dalam sprint satu dan memulai tugas sprint dua, tetapi rencana Anda tidak menyertakan pengujian sampai sprint tiga. Akibatnya, pengujian otomatis mengungkap cacat dari sprint satu yang harus diselesaikan sebelum pengujian kiriman sprint dua dapat dimulai dan seluruh rilis menjadi tertunda, sehingga menurunkan nilai pengujian otomatis Anda.
- Kasus pengujian regresi otomatis Anda untuk rilis produksi sudah selesai, tetapi Anda tidak memantau kondisi beban kerja. Karena Anda tidak memiliki visibilitas apakah layanan telah dimulai ulang atau belum, Anda tidak yakin apakah rollback diperlukan atau rollback sudah terjadi.

Manfaat menjalankan praktik terbaik ini: Pengujian otomatis meningkatkan transparansi proses pengujian Anda dan kemampuan Anda untuk mencakup lebih banyak fitur dalam periode waktu yang lebih singkat. Dengan menguji dan memvalidasi perubahan dalam produksi, Anda dapat segera mengidentifikasi masalah. Peningkatan konsistensi dengan alat pengujian otomatis memungkinkan deteksi cacat yang lebih baik. Dengan melakukan rollback otomatis ke versi sebelumnya, dampak pada pelanggan diminimalkan. Rollback otomatis pada akhirnya memunculkan keyakinan yang lebih tinggi pada kemampuan deployment Anda dengan mengurangi dampak bisnis. Secara keseluruhan, kemampuan ini mengurangi waktu pengiriman sekaligus memastikan kualitas.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Otomatiskan pengujian lingkungan yang di-deploy untuk mengonfirmasi hasil yang diinginkan dengan lebih cepat. Otomatiskan pengembalian ke keadaan yang diketahui baik sebelumnya ketika hasil yang ditetapkan di awal tidak tercapai, untuk mempersingkat waktu pemulihan dan mengurangi kesalahan yang disebabkan oleh proses manual. Integrasikan alat pengujian dengan alur kerja

pipeline Anda untuk menguji dan meminimalkan input manual secara konsisten. Prioritaskan otomatisasi kasus pengujian, seperti kasus pengujian yang mengurangi risiko terbesar dan perlu sering diuji dengan setiap perubahan. Selain itu, otomatiskan rollback berdasarkan kondisi tertentu yang telah ditentukan di awal dalam rencana pengujian Anda.

Langkah implementasi

1. Bangun siklus pengujian untuk siklus hidup pengembangan Anda yang menentukan setiap tahap proses pengujian mulai dari perencanaan persyaratan hingga pengembangan kasus pengujian, konfigurasi alat, pengujian otomatis, dan penutupan kasus pengujian.
   a. Buat pendekatan pengujian khusus beban kerja dari strategi pengujian Anda secara keseluruhan.
   b. Pertimbangkan strategi pengujian berkelanjutan jika diperlukan di seluruh siklus pengembangan.
2. Pilih alat otomatis untuk pengujian dan rollback berdasarkan kebutuhan bisnis dan investasi pipeline Anda.
3. Tentukan kasus pengujian mana yang ingin Anda otomatisasi dan mana yang harus dilakukan secara manual. Anda dapat menentukannya berdasarkan prioritas nilai bisnis dari fitur yang sedang diuji. Selaraskan semua anggota tim dengan rencana ini dan pastikan akuntabilitas untuk melakukan pengujian manual.
   a. Terapkan kemampuan pengujian otomatis ke kasus pengujian tertentu yang cocok untuk otomatisasi, seperti kasus berulang atau yang sering dijalankan, kasus yang memerlukan tugas berulang, atau kasus yang diperlukan di beberapa konfigurasi.
   b. Tentukan skrip otomatisasi pengujian serta kriteria keberhasilan di dalam alat otomatisasi sehingga otomatisasi alur kerja yang berkelanjutan dapat dimulai ketika ada kasus tertentu yang gagal.
   c. Tentukan kriteria kegagalan khusus untuk rollback otomatis.
4. Prioritaskan otomatisasi pengujian untuk mendorong hasil yang konsisten dengan pengembangan kasus pengujian menyeluruh di mana kompleksitas dan interaksi manusia memiliki risiko kegagalan yang lebih tinggi.
5. Integrasikan pengujian otomatis dan alat rollback Anda ke dalam pipeline CI/CD Anda.
   a. Kembangkan kriteria keberhasilan yang jelas untuk perubahan Anda.
   b. Pantau dan amati untuk mendeteksi kriteria ini dan secara otomatis membalikkan perubahan ketika kriteria rollback tertentu terpenuhi.
6. Lakukan berbagai jenis pengujian produksi otomatis, seperti:

   a. Pengujian A/B untuk menunjukkan hasil yang dibandingkan dengan versi saat ini antara dua kelompok pengujian pengguna.
   b. Pengujian canary yang memungkinkan Anda untuk meluncurkan perubahan Anda pada subset pengguna sebelum merilisnya ke semua pengguna.
   c. Pengujian penandaan fitur (feature flag) yang memungkinkan satu per satu fitur dari versi baru untuk ditandai atau dihapus tandanya dari luar aplikasi sehingga setiap fitur baru dapat divalidasi satu per satu.
   d. Pengujian regresi untuk memverifikasi fungsionalitas baru dengan komponen yang saling terkait.
7. Pantau aspek operasional aplikasi, transaksi, dan interaksi dengan aplikasi dan komponen lain. Kembangkan laporan untuk menunjukkan keberhasilan perubahan berdasarkan beban kerja sehingga Anda dapat mengidentifikasi bagian otomatisasi dan alur kerja apa yang dapat dioptimalkan lebih lanjut.
   a. Kembangkan laporan hasil pengujian yang membantu Anda mengambil keputusan cepat terkait apakah prosedur rollback perlu dimulai.
   b. Terapkan strategi yang memungkinkan rollback otomatis berdasarkan kondisi kegagalan yang telah ditentukan di awal yang dihasilkan dari satu atau beberapa metode pengujian Anda.
8. Kembangkan kasus pengujian otomatis untuk memungkinkan penggunaan ulang di seluruh perubahan berulang di masa mendatang.

Tingkat upaya untuk rencana implementasi: Sedang

Sumber daya

Praktik terbaik terkait:

- OPS06-BP01 Antisipasikan perubahan yang tidak berhasil
- OPS06-BP02 Menguji deployment

Dokumen terkait:

- AWS Builders Library | Memastikan keamanan rollback selama deployment
- Deployment ulang dan membatalkan deployment dengan AWS CodeDeploy
- 8 praktik terbaik saat mengotomatiskan deployment Anda dengan AWS CloudFormation

Contoh terkait:

- Pengujian UI nirserver menggunakan Selenium, AWS Lambda, AWS Fargate (Fargate), dan Alat Developer AWS

Video terkait:

- re:Invent 2020 | Hands-off: Mengotomatiskan pipeline pengiriman berkelanjutan di Amazon
- re:Invent 2019 | Pendekatan deployment ketersediaan tinggi Amazon

Kesiapan operasional dan manajemen perubahan

Evaluasi kesiapan operasional beban kerja, proses, prosedur, dan personel Anda untuk memahami risiko operasional terkait beban kerja Anda. Kelola alur perubahan ke dalam lingkungan Anda.

Anda harus menggunakan proses yang konsisten (termasuk daftar periksa manual dan otomatis) untuk mengetahui saat Anda siap untuk mengoperasionalkan beban kerja Anda atau untuk melakukan perubahan. Hal ini juga akan membantu Anda menemukan area mana pun yang Anda butuhkan untuk membuat rencana untuk ditangani. Anda akan memiliki runbook yang mendokumentasikan aktivitas rutin serta pedoman yang memandu proses penyelesaian masalah Anda. Gunakan mekanisme untuk mengelola perubahan yang mendukung penghadiran nilai bisnis dan bantu mitigasi risiko terkait perubahan.

Praktik terbaik

- OPS07-BP01 Memastikan kemampuan personel
- OPS07-BP02 Memastikan peninjauan yang konsisten terkait kesiapan operasional
- OPS07-BP03 Menggunakan runbook untuk menjalankan prosedur
- OPS07-BP04 Menggunakan buku panduan untuk menyelidiki masalah
- OPS07-BP05 Membuat keputusan yang tepat untuk melakukan deployment sistem dan perubahan
- OPS07-BP06 Mengaktifkan rencana dukungan untuk beban kerja produksi

OPS07-BP01 Memastikan kemampuan personel

Miliki mekanisme untuk memvalidasi bahwa Anda memiliki jumlah personel terlatih yang sesuai untuk mendukung beban kerja. Mereka harus diberi pelatihan tentang platform dan layanan yang membentuk beban kerja Anda. Berikan kepada mereka pengetahuan yang diperlukan untuk

mengoperasikan beban kerja. Anda harus memiliki cukup banyak personel terlatih untuk mendukung pengoperasian normal beban kerja dan menyelesaikan masalah terkait insiden yang muncul. Miliki cukup banyak personel sehingga Anda dapat melakukan rotasi untuk personel yang siap tugas mendadak dan personel yang liburan guna menghindari personel menjadi terlalu lelah.

Hasil yang diinginkan:

- Ada cukup banyak personel terlatih untuk mendukung beban kerja pada saat beban kerja tersedia.
- Anda memberikan pelatihan untuk personel tentang perangkat lunak dan layanan yang membentuk beban kerja Anda.

Antipola umum:

- Melakukan deployment beban kerja tanpa anggota tim yang terlatih untuk mengoperasikan platform dan layanan yang digunakan.
- Tidak memiliki cukup banyak personel untuk mendukung rotasi personel yang siap tugas mendadak atau personel yang sedang libur.

Manfaat menjalankan praktik terbaik ini:

- Memiliki anggota tim yang terampil memungkinkan dukungan yang efektif untuk beban kerja.
- Dengan cukup banyak anggota tim, Anda dapat mendukung beban kerja dan rotasi personel yang siap tugas mendadak sekaligus mengurangi risiko personel terlalu lelah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Validasikan bahwa terdapat personel yang terlatih dengan memadai untuk mendukung beban kerja. Pastikan Anda memiliki jumlah anggota tim yang cukup untuk menangani aktivitas operasional normal, termasuk rotasi personel siap tugas mendadak.

Contoh pelanggan

AnyCompany Retail memastikan tim yang mendukung beban kerja memiliki staf yang terlatih dalam jumlah yang sesuai. Mereka memiliki cukup banyak rekayasawan untuk mendukung rotasi personel yang siap tugas mendadak. Personel mendapatkan pelatihan tentang perangkat lunak dan platform yang merupakan dasar pembangunan beban kerja dan mereka didorong untuk mendapatkan

sertifikasi. Ada cukup banyak personel sehingga orang dapat mengambil cuti sambil tetap ada dukungan untuk beban kerja dan rotasi personel yang siap tugas mendadak.

Langkah implementasi

1. Tetapkan jumlah personel yang memadai untuk mengoperasikan dan mendukung beban kerja Anda, termasuk tugas mendadak.
2. Latih personel Anda tentang perangkat lunak dan platform yang membentuk beban kerja Anda.
   a. AWS Training and Certification memiliki pustaka kursus tentang AWS. Kursus-kursus ini disediakan gratis dan berbayar, baik secara online maupun tatap muka.
   b. AWS melakukan hosting acara dan webinar di mana Anda belajar dari para ahli AWS.
3. Evaluasi ukuran dan keterampilan tim secara teratur seiring perubahan kondisi pengoperasian dan beban kerja. Sesuaikan ukuran dan keterampilan tim agar memenuhi persyaratan operasional.

Tingkat upaya untuk rencana implementasi: Tinggi. Mempekerjakan dan melatih tim untuk mendukung beban kerja dapat memerlukan upaya yang cukup besar tetapi memiliki manfaat jangka panjang yang substansial.

Sumber daya

Praktik Terbaik Terkait:

- OPS11-BP04 Menjalankan manajemen pengetahuan - Anggota tim harus memiliki informasi yang diperlukan untuk mengoperasikan dan mendukung beban kerja. Manajemen pengetahuan merupakan kunci untuk penyediaan tersebut.

Dokumen terkait:

- Acara dan Webinar AWS
- AWS Training and Certification

OPS07-BP02 Memastikan peninjauan yang konsisten terkait kesiapan operasional

Gunakan Peninjauan Kesiapan Operasional (ORR) untuk memvalidasi bahwa Anda dapat mengoperasikan beban kerja Anda. ORR adalah mekanisme yang dikembangkan di Amazon untuk

memvalidasi bahwa tim dapat mengoperasikan beban kerja mereka dengan aman. ORR adalah proses peninjauan dan inspeksi menggunakan daftar periksa persyaratan. ORR adalah pengalaman layanan mandiri yang digunakan tim untuk memastikan beban kerja mereka. ORR mencakup praktik terbaik dari pelajaran yang kami dapatkan selama bertahun-tahun membangun perangkat lunak.

Daftar periksa ORR terdiri dari rekomendasi arsitektur, proses operasional, manajemen peristiwa, dan kualitas rilis. Proses Koreksi Kesalahan (CoE) kami merupakan pendorong utama item-item ini. Analisis pascainsiden Anda sendiri harus mendorong pengembangan ORR Anda. ORR tidak hanya tentang mengikuti praktik terbaik tapi juga mencegah kemungkinan peristiwa yang telah Anda lihat sebelumnya. Terakhir, keamanan, pengelolaan, dan kepatuhan persyaratan juga dapat disertakan dalam ORR.

Jalankan ORR sebelum beban kerja meluncur ke ketersediaan umum dan kemudian ke seluruh siklus pengembangan perangkat lunak. Menjalankan ORR sebelum peluncuran meningkatkan kemampuan Anda untuk mengoperasikan beban kerja dengan aman. Jalankan kembali ORR Anda secara berkala pada beban kerja untuk mengetahui penyimpangan dari praktik terbaik. Anda dapat memiliki daftar periksa ORR untuk peluncuran layanan baru dan ORR untuk peninjauan berkala. Ini membantu Anda untuk tetap up to date dengan praktik terbaik yang muncul dan menggabungkan pelajaran yang didapatkan dari analisis pascainsiden. Saat penggunaan cloud Anda matang, Anda dapat membangun persyaratan ORR ke dalam arsitektur Anda secara default.

Hasil yang diinginkan: Anda memiliki daftar periksa ORR dengan praktik terbaik untuk organisasi Anda. ORR dilakukan sebelum peluncuran beban kerja. ORR dijalankan secara berkala selama kursus siklus beban kerja.

Antipola umum:

- Anda meluncurkan beban kerja tanpa mengetahui apakah Anda dapat mengoperasikannya.
- Persyaratan pengelolaan dan keamanan tidak diikutsertakan ketika menyertifikasi beban kerja untuk peluncuran.
- Beban kerja tidak dievaluasi kembali secara berkala.
- Beban kerja diluncurkan tanpa diterapkannya prosedur yang diperlukan.
- Anda melihat pengulangan kegagalan akar masalah yang sama di beberapa beban kerja.

Manfaat menjalankan praktik terbaik ini:

- Beban kerja Anda mencakup praktik terbaik arsitektur, proses, dan manajemen.

- Pelajaran yang didapatkan digabungkan dalam proses ORR.
- Prosedur yang diperlukan tersedia ketika beban kerja diluncurkan.
- ORR dijalankan di seluruh siklus perangkat lunak beban kerja Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

ORR adalah dua hal: proses dan daftar periksa. Proses ORR Anda harus diadopsi oleh organisasi Anda dan didukung oleh sponsor eksekutif. Minimal, ORR harus dilakukan sebelum beban kerja meluncur ke ketersediaan umum. Jalankan ORR di seluruh siklus pengembangan perangkat lunak untuk tetap up to date dengan praktik terbaik atau persyaratan baru. Daftar periksa ORR harus mencakup item konfigurasi, persyaratan keamanan dan pengelolaan, serta praktik terbaik dari organisasi Anda. Seiring waktu, Anda dapat menggunakan layanan, seperti AWS Config, AWS Security Hub, dan Pagar Pembatas AWS Control Tower, untuk membangun praktik terbaik dari ORR ke pagar pembatas untuk deteksi praktik terbaik secara otomatis.

Contoh pelanggan

Setelah beberapa insiden produksi, AnyCompany Retail memutuskan untuk menerapkan proses ORR. Mereka membangun daftar periksa yang terdiri dari praktik terbaik, persyaratan pengelolaan dan kepatuhan, serta pelajaran yang didapatkan dari pemadaman. Beban kerja baru melakukan ORR sebelum diluncurkan. Setiap beban kerja melakukan ORR setiap tahun dengan sebagian praktik terbaik untuk menggabungkan praktik terbaik dan persyaratan baru yang ditambahkan ke daftar periksa ORR. Seiring waktu, AnyCompany Retail menggunakan AWS Config untuk mendeteksi beberapa praktik terbaik, yang mempercepat proses ORR.

Langkah implementasi

Untuk mempelajari selengkapnya tentang ORR, baca: laporan resmi Peninjauan Kesiapan Operasional (ORR). Laporan resmi ini menyediakan detail informasi tentang riwayat proses ORR, cara membangun praktik ORR Anda sendiri, dan cara mengembangkan daftar periksa ORR Anda. Langkah-langkah berikut ini merupakan versi singkat dari dokumen tersebut. Untuk pemahaman yang mendalam tentang apa itu ORR dan bagaimana membangunnya, sebaiknya baca laporan resmi tersebut.

1. Kumpulkan pemangku kepentingan utama, termasuk perwakilan dari keamanan, operasi, dan pengembangan.

2. Minta setiap pemangku kepentingan untuk menyediakan setidaknya satu persyaratan. Untuk iterasi pertama, coba batasi jumlah item menjadi 30 atau kurang.
   - Lampiran B: Contoh pertanyaan ORR dari laporan resmi Peninjauan Kesiapan Operasional (ORR) yang berisi sampel pertanyaan yang dapat Anda gunakan untuk memulai.
3. Kumpulkan persyaratan Anda ke dalam lembar kerja.
   - Anda dapat menggunakan lensa kustom di AWS Well-Architected Tool untuk mengembangkan ORR Anda dan membagikannya ke seluruh akun dan Organisasi AWS Anda.
4. Identifikasi satu beban kerja untuk diberikan ORR. Idealnya adalah beban kerja sebelum peluncuran atau beban kerja internal.
5. Pelajari daftar periksa ORR dan catat semua penemuan yang dibuat. Penemuannya mungkin akan buruk jika terdapat mitigasi. Untuk penemuan yang minim mitigasi, tambahkan beban kerja ke backlog item Anda dan implementasikan sebelum peluncuran.
6. Lanjutkan penambahan praktik terbaik dan persyaratan ke daftar periksa ORR Anda seiring waktu.

Pelanggan AWS Support dengan Enterprise Support dapat mengajukan permintaan Lokakarya Peninjauan Kesiapan Operasional dari Manajer Akun Teknis mereka. Lokakarya ini adalah sesi penelusuran mundur (working backward) interaktif untuk mengembangkan daftar periksa ORR Anda.

Tingkat upaya untuk rencana implementasi: Tinggi. Untuk mengadopsi praktik ORR pada organisasi Anda diperlukan sponsor eksekutif dan dukungan pemangku kepentingan. Buat dan perbarui daftar periksa dengan masukan dari seluruh organisasi Anda.

Sumber daya

Praktik Terbaik Terkait:

- OPS01-BP03 Evaluasi persyaratan tata kelola – Persyaratan tata kelola sangat sesuai untuk daftar periksa ORR.
- OPS01-BP04 Evaluasi persyaratan kepatuhan – Terkadang persyaratan kepatuhan tercantum di daftar periksa ORR. Terkadang persyaratan kepatuhan adalah proses yang terpisah.
- OPS03-BP07 Bekali tim dengan sumber daya dengan sesuai – Kemampuan tim merupakan kandidat yang bagus untuk persyaratan ORR.
- OPS06-BP01 Antisipasikan perubahan yang tidak berhasil – Rencana rollback atau rollfoward harus dibuat sebelum Anda meluncurkan beban kerja Anda.
- OPS07-BP01 Memastikan kemampuan personel – Untuk mendukung beban kerja, Anda harus memiliki personel yang diperlukan.

- SEC01-BP03 Mengidentifikasi dan memvalidasi tujuan kontrol – Tujuan kontrol keamanan menyempurnakan persyaratan ORR.
- REL13-BP01 Menetapkan sasaran pemulihan untuk waktu henti dan kehilangan data – Rencana pemulihan bencana merupakan persyaratan ORR yang bagus.
- COST02-BP01 Mengembangkan kebijakan berdasarkan keperluan organisasi Anda – Kebijakan manajemen biaya bagus untuk dicantumkan dalam daftar ORR Anda.

Dokumen terkait:

- AWS Control Tower - Pagar Pembatas di AWS Control Tower
- AWS Well-Architected Tool - Lensa Kustom
- Templat Peninjauan Kesiapan Operasional oleh Adrian Hornsby
- Laporan Resmi Peninjauan Kesiapan Operasional (ORR)

Video terkait:

- AWS Support Anda | Membangun Peninjauan Kesiapan Operasional (ORR) yang Efektif

Contoh terkait:

- Sampel Lensa Peninjauan Kesiapan Operasional (ORR)

Layanan terkait:

- AWS Config
- AWS Control Tower
- AWS Security Hub
- AWS Well-Architected Tool

OPS07-BP03 Menggunakan runbook untuk menjalankan prosedur

Sebuah runbook adalah proses terdokumentasi untuk mencapai hasil tertentu. Runbook terdiri dari serangkaian langkah yang diikuti seseorang untuk menyelesaikan sesuatu. Runbook telah digunakan dalam operasi sejak masa-masa awal industri penerbangan. Dalam operasi cloud, kita

menggunakan runbook untuk mengurangi risiko dan mencapai hasil yang diinginkan. Dalam bentuk paling sederhananya, runbook adalah daftar periksa untuk menyelesaikan tugas.

Runbook adalah bagian penting dari operasi beban kerja Anda. Mulai dari orientasi anggota tim baru hingga melakukan deployment rilis utama, runbook adalah proses terkodifikasi yang memberikan hasil konsisten, siapa pun yang menggunakannya. Runbook harus dipublikasikan di lokasi sentral dan diperbarui seiring prosesnya berkembang karena memperbarui runbook adalah komponen utama dari proses manajemen perubahan. Runbook juga harus menyertakan panduan tentang penanganan kesalahan, alat, izin, pengecualian, dan eskalasi jika terjadi masalah.

Saat organisasi Anda matang, mulailah mengotomatiskan runbook. Mulailah dengan runbook yang singkat dan sering digunakan. Gunakan bahasa skrip untuk mengotomatiskan langkah-langkah atau mempermudah pelaksanaan langkah-langkah. Seiring Anda mengotomatiskan beberapa runbook pertama, Anda akan mendedikasikan waktu untuk mengotomatiskan runbook yang lebih kompleks. Seiring waktu, sebagian besar runbook Anda harus diotomatiskan dalam cara tertentu.

Hasil yang diinginkan: Tim Anda memiliki kumpulan panduan langkah demi langkah untuk melakukan tugas beban kerja. Runbook berisi hasil yang diinginkan, alat dan izin yang diperlukan, serta petunjuk untuk penanganan kesalahan. Runbook disimpan di lokasi sentral dan sering diperbarui.

Antipola umum:

- Mengandalkan memori untuk menyelesaikan setiap langkah dari suatu proses.
- Menerapkan perubahan secara manual tanpa daftar periksa.
- Anggota tim yang berbeda-beda melakukan proses yang sama, tetapi dengan langkah atau hasil yang berbeda.
- Membiarkan runbook tidak sinkron dengan perubahan sistem dan otomatisasi.

Manfaat menjalankan praktik terbaik ini:

- Mengurangi tingkat kesalahan untuk tugas manual.
- Operasi dilakukan secara konsisten.
- Anggota tim baru dapat mulai melakukan tugas dengan lebih cepat.
- Runbook dapat diotomatiskan untuk mengurangi upaya yang diperlukan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Runbook dapat memiliki beberapa bentuk, bergantung pada tingkat kematangan organisasi Anda. Minimal, runbook harus terdiri dari dokumen teks langkah demi langkah. Hasil yang diinginkan harus ditunjukkan dengan jelas. Dokumentasikan dengan jelas izin atau alat khusus yang diperlukan. Berikan panduan mendetail tentang penanganan kesalahan dan eskalasi jika terjadi kesalahan. Cantumkan pemilik runbook dan publikasikan di lokasi sentral. Setelah runbook Anda didokumentasikan, validasikan dengan meminta orang lain di tim Anda untuk menjalankannya. Seiring prosedur berkembang, perbarui runbook Anda sesuai dengan proses manajemen perubahan Anda.

Runbook teks Anda harus diotomatiskan seiring organisasi Anda makin matang. Dengan layanan seperti otomatisasi AWS Systems Manager, Anda dapat mentransformasikan teks biasa menjadi otomatisasi yang dapat dijalankan dengan beban kerja Anda. Otomatisasi ini dapat dijalankan sebagai respons terhadap peristiwa, sehingga mengurangi beban operasional untuk memelihara beban kerja Anda.

Contoh pelanggan

AnyCompany Retail harus melakukan pembaruan skema basis data selama deployment perangkat lunak. Tim Operasi Cloud bekerja sama dengan Tim Administrasi Basis Data untuk membuat runbook guna menerapkan perubahan ini secara manual. Runbook ini mencantumkan setiap langkah prosesnya dalam bentuk daftar periksa. Runbook ini berisi bagian tentang penanganan kesalahan jika terjadi kesalahan. Mereka memublikasikan runbook di wiki internal mereka bersama dengan runbook mereka yang lain. Tim Operasi Cloud berencana untuk mengotomatiskan runbook dalam sprint mendatang.

Langkah implementasi

Jika Anda belum memiliki repositori dokumen, repositori kontrol versi adalah tempat yang tepat untuk mulai membangun pustaka runbook Anda. Anda dapat membangun runbook Anda menggunakan Markdown. Kami telah menyediakan contoh templat runbook yang dapat Anda gunakan untuk mulai membangun runbook.

```
# Judul Runbook ## Info Runbook | ID Runbook | Deskripsi | Alat yang Digunakan
 | Izin Khusus | Penulis Runbook | Terakhir Diperbarui | POC Eskalasi |
 |-------|-------|-------|-------|-------|-------|-------| | RUN001 | Apa tujuan
 penggunaan runbook ini? Apa hasil yang diinginkan? | Alat | Izin | Nama Anda |
 21-9-2022 | Nama Eskalasi | ## Langkah 1. Langkah pertama 2. Langkah kedua
```

1. Jika Anda belum memiliki repositori atau wiki dokumentasi, buat repositori kontrol versi baru di sistem kontrol versi Anda.
2. Identifikasi proses yang tidak memiliki runbook. Proses yang ideal adalah proses yang dilakukan secara semireguler, sedikit jumlah langkahnya, dan memiliki kegagalan berdampak rendah.
3. Di repositori dokumen Anda, buat draf dokumen Markdown baru menggunakan templat tersebut. Isi `Judul Runbook` dan bidang yang diperlukan di bagian `Info Runbook`.
4. Dimulai dengan langkah pertama, isi bagian `Langkah` dalam runbook ini.
5. Berikan runbook ini kepada para anggota tim. Minta mereka menggunakan runbook ini untuk memvalidasi langkah-langkahnya. Jika ada sesuatu yang belum dimasukkan atau memerlukan kejelasan, perbarui runbook ini.
6. Publikasikan runbook ini ke penyimpanan dokumentasi internal Anda. Setelah dipublikasikan, beri tahu tim Anda dan pemangku kepentingan lainnya.
7. Seiring waktu, Anda akan membangun pustaka runbook. Saat pustaka tersebut tumbuh, mulailah bekerja untuk mengotomatiskan runbook.

Tingkat upaya untuk rencana implementasi: Rendah. Standar minimum untuk runbook adalah panduan teks langkah demi langkah. Mengotomatiskan runbook dapat meningkatkan upaya implementasi.

Sumber daya

Praktik terbaik terkait:

- OPS02-BP02 Proses dan Prosedur memiliki pemilik teridentifikasi: Runbook harus memiliki pemilik yang bertanggung jawab untuk memeliharanya.
- OPS07-BP04 Menggunakan buku panduan untuk menyelidiki masalah: Runbook dan playbook mirip satu sama lain dengan satu perbedaan utama: runbook memiliki hasil yang diinginkan. Dalam banyak kasus, runbook dipicu setelah playbook mengidentifikasi akar penyebab.
- OPS10-BP01 Menggunakan proses untuk manajemen peristiwa, insiden, dan masalah: Runbook adalah bagian dari praktik manajemen yang baik terkait peristiwa, insiden, dan masalah.
- OPS10-BP02 Menjalankan proses untuk setiap peringatan: Runbook dan playbook harus digunakan untuk menanggapi peringatan. Seiring waktu, reaksi ini harus diotomatiskan.
- OPS11-BP04 Menjalankan manajemen pengetahuan: Memelihara runbook adalah bagian penting dari manajemen pengetahuan.

Dokumen terkait:

- Mencapai Keunggulan Operasional menggunakan playbook dan runbook otomatis
- AWS Systems Manager: Bekerja dengan runbook
- Playbook migrasi untuk migrasi besar AWS - Tugas 4: Meningkatkan runbook migrasi Anda
- Gunakan runbook Otomatisasi AWS Systems Manager untuk menyelesaikan tugas operasional

Video terkait:

- AWS re:Invent 2019: Panduan mandiri untuk runbook, laporan insiden, dan respons insiden (SEC318-R1)
- Cara mengotomatiskan Operasi IT di AWS | Amazon Web Services
- Integrasikan Skrip ke dalam AWS Systems Manager

Contoh terkait:

- AWS Systems Manager: Panduan otomatisasi
- AWS Systems Manager: Pulihkan volume root dari snapshot runbook terbaru
- Membangun runbook respons insiden AWS menggunakan notebook Jupyter dan CloudTrail Lake
- Gitlab - Runbook
- Rubix - Pustaka Python untuk membuat runbook di Notebook Jupyter
- Menggunakan Document Builder untuk membuat runbook kustom
- Lab Well-Architected: Mengotomatiskan operasi dengan Playbook dan Runbook

Layanan terkait:

- Otomatisasi AWS Systems Manager

OPS07-BP04 Menggunakan buku panduan untuk menyelidiki masalah

Playbook adalah panduan mendetail yang digunakan untuk menyelidiki insiden. Ketika terjadi sebuah insiden, playbook digunakan untuk menyelidiki, membuat cakupan dampak, dan mengidentifikasi akar masalah. Playbook digunakan untuk berbagai skenario, dari deployment yang gagal hingga insiden keamanan. Dalam banyak kasus, playbook mengidentifikasi akar masalah yang dimitigasi

menggunakan runbook. Playbook adalah komponen pokok dalam rencana respons insiden organisasi Anda.

Playbook yang baik memiliki sejumlah fitur utama. Playbook memberikan panduan secara mendetail bagi pengguna, dalam proses penemuan. Dengan berpikir secara holistik, langkah apa saja yang sebaiknya diikuti seseorang untuk mendiagnosis insiden? Tetapkan secara jelas di dalam playbook jika alat-alat khusus atau izin yang dinaikkan diperlukan di dalam playbook. Memiliki rencana komunikasi untuk memberi informasi kepada para pemangku kepentingan mengenai status penyelidikan adalah komponen utama. Dalam situasi ketika akar penyebab tidak dapat diidentifikasi, playbook harus memiliki rencana eskalasi. Jika akar masalah diidentifikasi, playbook harus mengarah ke runbook yang menjelaskan cara menyelesaikannya. Playbook harus disimpan secara terpusat dan dipelihara secara rutin. Jika playbook digunakan untuk pemberitahuan khusus, bekali tim Anda dengan penunjuk ke playbook di dalam pemberitahuan tersebut.

Otomatisasi playbook Anda seiring kematangan organisasi. Mulai dengan playbook yang mencakup insiden berisiko rendah. Gunakan penulisan skrip untuk mengotomatiskan langkah-langkah penemuan. Pastikan Anda memiliki runbook pendamping untuk memitigasi akar masalah umum.

Hasil yang diinginkan: Organisasi Anda memiliki playbook untuk insiden umum. Playbook disimpan di lokasi terpusat dan tersedia untuk anggota tim Anda. Playbook sering diperbarui. Runbook pendamping dibuat untuk akar masalah apa pun yang diketahui.

Antipola umum:

- Tidak ada cara standar untuk menyelidiki insiden.
- Anggota tim mengandalkan memori otot atau pengetahuan institusional untuk memecahkan masalah kegagalan deployment.
- Anggota tim baru mempelajari cara menyelidiki permasalahan melalui coba-coba.
- Praktik terbaik untuk menyelidiki permasalahan tidak dibagikan ke seluruh tim.

Manfaat menjalankan praktik terbaik ini:

- Playbook meningkatkan upaya Anda untuk memitigasi insiden.
- Anggota tim yang berbeda-beda dapat menggunakan playbook yang sama untuk mengidentifikasi akar masalah secara konsisten.
- Setelah akar masalah diketahui kemudian bisa dikembangkan runbook, sehingga dapat mempercepat waktu pemulihan.

- Playbook memungkinkan anggota tim untuk mulai berkontribusi lebih cepat.
- Tim dapat menskalakan proses mereka dengan playbook yang dapat diulang.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Bagaimana Anda membangun dan menggunakan playbook bergantung pada kematangan organisasi Anda. Jika Anda baru mengenal cloud, bangun playbook dalam bentuk teks di dalam repositori dokumen pusat. Seiring kematangan organisasi, playbook dapat menjadi semi-otomatis dengan bahasa skrip seperti Python. Skrip-skrip ini dapat dijalankan di dalam notebook Jupyter untuk mempercepat penemuan. Organisasi tingkat lanjut memiliki playbook yang sepenuhnya otomatis untuk permasalahan umum yang diperbaiki secara otomatis dengan runbook.

Mulai bangun playbook Anda dengan mengidentifikasi insiden-insiden umum yang terjadi pada beban kerja Anda. Pilih playbook untuk insiden berisiko rendah dan dengan akar masalah yang telah dipersempit menjadi beberapa permasalahan untuk mengawalinya. Setelah Anda memiliki playbook untuk skenario yang lebih sederhana, beralihlah ke skenario berisiko lebih tinggi atau skenario dengan akar masalah yang tidak dikenal dengan baik.

Playbook teks Anda harus diotomatiskan seiring pematangan organisasi Anda. Dengan layanan seperti AWS Systems Manager Automations, teks biasa dapat ditransformasikan menjadi otomatis. Otomatisasi ini dapat dijalankan terhadap beban kerja untuk mempercepat penyelidikan. Otomatisasi ini dapat diaktifkan untuk merespons peristiwa, sehingga mengurangi rata-rata waktu untuk menemukan dan menyelesaikan insiden.

Pelanggan dapat menggunakan AWS Systems Manager Incident Manager untuk merespons insiden. Layanan ini menyediakan satu antarmuka untuk memeriksa insiden, memberi informasi kepada pemangku kepentingan selama penemuan dan mitigasi, dan berkolaborasi melalui insiden. Layanan ini menggunakan AWS Systems Manager Automations untuk mempercepat deteksi dan pemulihan.

Contoh pelanggan

Insiden produksi memberikan dampak pada AnyCompany Retail. Rekayasawan yang siap dipanggil kapan saja (on-call) menggunakan playbook untuk menyelidiki permasalahan. Seiring mereka mengikuti langkah-langkahnya, mereka terus memutakhirkan pemangku kepentingan utama yang diidentifikasi di dalam playbook. Rekayasawan mengidentifikasi akar masalah sebagai kondisi pacu di dalam layanan backend. Menggunakan runbook, rekayasawan meluncurkan ulang layanan, sehingga AnyCompany Retail dapat kembali online.

Langkah implementasi

Jika Anda belum memiliki repositori dokumen, kami menyarankan pembuatan repositori kontrol versi untuk pustaka playbook Anda. Anda dapat membangun playbook Anda menggunakan Markdown, yang kompatibel dengan sebagian besar sistem otomatisasi playbook. Jika Anda memulai dari nol, gunakan contoh templat playbook berikut ini.

```
# Judul Playbook ## Info Playbook | ID Playbook | Deskripsi | Alat
 yang Digunakan | Izin Khusus | Penyusun Playbook | Terakhir Diperbarui
 | POC Eskalasi | Pemangku Kepentingan | Rencana Komunikasi |
 |-------|-------|-------|-------|-------|-------|-------|-------|-------| | RUN001
 | Untuk apa playbook ini? Untuk insiden apa playbook ini? | Alat | Izin | Nama Anda
 | 21-09-2022 | Nama Eskalasi | Nama Pemangku Kepentingan | Bagaimana pembaruan akan
 disampaikan selama penyelidikan? | ## Langkah 1. Langkah pertama 2. Langkah kedua
```

1. Jika Anda belum memiliki repositori dokumen atau wiki, buat repositori kontrol versi baru untuk playbook Anda di sistem kontrol versi Anda.
2. Identifikasi permasalahan umum yang memerlukan penyelidikan. Ini sebaiknya adalah skenario dengan akar masalah yang dibatasi ke beberapa permasalahan dan penyelesaiannya berisiko rendah.
3. Menggunakan templat Markdown, lengkapi bagian `Nama Playbook` dan bidang di bawah `Info Playbook`.
4. Lengkapi langkah-langkah pemecahan masalah. Sampaikan sejelas mungkin tindakan yang akan dilakukan atau area apa saja yang harus Anda selidiki.
5. Berikan playbook tersebut kepada anggota tim dan minta mereka mempelajari dan memvalidasinya. Jika terdapat hal yang terlewat atau tidak jelas, perbarui playbook.
6. Terbitkan playbook di dalam repositori dokumen Anda dan informasikan kepada tim dan pemangku kepentingan.
7. Pustaka playbook ini akan tumbuh seiring Anda menambahkan lebih banyak playbook. Setelah Anda memiliki beberapa playbook, mulailah mengotomatiskannya menggunakan alat seperti AWS Systems Manager Automations untuk terus menyinkronkan otomatisasi dan playbook.

Tingkat upaya untuk rencana implementasi: Rendah. Playbook Anda harus berupa dokumen teks yang disimpan di lokasi terpusat. Organisasi yang lebih matang akan beralih ke otomatisasi playbook.

Sumber daya

Praktik terbaik terkait:

- OPS02-BP02 Proses dan Prosedur memiliki pemilik teridentifikasi: Runbook harus memiliki pemilik yang bertanggung jawab untuk memeliharanya.
- OPS07-BP03 Menggunakan runbook untuk menjalankan prosedur: Runbook dan playbook mirip, tetapi dengan satu perbedaan utama: runbook memiliki hasil yang diinginkan. Dalam banyak kasus, runbook digunakan setelah playbook mengidentifikasi akar penyebab.
- OPS10-BP01 Menggunakan proses untuk manajemen peristiwa, insiden, dan masalah: Playbook adalah bagian dari praktik manajemen yang baik terkait peristiwa, insiden, dan masalah.
- OPS10-BP02 Menjalankan proses untuk setiap peringatan: Runbook dan playbook harus digunakan untuk menanggapi peringatan. Seiring waktu, reaksi ini harus diotomatiskan.
- OPS11-BP04 Menjalankan manajemen pengetahuan: Memelihara playbook adalah bagian penting dari manajemen pengetahuan.

Dokumen terkait:

- Mencapai Keunggulan Operasional menggunakan playbook dan runbook otomatis
- AWS Systems Manager: Bekerja dengan runbook
- Gunakan runbook AWS Systems Manager Automations untuk menyelesaikan tugas operasional

Video terkait:

- AWS re:Invent 2019: Panduan mandiri untuk runbook, laporan insiden, dan respons insiden (SEC318-R1)
- AWS Systems Manager Incident Manager - Lokakarya Virtual AWS
- Integrasikan Skrip ke dalam AWS Systems Manager

Contoh terkait:

- Kerangka Kerja Playbook Pelanggan AWS
- AWS Systems Manager: Panduan otomatisasi
- Membangun runbook respons insiden AWS menggunakan notebook Jupyter dan CloudTrail Lake
- Rubix – Pustaka Python untuk membuat runbook di Notebook Jupyter

- Menggunakan Document Builder untuk membuat runbook kustom
- Lab Well-Architected: Mengotomatiskan operasi dengan Playbook dan Runbook
- Lab Well-Architect: Playbook respons insiden dengan Jupyter

Layanan terkait:

- AWS Systems Manager Automation
- AWS Systems Manager Incident Manager

OPS07-BP05 Membuat keputusan yang tepat untuk melakukan deployment sistem dan perubahan

Miliki proses untuk perubahan yang sukses dan tidak sukses pada beban kerja Anda. Pre-mortem adalah latihan simulasi tim terhadap kegagalan untuk mengembangkan strategi mitigasi. Gunakan pre-mortem untuk mengantisipasi kegagalan dan menciptakan prosedur ketika diperlukan. Evaluasi manfaat dan risiko dari deployment perubahan ke beban kerja Anda. Verifikasi apakah semua perubahan mematuhi tata kelola.

Hasil yang diinginkan:

- Anda mengambil keputusan yang tepat ketika melakukan deployment perubahan ke beban kerja Anda.
- Perubahan mematuhi tata kelola.

Antipola umum:

- Melakukan deployment perubahan ke beban kerja tanpa proses untuk menangani deployment yang gagal.
- Membuat perubahan pada lingkungan produksi Anda yang tidak mematuhi persyaratan tata kelola.
- Melakukan deployment versi baru beban kerja Anda tanpa menetapkan garis dasar untuk pemanfaatan sumber daya.

Manfaat menjalankan praktik terbaik ini:

- Anda siap untuk menangani perubahan yang tidak sukses pada beban kerja Anda.

- Perubahan pada beban kerja Anda mematuhi kebijakan tata kelola.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Rendah

Panduan implementasi

Gunakan pre-mortem guna mengembangkan proses untuk perubahan yang tidak sukses. Dokumentasikan proses Anda untuk perubahan yang tidak sukses. Pastikan semua perubahan mematuhi tata kelola. Evaluasi manfaat dan risiko deployment perubahan ke beban kerja Anda.

Contoh pelanggan

AnyCompany Retail melakukan pre-mortem secara teratur guna memvalidasi proses mereka untuk perubahan yang tidak sukses. Mereka mendokumentasikan proses mereka di Wiki bersama dan sering kali memperbaruinya. Semua perubahan mematuhi persyaratan tata kelola.

Langkah implementasi

1. Ambil keputusan yang tepat ketika melakukan deployment perubahan ke beban kerja Anda. Tetapkan dan tinjau kriteria untuk deployment yang sukses. Kembangkan skenario atau kriteria yang akan memicu pengembalian perubahan ke versi sebelumnya. Pikirkan manfaat deployment perubahan dibandingkan risiko perubahan yang tidak sukses.
2. Verifikasi apakah semua perubahan mematuhi kebijakan tata kelola.
3. Gunakan pre-mortem guna membuat rencana untuk perubahan yang tidak sukses dan mendokumentasikan strategi mitigasi. Jalankan sesi latihan table-top untuk memperagakan perubahan yang tidak sukses dan memvalidasi prosedur pengembalian ke versi sebelumnya.

Tingkat upaya untuk rencana implementasi: Sedang. Mengimplementasikan praktik pre-mortem memerlukan koordinasi dan upaya dari para pemangku kepentingan dalam seluruh organisasi Anda

Sumber daya

Praktik terbaik terkait:

- OPS01-BP03 Evaluasi persyaratan tata kelola - Persyaratan tata kelola merupakan faktor kunci dalam menentukan apakah akan melakukan deployment perubahan.
- OPS06-BP01 Antisipasikan perubahan yang tidak berhasil - Buat rencana untuk memitigasi deployment yang gagal dan gunakan pre-mortem untuk memvalidasinya.

- OPS06-BP02 Menguji deployment - Setiap perubahan perangkat lunak harus diuji dengan tepat sebelum deployment untuk mengurangi kecacatan dalam produksi.
- OPS07-BP01 Memastikan kemampuan personel - Memiliki cukup banyak personel yang terlatih untuk mendukung beban kerja sangat penting dalam mengambil keputusan yang tepat dalam hal deployment perubahan sistem.

Dokumen terkait:

- Amazon Web Services: Risiko dan Kepatuhan
- Model Tanggung Jawab Bersama AWS
- Tata Kelola dalam AWS Cloud: Keseimbangan yang Tepat Antara Ketangkasan dan Keamanan

OPS07-BP06 Mengaktifkan rencana dukungan untuk beban kerja produksi

Aktifkan dukungan untuk perangkat lunak dan layanan yang diandalkan beban kerja produksi Anda. Pilih tingkat dukungan yang sesuai untuk memenuhi kebutuhan tingkat layanan produksi Anda. Rencana dukungan untuk dependensi ini diperlukan untuk berjaga-jaga jika ada gangguan layanan atau masalah perangkat lunak. Dokumentasikan rencana dukungan dan cara meminta dukungan untuk semua vendor perangkat lunak dan layanan. Implementasikan mekanisme yang memverifikasi bahwa titik kontak dukungan selalu yang terbaru.

Hasil yang diinginkan:

- Implementasikan rencana dukungan untuk perangkat lunak dan layanan yang diandalkan beban kerja produksi.
- Pilih rencana dukungan yang sesuai berdasarkan kebutuhan tingkat layanan.
- Dokumentasikan rencana dukungan, tingkat dukungan, dan cara meminta dukungan.

Antipola umum:

- Anda tidak memiliki rencana dukungan untuk vendor perangkat lunak yang penting. Beban kerja Anda terkena dampaknya dan Anda tidak dapat melakukan apa-apa untuk mempercepat perbaikan atau mendapatkan informasi terbaru yang tepat waktu dari vendor.
- Developer yang merupakan titik utama kontak untuk vendor perangkat lunak tidak lagi bekerja di perusahaan. Anda tidak dapat menghubungi dukungan vendor secara langsung. Anda

harus meluangkan waktu menelusuri dan mencari-cari dalam sistem kontak generik, sehingga menambah waktu yang diperlukan untuk merespons ketika diperlukan.

- Penghentian produksi terjadi pada vendor perangkat lunak. Tidak ada dokumentasi tentang cara mengajukan kasus dukungan.

Manfaat menjalankan praktik terbaik ini:

- Dengan tingkat dukungan yang sesuai, Anda dapat memperoleh respons dalam kerangka waktu yang diperlukan untuk memenuhi kebutuhan tingkat layanan.
- Sebagai pelanggan yang didukung, Anda dapat melapor jika ada masalah produksi.
- Vendor layanan dan perangkat lunak dapat membantu menyelesaikan masalah selama insiden.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Rendah

Panduan implementasi

Aktifkan rencana dukungan untuk vendor perangkat lunak dan layanan yang diandalkan beban kerja produksi Anda. Atur rencana dukungan yang sesuai untuk memenuhi kebutuhan tingkat layanan Anda. Untuk pelanggan AWS, ini artinya mengaktifkan Business Support AWS atau yang lebih tinggi pada akun apa pun tempat Anda memiliki beban kerja produksi. Temui vendor dukungan secara teratur untuk mendapatkan informasi terbaru mengenai kontak, proses, dan penawaran dukungan. Dokumentasikan cara meminta dukungan dari vendor perangkat lunak dan layanan, termasuk cara melapor jika ada penghentian. Implementasikan mekanisme untuk menjaga agar kontak selalu yang terbaru.

Contoh pelanggan

Di AnyCompany Retail, semua dependensi layanan dan perangkat lunak komersial memiliki rencana dukungan. Contohnya, mereka mengaktifkan AWS Enterprise Support di semua akun dengan beban kerja produksi. Semua developer dapat membuka kasus dukungan bila ada masalah. Ada halaman wiki dengan informasi tentang cara meminta dukungan, siapa yang harus diberi tahu, dan praktik terbaik untuk mempercepat kasus.

Langkah implementasi

1. Bekerjasamalah dengan pemangku kepentingan di organisasi Anda untuk mengidentifikasi vendor perangkat lunak dan layanan yang diandalkan beban kerja Anda. Dokumentasikan dependensi ini.

2. Tentukan kebutuhan tingkat layanan untuk beban kerja Anda. Pilih rencana dukungan yang selaras dengannya.
3. Untuk layanan dan perangkat lunak komersial, tetapkan rencana dukungan dengan vendor.
    a. Dengan berlangganan AWS Business Support atau lebih tinggi untuk semua akun produksi, waktu respons AWS Support akan lebih cepat dan hal ini sangat disarankan. Jika Anda tidak memiliki dukungan premium, Anda harus memiliki rencana tindakan untuk menangani masalah, yang memerlukan bantuan dari AWS Support. AWS Support memberikan kombinasi alat dan teknologi, orang, dan program yang dirancang untuk secara proaktif membantu Anda mengoptimalkan performa, menurunkan biaya, dan berinovasi dengan lebih cepat. AWS Business Support memberikan manfaat tambahan, termasuk akses ke AWS Trusted Advisor dan AWS Personal Health Dashboard dan waktu respons yang lebih cepat.
4. Dokumentasikan rencana dukungan di alat manajemen pengetahuan Anda. Sertakan cara untuk meminta dukungan, siapa yang harus diberi tahu jika kasus dukungan diajukan, dan cara untuk melapor selama insiden. Wiki merupakan mekanisme yang bagus untuk memungkinkan semua orang membuat pembaruan yang diperlukan pada dokumentasi ketika mereka mengetahui tentang perubahan untuk mendukung proses atau kontak.

Tingkat upaya untuk rencana implementasi: Rendah. Sebagian besar vendor perangkat lunak dan layanan menawarkan pilihan rencana dukungan. Mendokumentasikan dan berbagi praktik terbaik terkait dukungan di sistem manajemen pengetahuan Anda akan memastikan tim Anda mengetahui tindakan yang harus dilakukan jika ada masalah produksi.

Sumber daya

Praktik terbaik terkait:

- OPS02-BP02 Proses dan Prosedur memiliki pemilik teridentifikasi

Dokumen terkait:

- AWS Support Plans

Layanan terkait:

- AWS Business Support
- AWS Enterprise Support

Jalankan

Keberhasilan adalah pencapaian hasil bisnis yang diukur oleh metrik yang Anda tetapkan. Dengan memahami kesehatan beban kerja dan operasi Anda, Anda dapat mengidentifikasi ketika hasil organisasi dan bisnis mungkin akan terpapar risiko, atau sedang terpapar risiko, dan dapat meresponsnya dengan tepat.

Agar berhasil, Anda harus mampu:

Topik

Memanfaatkan observabilitas beban kerja

Memastikan kondisi beban kerja yang optimal dengan memanfaatkan observabilitas. Memanfaatkan metrik, log, dan jejak yang relevan untuk mendapatkan pandangan komprehensif tentang kinerja beban kerja Anda dan mengatasi masalah secara efisien.

Observabilitas memungkinkan Anda untuk fokus pada data yang bermakna serta memahami interaksi dan output beban kerja Anda. Dengan berkonsentrasi pada wawasan penting dan menghilangkan data yang tidak perlu, Anda mempertahankan pendekatan langsung untuk memahami kinerja beban kerja.

Hal ini sangat penting tidak hanya untuk mengumpulkan data tetapi juga untuk menafsirkannya dengan benar. Menentukan garis acuan yang jelas, menetapkan ambang batas peringatan yang sesuai, dan memantau secara aktif setiap penyimpangan. Pergeseran metrik kunci, terutama ketika berkorelasi dengan data lain, dapat menunjukkan dengan tepat area masalah tertentu.

Dengan observabilitas, Anda lebih siap untuk memperkirakan dan mengatasi tantangan potensial, memastikan bahwa beban kerja Anda beroperasi dengan lancar dan memenuhi kebutuhan bisnis.

AWS menawarkan alat khusus seperti Amazon CloudWatch untuk pemantauan dan pembuatan log, dan AWS X-Ray untuk penelusuran terdistribusi. Layanan ini terintegrasi dengan mudah dengan berbagai sumber daya AWS, memungkinkan pengumpulan data yang efisien, menyiapkan peringatan berdasarkan ambang batas yang telah ditentukan, dan menyajikan data di dasbor untuk interpretasi

yang mudah. Dengan memanfaatkan wawasan ini, Anda dapat membuat keputusan berdasarkan data yang matang yang sesuai dengan tujuan operasional Anda.

Praktik terbaik

- OPS08-BP01 Menganalisis metrik beban kerja
- OPS08-BP02 Menganalisis log beban kerja
- OPS08-BP03 Menganalisis jejak beban kerja
- OPS08-BP04 Membuat peringatan yang dapat ditindaklanjuti
- OPS08-BP05 Membuat dasbor

OPS08-BP01 Menganalisis metrik beban kerja

Setelah mengimplementasikan telemetri aplikasi, analisis metrik yang dikumpulkan secara rutin. Latensi, permintaan, kesalahan, dan kapasitas (atau kuota) memang memberikan wawasan tentang performa sistem, tetapi memprioritaskan peninjauan metrik hasil bisnis adalah hal yang sangat penting. Ini memastikan Anda mengambil keputusan berbasis data yang selaras dengan tujuan bisnis Anda.

Hasil yang diinginkan: Wawasan akurat tentang performa beban kerja yang mendorong keputusan berdasarkan informasi data, sehingga memastikan keselarasan dengan tujuan bisnis.

Antipola umum:

- Menganalisis metrik secara terpisah tanpa mempertimbangkan dampaknya terhadap hasil bisnis.
- Ketergantungan berlebihan pada metrik teknis sambil mengesampingkan metrik bisnis.
- Peninjauan metrik jarang dilakukan, sehingga peluang pengambilan keputusan waktu nyata terlewatkan.

Manfaat menjalankan praktik terbaik ini:

- Peningkatan pemahaman tentang korelasi antara performa teknis dan hasil bisnis.
- Perbaikan proses pengambilan keputusan yang berlandaskan data waktu nyata.
- Identifikasikan dan mitigasi masalah secara proaktif sebelum hasil bisnis terkena dampaknya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Manfaatkan alat seperti Amazon CloudWatch untuk melakukan analisis metrik. Layanan AWS seperti AWS Cost Anomaly Detection dan Amazon DevOps Guru dapat digunakan untuk mendeteksi anomali, terutama ketika ambang batas statis tidak diketahui atau ketika pola perilaku lebih cocok untuk deteksi anomali.

Langkah implementasi

1. Analisis dan tinjau: Tinjau dan tafsirkan metrik beban kerja Anda secara rutin.
   a. Prioritaskan metrik hasil bisnis daripada metrik teknis murni.
   b. Pahami signifikansi lonjakan, penurunan, atau pola dalam data Anda.
2. Manfaatkan Amazon CloudWatch: Gunakan Amazon CloudWatch untuk mendapatkan tampilan terpusat dan analisis mendalam.
   a. Konfigurasikan dasbor CloudWatch untuk memvisualisasikan metrik Anda dan membandingkannya dari waktu ke waktu.
   b. Gunakan persentil di CloudWatch untuk mendapatkan pandangan yang jelas tentang distribusi metrik, yang dapat membantu dalam mendefinisikan SLA dan memahami penyimpangan.
   c. Siapkan AWS Cost Anomaly Detection untuk mengidentifikasi pola yang tidak biasa tanpa bergantung pada ambang batas statis.
   d. Implementasikan observabilitas lintas akun CloudWatch untuk memantau dan memecahkan masalah aplikasi yang terjadi di beberapa akun di dalam suatu Wilayah.
   e. Gunakan Wawasan Metrik CloudWatch untuk mengkueri dan menganalisis data metrik di seluruh akun dan Wilayah, sehingga tren dan anomali dapat terdeteksi.
   f. Terapkan CloudWatch Metric Math untuk mengubah, menggabungkan, atau melakukan perhitungan pada metrik Anda untuk mendapatkan wawasan yang lebih mendalam.
3. Gunakan Amazon DevOps Guru: Sertakan Amazon DevOps Guru untuk memanfaatkan deteksi anomali yang disempurnakan dengan machine learning-nya untuk mengidentifikasi tanda-tanda awal masalah operasional untuk aplikasi nirserver Anda dan memperbaikinya sebelum berdampak pada pelanggan Anda.
4. Lakukan optimalisasi berdasarkan wawasan: Ambil keputusan cerdas berdasarkan analisis metrik Anda untuk menyesuaikan dan meningkatkan beban kerja Anda.

Tingkat upaya untuk Rencana Implementasi: Sedang

Sumber daya

Praktik terbaik terkait:

- OPS04-BP01 Identifikasikan indikator performa utama
- OPS04-BP02 Mengimplementasikan telemetri aplikasi

Dokumen terkait:

- The Wheel Blog - Menekankan pentingnya peninjauan metrik secara terus-menerus
- Persentil itu penting
- Menggunakan AWS Cost Anomaly Detection
- observabilitas lintas akun CloudWatch
- Mengkueri metrik dengan Wawasan Metrik CloudWatch

Video terkait:

- Mengaktifkan Observabilitas Lintas Akun di Amazon CloudWatch
- Pengantar Amazon DevOps Guru
- Menganalisis Metrik secara Berkelanjutan Menggunakan AWS Cost Anomaly Detection

Contoh terkait:

- Lokakarya One Observability
- Mendapatkan wawasan operasional dengan AIOps menggunakan Amazon DevOps Guru

OPS08-BP02 Menganalisis log beban kerja

Menganalisis log beban kerja secara rutin sangatlah penting untuk mendapatkan pemahaman yang lebih mendalam tentang aspek-aspek operasional aplikasi Anda. Dengan memilah-milah, memvisualisasikan, dan menafsirkan data log secara efisien, Anda dapat terus mengoptimalkan performa dan keamanan aplikasi.

Hasil yang diinginkan: Wawasan yang kaya tentang perilaku dan operasi aplikasi yang berasal dari analisis log yang menyeluruh, sehingga memastikan deteksi dan mitigasi masalah yang proaktif.

Antipola umum:

- Mengabaikan analisis log sampai masalah kritis muncul.
- Tidak menggunakan rangkaian alat lengkap yang tersedia untuk analisis log, sehingga wawasan kritis terlewatkan.
- Hanya mengandalkan tinjauan log manual tanpa memanfaatkan kemampuan otomatisasi dan kueri.

Manfaat menjalankan praktik terbaik ini:

- Identifikasikan kemacetan operasional, ancaman keamanan, dan masalah potensial lain secara proaktif.
- Pemanfaatan data log yang efisien untuk optimalisasi aplikasi berkelanjutan.
- Peningkatan pemahaman tentang perilaku aplikasi, sehingga membantu upaya debugging dan pemecahan masalah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Amazon CloudWatch Logs adalah alat yang ampuh untuk analisis log. Fitur terintegrasi seperti Wawasan dan Wawasan Kontributor CloudWatch Logs membuat proses perolehan informasi yang bermakna dari log menjadi intuitif dan efisien.

Langkah implementasi

1. Siapkan CloudWatch Logs: Konfigurasikan aplikasi dan layanan untuk mengirim log ke CloudWatch Logs.
2. Siapkan Wawasan CloudWatch Logs: Gunakan CloudWatch Logs Insights untuk mencari dan menganalisis data log Anda secara interaktif.
   a. Buat kueri untuk mengekstrak pola, memvisualisasikan data log, dan memperoleh wawasan yang dapat ditindaklanjuti.
3. Manfaatkan Wawasan Kontributor Gunakan CloudWatch Contributor Insights untuk mengidentifikasi sumber data teratas dalam dimensi kardinalitas tinggi seperti alamat IP atau agen pengguna.

4. Implementasikan filter metrik CloudWatch Logs: Konfigurasikan filter metrik log CloudWatch untuk mengubah data log menjadi metrik yang dapat ditindaklanjuti. Ini memungkinkan Anda untuk mengatur alarm atau menganalisis pola lebih lanjut.
5. Tinjauan dan penyempurnaan rutin: Tinjau strategi analisis log Anda secara berkala untuk menangkap semua informasi yang relevan dan terus mengoptimalkan performa aplikasi.

Tingkat upaya untuk rencana implementasi: Sedang.

Sumber daya

Praktik terbaik terkait:

- OPS04-BP01 Identifikasikan indikator performa utama
- OPS04-BP02 Mengimplementasikan telemetri aplikasi
- OPS08-BP01 Menganalisis metrik beban kerja

Dokumen terkait:

- Menganalisis Data Log dengan Wawasan CloudWatch Logs
- Menggunakan Wawasan Kontributor CloudWatch
- Membuat dan Mengelola Filter Metrik Log CloudWatch Logs

Video terkait:

- Menganalisis Data Log dengan Wawasan CloudWatch Logs
- Menggunakan Wawasan Kontributor CloudWatch untuk Menganalisis Data Kardinalitas Tinggi

Contoh terkait:

- Contoh Kueri CloudWatch Logs
- Lokakarya One Observability

OPS08-BP03 Menganalisis jejak beban kerja

Menganalisis data jejak sangatlah penting untuk mencapai pandangan yang komprehensif tentang perjalanan operasional aplikasi. Dengan memvisualisasikan dan memahami interaksi antara berbagai

komponen, performa dapat disesuaikan, kemacetan dapat diidentifikasi, dan pengalaman pengguna dapat ditingkatkan.

Hasil yang diinginkan: Dapatkan visibilitas yang jelas tentang operasi terdistribusi aplikasi Anda, sehingga memungkinkan penyelesaian masalah yang lebih cepat dan pengalaman pengguna yang disempurnakan.

Antipola umum:

- Mengabaikan data jejak, dan hanya mengandalkan log serta metrik.
- Tidak mengorelasikan data jejak dengan log terkait.
- Mengabaikan metrik yang berasal dari jejak, seperti latensi dan tingkat kesalahan.

Manfaat menjalankan praktik terbaik ini:

- Perbaiki kualitas pemecahan masalah dan kurangi rata-rata waktu penyelesaian (MTTR).
- Dapatkan wawasan tentang dependensi dan dampaknya.
- Identifikasikan dan perbaikan masalah performa secara cepat.
- Memanfaatkan metrik yang berasal dari jejak untuk pengambilan keputusan yang bijak.
- Pengalaman pengguna yang ditingkatkan melalui interaksi komponen yang dioptimalkan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

AWS X-Ray menawarkan rangkaian fitur komprehensif untuk analisis data jejak, sehingga menyediakan pandangan yang menyeluruh tentang interaksi layanan, memantau aktivitas pengguna, dan mendeteksi masalah performa. Fitur seperti ServiceLens, Wawasan X-Ray, Analitik X-Ray, dan Amazon DevOps Guru meningkatkan kedalaman wawasan yang dapat ditindaklanjuti yang berasal dari data jejak.

Langkah implementasi

Langkah-langkah berikut ini menawarkan pendekatan terstruktur untuk menerapkan analisis data jejak secara efektif menggunakan layanan AWS:

1. Integrasikan AWS X-Ray: Pastikan X-Ray terintegrasi dengan aplikasi Anda untuk menangkap data jejak.

2. Analisis metrik X-Ray: Selidiki metrik yang berasal dari jejak X-Ray seperti latensi, tingkat permintaan, tingkat kesalahan, dan distribusi waktu respons menggunakan peta layanan untuk memantau kondisi aplikasi.
3. Gunakan ServiceLens: Manfaatkan peta ServiceLens untuk meningkatkan observabilitas layanan dan aplikasi Anda. Fitur ini memungkinkan tampilan jejak, metrik, log, alarm, dan informasi kondisi lainnya secara terpadu.
4. Aktifkan Wawasan X-Ray:
   a. Aktifkan Wawasan X-Ray untuk deteksi anomali dalam jejak secara otomatis.
   b. Periksa wawasan untuk menentukan pola dan memastikan akar masalah, seperti peningkatan tingkat kesalahan atau latensi.
   c. Pelajari lini waktu wawasan untuk mendapatkan analisis kronologis pada masalah yang terdeteksi.
5. Gunakan Analitik X-Ray: Analitik X-Ray memungkinkan Anda menjelajahi data jejak, menentukan pola, dan mengekstrak wawasan secara menyeluruh.
6. Gunakan grup di X-Ray: Buat grup di X-Ray untuk memfilter jejak berdasarkan kriteria seperti latensi tinggi, sehingga memungkinkan analisis yang lebih tertarget.
7. Sertakan Amazon DevOps Guru: Libatkan Amazon DevOps Guru untuk mendapatkan manfaat dari model machine learning yang mengenali anomali operasional dalam jejak.
8. Gunakan CloudWatch Synthetics: Gunakan CloudWatch Synthetics untuk membuat canary untuk terus memantau titik akhir dan alur kerja Anda. Canary ini dapat terintegrasi dengan X-Ray untuk menyediakan data jejak untuk analisis aplikasi yang sedang diuji secara mendalam.
9. Gunakan Pemantauan Pengguna Nyata (RUM): Dengan AWS X-Ray dan CloudWatch RUM, Anda dapat menganalisis dan men-debug jalur permintaan mulai dari pengguna akhir aplikasi Anda hingga layanan AWS terkelola di hilir. Ini membantu Anda mengidentifikasi tren latensi dan kesalahan yang berdampak pada pengguna Anda.
10. Korelasikan dengan log: Korelasikan data jejak dengan log terkait di dalam tampilan jejak X-Ray untuk perspektif yang mendetail tentang perilaku aplikasi. Ini memungkinkan Anda melihat peristiwa log yang terkait langsung dengan transaksi yang dilacak.

Tingkat upaya untuk rencana implementasi: Sedang.

Sumber daya

Praktik terbaik terkait:

- OPS08-BP01 Menganalisis metrik beban kerja
- OPS08-BP02 Menganalisis log beban kerja

Dokumen terkait:

- Menggunakan ServiceLens untuk Memantau Kondisi Aplikasi
- Menjelajahi Data Jejak dengan X-Ray Analytics
- Mendeteksi Anomali di dalam Jejak dengan Wawasan X-Ray
- Pemantauan Berkelanjutan dengan CloudWatch Synthetics

Video terkait:

- Menganalisis dan Men-debug Aplikasi Menggunakan Amazon CloudWatch Synthetics dan AWS X-Ray
- Gunakan Wawasan AWS X-Ray

Contoh terkait:

- Lokakarya One Observability
- Mengimplementasikan X-Ray dengan AWS Lambda
- Templat Canary CloudWatch Synthetics

OPS08-BP04 Membuat peringatan yang dapat ditindaklanjuti

Sangat penting mendeteksi dan merespons penyimpangan dalam perilaku aplikasi Anda segera. Lebih penting lagi adalah mengenali ketika hasil yang didasarkan pada indikator kinerja utama (KPI) terpapar risiko atau ketika anomali tak terduga muncul. Mendasarkan peringatan pada KPI memastikan bahwa sinyal yang Anda terima berkaitan langsung dengan dampak bisnis atau operasional. Pendekatan terhadap peringatan yang dapat ditindaklanjuti ini mempromosikan respons proaktif dan membantu mempertahankan performa dan keandalan sistem.

Hasil yang diinginkan: Terima peringatan yang tepat waktu, relevan, dan dapat ditindaklanjuti untuk identifikasi dan mitigasi potensi masalah dengan cepat, terutama ketika hasil KPI berisiko.

Antipola umum:

- Mengonfigurasi terlalu banyak peringatan non-kritis, yang mengakibatkan kewalahan.
- Tidak memprioritaskan peringatan berdasarkan KPI, sehingga dampak masalah terhadap bisnis menjadi sulit dipahami.
- Mengabaikan penanganan akar masalah, yang berimbas pada peringatan yang repetitif untuk masalah yang sama.

Manfaat menjalankan praktik terbaik ini:

- Berkurangnya kewalahan akibat peringatan dengan memusatkan perhatian pada peringatan yang dapat ditindaklanjuti dan relevan.
- Waktu aktif dan keandalan sistem yang lebih baik melalui deteksi dan mitigasi masalah secara proaktif.
- Kolaborasi tim yang disempurnakan dan penyelesaian masalah yang lebih cepat melalui integrasi alat-alat peringatan dan komunikasi populer.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Untuk membuat mekanisme peringatan yang efektif, sangat penting untuk menggunakan metrik, log, dan data jejak yang menandai kapan hasil yang didasarkan pada KPI mengandung risiko atau terdapat anomali yang terdeteksi.

Langkah implementasi

1. Tentukan indikator kinerja utama (KPI): Identifikasikan KPI aplikasi Anda. Peringatan harus dikaitkan dengan KPI tersebut agar mencerminkan dampak bisnis secara akurat.
2. Implementasikan deteksi anomali:
   - Gunakan AWS Cost Anomaly Detection: Siapkan AWS Cost Anomaly Detection untuk secara otomatis mendeteksi pola yang tidak biasa, sehingga memastikan peringatan hanya dihasilkan untuk anomali asli.
   - Gunakan Wawasan X-Ray:
     a. Siapkan Wawasan X-Ray untuk mendeteksi anomali dalam data jejak.
     b. Konfigurasikan notifikasi untuk Wawasan X-Ray untuk menerima peringatan tentang masalah yang terdeteksi.
   - Integrasikan dengan DevOps Guru:

   a. Manfaatkan Amazon DevOps Guru untuk kemampuan machine learning-nya dalam mendeteksi anomali operasional pada data yang ada.
   b. Buka pengaturan notifikasi di dalam DevOps Guru untuk menyiapkan peringatan anomali.
3. Implementasikan peringatan yang dapat ditindaklanjuti: Rancang peringatan yang menyediakan informasi yang memadai untuk tindakan cepat.
4. Kurangi kewalahan akibat alarm: Minimalkan peringatan non-kritis. Tim yang kewalahan dengan banyaknya peringatan yang tidak penting dapat menyebabkan terlewatkannya masalah kritis dan mengurangi efektivitas mekanisme peringatan secara keseluruhan.
5. Siapkan alarm komposit: Gunakan alarm komposit Amazon CloudWatch untuk menggabungkan beberapa alarm.
6. Integrasikan dengan alat peringatan: Sertakan alat-alat seperti Ops Genie dan PagerDuty.
7. Libatkan AWS Chatbot Integrasikan AWS Chatbotuntuk mengirimkan peringatan ke Chime, Microsoft Teams, dan Slack.
8. Buat peringatan berdasarkan log: Gunakan filter metrik log di CloudWatch untuk membuat alarm berdasarkan peristiwa log tertentu.
9. Tinjau dan lakukan iterasi: Tinjau dan sempurnakan konfigurasi peringatan secara rutin.

Tingkat upaya untuk rencana implementasi: Sedang.

Sumber daya

Praktik terbaik terkait:

- OPS04-BP01 Identifikasikan indikator performa utama
- OPS04-BP02 Mengimplementasikan telemetri aplikasi
- OPS04-BP03 Mengimplementasikan telemetri pengalaman pengguna
- OPS04-BP04 Mengimplementasikan telemetri dependensi
- OPS04-BP05 Mengimplementasikan penelusuran terdistribusi
- OPS08-BP01 Menganalisis metrik beban kerja
- OPS08-BP02 Menganalisis log beban kerja
- OPS08-BP03 Menganalisis jejak beban kerja

Dokumen terkait:

- Menggunakan Alarm Amazon CloudWatch
- Membuat alarm komposit
- Membuat alarm CloudWatch berdasarkan deteksi anomali
- Notifikasi DevOps Guru
- Notifikasi Wawasan X-Ray
- Pantau, operasikan, dan pecahkan masalah sumber daya AWS Anda dengan ChatOps interaktif
- Panduan Integrasi Amazon CloudWatch | PagerDuty
- Integrasikan OPSGenie dengan Amazon CloudWatch

Video terkait:

- Membuat Alarm Komposit di Amazon CloudWatch
- Ikhtisar AWS Chatbot
- AWS on Air ft. Perintah Mutatif di AWS Chatbot

Contoh terkait:

- Alarm, manajemen insiden, dan remediasi di cloud dengan Amazon CloudWatch
- Tutorial: Membuat aturan Amazon EventBridge yang mengirimkan notifikasi ke AWS Chatbot
- Lokakarya One Observability

OPS08-BP05 Membuat dasbor

Dasbor adalah tampilan yang berpusat pada manusia tentang data telemetri beban kerja Anda. Meskipun menyediakan antarmuka visual yang vital, dasbor tidak boleh menggantikan mekanisme peringatan, melainkan hanya melengkapinya. Ketika dibuat dengan cermat, dasbor tidak hanya dapat menawarkan wawasan cepat tentang kondisi dan kinerja sistem, tetapi juga dapat menyajikan informasi waktu nyata kepada pemangku kepentingan tentang hasil bisnis dan dampak masalah.

Hasil yang diinginkan: Wawasan yang jelas dan dapat ditindaklanjuti tentang kondisi sistem dan bisnis menggunakan representasi visual.

Antipola umum:

- Dasbor yang terlalu rumit dengan terlalu banyak metrik.

- Mengandalkan dasbor tanpa peringatan untuk deteksi anomali.
- Tidak memperbarui dasbor seiring perkembangan beban kerja.

Manfaat menjalankan praktik terbaik ini:

- Visibilitas langsung tentang metrik sistem kritis dan KPI.
- Komunikasi dan pemahaman pemangku kepentingan yang ditingkatkan.
- Wawasan cepat tentang dampak masalah operasional.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Dasbor yang berpusat pada bisnis

Dasbor yang disesuaikan dengan KPI bisnis melibatkan lebih banyak pemangku kepentingan. Meskipun individu ini mungkin tidak tertarik pada metrik sistem, mereka tertarik untuk memahami implikasi bisnis dari angka-angka ini. Dasbor yang berpusat pada bisnis memastikan semua metrik teknis dan operasional yang dipantau dan dianalisis selaras dengan tujuan bisnis menyeluruh. Penyelarasan ini memberikan kejelasan, memastikan semua orang memiliki pemahaman yang sama mengenai hal-hal yang penting dan hal-hal yang tidak penting. Selain itu, dasbor yang menyoroti KPI bisnis cenderung lebih mudah ditindaklanjuti. Pemangku kepentingan dapat dengan cepat memahami kondisi operasi, area yang perlu diperhatikan, dan potensi dampak terhadap hasil bisnis.

Dengan mempertimbangkan hal ini, saat membuat dasbor Anda, pastikan ada keseimbangan antara metrik teknis dan KPI bisnis. Keduanya penting tetapi melayani audiens yang berbeda. Idealnya, Anda harus memiliki dasbor yang memberikan pandangan menyeluruh tentang kondisi dan performa sistem sekaligus menekankan hasil bisnis utama serta implikasinya.

Dasbor Amazon CloudWatch adalah halaman beranda yang dapat dikustomisasi di konsol CloudWatch yang dapat Anda gunakan untuk memantau sumber daya dalam satu tampilan, bahkan sumber daya yang tersebar di Wilayah AWS dan akun yang berbeda-beda.

Langkah implementasi

1. Buat dasbor dasar: Buat dasbor baru di CloudWatch, dan berikan nama yang jelas.
2. Gunakan widget Markdown: Sebelum menjelajahi metrik, gunakan widget Markdown untuk menambahkan konteks tekstual di bagian atas dasbor Anda. Widget ini akan menjelaskan

cakupan dasbor, tingkat pentingnya metrik yang ditampilkan, dan juga dapat diisi dengan tautan ke dasbor serta alat pemecahan masalah lainnya.

3. Buat variabel dasbor: Gabungkan variabel dasbor seperlunya agar tampilan dasbor menjadi dinamis dan fleksibel.
4. Buat widget metrik: Tambahkan widget metrik untuk memvisualisasikan berbagai metrik yang dihasilkan oleh aplikasi Anda, lalu sesuaikan semua widget agar efektif menampilkan kondisi sistem dan hasil bisnis.
5. Kueri Wawasan Log: Manfaatkan CloudWatch Logs Insights untuk mendapatkan metrik yang dapat ditindaklanjuti dari log Anda dan menampilkan wawasan ini di dasbor Anda.
6. Siapkan alarm: Integrasikan alarm CloudWatch ke dasbor agar Anda dapat mengetahui dengan cepat metrik yang melanggar ambang batasnya.
7. Gunakan Contributor Insights: Sertakan CloudWatch Contributor Insights untuk menganalisis bidang dengan kardinalitas tinggi dan mendapatkan pemahaman yang lebih jelas tentang kontributor utama sumber daya Anda.
8. Rancang widget kustom: Untuk kebutuhan spesifik yang tidak terpenuhi oleh widget standar, pertimbangkan untuk membuat widget kustom. Widget kustom dapat menarik dari berbagai sumber data atau menyajikan data dengan cara yang unik.
9. Lakukan iterasi dan sempurnakan: Saat aplikasi Anda berkembang, tinjau kembali dasbor Anda secara teratur untuk memastikan relevansinya.

Sumber daya

Praktik terbaik terkait:

- OPS04-BP01 Identifikasikan indikator performa utama
- OPS08-BP01 Menganalisis metrik beban kerja
- OPS08-BP02 Menganalisis log beban kerja
- OPS08-BP03 Menganalisis jejak beban kerja
- OPS08-BP04 Membuat peringatan yang dapat ditindaklanjuti

Dokumen terkait:

- Membangun Dasbor untuk Visibilitas Operasional
- Menggunakan Dasbor Amazon CloudWatch

Video terkait:

- Membuat Dasbor CloudWatch Lintas Akun & Lintas Wilayah
- AWS re:Invent 2021 - Mendapatkan visibilitas korporasi dengan dasbor operasional AWS Cloud

Contoh terkait:

- Lokakarya One Observability
- Pemantauan Aplikasi dengan Amazon CloudWatch

Memahami kesehatan operasional

Tetapkan, rekam, dan analisis metrik operasi untuk mendapatkan visibilitas terhadap aktivitas tim operasi sehingga Anda dapat mengambil tindakan yang tepat.

Organisasi Anda harus mampu memahami kesehatan operasi Anda dengan mudah. Anda perlu menentukan tujuan bisnis tim operasi Anda, mengidentifikasi indikator kinerja utama yang mencerminkan mereka, menggunakan kemudian mengembangkan metrik berdasarkan hasil operasi untuk mendapatkan wawasan yang berguna. Anda sebaiknya menggunakan metrik-metrik ini untuk mengimplementasikan dasbor dan laporan dengan sudut pandang bisnis dan teknis yang akan membantu para pemimpin dan pemangku kepentingan mengambil keputusan yang matang.

AWS memudahkan penggabungan dan analisis log operasi sehingga Anda dapat menghasilkan metrik, mengetahui status operasi, dan mendapatkan wawasan dari operasi seiring waktu.

Praktik terbaik

- OPS09-BP01 Mengukur sasaran operasi dan KPI dengan metrik
- OPS09-BP02 Mengomunikasikan status dan tren untuk memastikan visibilitas beroperasi
- OPS09-BP03 Meninjau metrik operasi dan memprioritaskan perbaikan

OPS09-BP01 Mengukur sasaran operasi dan KPI dengan metrik

Dapatkan sasaran dan KPI yang menentukan keberhasilan operasi dari organisasi Anda dan pastikan metrik mencerminkan hal ini. Tetapkan garis acuan sebagai titik referensi dan evaluasi kembali secara rutin. Kembangkan mekanisme untuk mengumpulkan metrik-metrik tersebut dari tim untuk dievaluasi.

Hasil yang diinginkan:

- Sasaran dan KPI untuk tim operasi organisasi telah dipublikasikan dan dibagikan.
- Metrik yang mencerminkan KPI ini ditetapkan. Di antara contohnya adalah:
  - Kedalaman antrean tiket atau rata-rata umur tiket
  - Jumlah tiket yang dikelompokkan berdasarkan jenis masalah
  - Waktu yang dihabiskan untuk mengurusi masalah dengan atau tanpa prosedur operasi standar (SOP)
  - Jumlah waktu yang dihabiskan untuk pulih dari push kode yang gagal
  - Volume panggilan

Antipola umum:

- Tenggat waktu deployment tidak terpenuhi karena developer disibukkan dengan tugas pemecahan masalah. Tim pengembangan menuntut lebih banyak personel, tetapi tidak dapat mengukur berapa orang yang mereka butuhkan karena waktu yang tersita tidak dapat diukur.
- Meja Tingkat 1 disiapkan untuk menangani panggilan pengguna. Seiring waktu, makin banyak beban kerja ditambahkan, tetapi tidak ada personel yang dialokasikan ke meja Tingkat 1 tersebut. Kepuasan pelanggan sangat rendah karena waktu panggilan meningkat dan masalah berlarut-larut tanpa penyelesaian, tetapi manajemen tidak melihat indikator permasalahan ini, sehingga tidak ada tindakan yang dilakukan.
- Beban kerja yang bermasalah diserahkan kepada tim operasi terpisah untuk pemeliharaan. Tidak seperti beban kerja lainnya, beban kerja tersebut tidak dilengkapi dengan dokumentasi dan runbook yang tepat. Akibatnya, tim menghabiskan waktu lebih lama untuk memecahkan masalah dan mengurusi kegagalan. Namun, tidak ada metrik yang mendokumentasikan hal ini, sehingga akuntabilitas menjadi sulit.

Manfaat menjalankan praktik terbaik ini: Ketika pemantauan beban kerja menunjukkan status aplikasi dan layanan kita, tim operasi pemantauan memberi pemilik wawasan tentang perubahan pada pemakai beban kerja tersebut, seperti perubahan kebutuhan bisnis. Ukur efektivitas tim-tim tersebut dan evaluasi mereka berdasarkan sasaran bisnis dengan membuat metrik yang dapat mencerminkan status operasi. Metrik dapat menyoroti masalah dukungan atau mengidentifikasi ketika terjadi pergeseran dari target tingkat layanan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Jadwalkan waktu dengan para pemimpin bisnis dan pemangku kepentingan untuk menentukan sasaran layanan secara keseluruhan. Tentukan tugas apa saja yang seharusnya dijalankan oleh berbagai tim operasi dan tantangan apa yang dapat mereka hadapi. Dengan menggunakan hal ini, lakukan curah pendapat indikator kinerja utama (KPI) yang mungkin mencerminkan semua sasaran operasi ini. Indikator tersebut mungkin berupa kepuasan pelanggan, waktu dari konsepsi fitur hingga deployment, waktu penyelesaian masalah rata-rata, dan lain-lain.

Berpatokan pada KPI, identifikasi metrik dan sumber data yang mungkin paling mencerminkan semua sasaran ini. Kepuasan pelanggan dapat berupa kombinasi dari berbagai metrik seperti waktu tunggu atau respons panggilan, skor kepuasan, dan jenis-jenis masalah yang disampaikan. Waktu deployment mungkin merupakan jumlah waktu yang diperlukan untuk pengujian dan deployment, serta perbaikan pasca-deployment yang perlu ditambahkan. Statistik yang menunjukkan waktu yang dihabiskan untuk berbagai jenis masalah (atau jumlah masalah tersebut) dapat memberikan wawasan tentang bagian yang memerlukan upaya tertarget.

Sumber daya

Dokumen terkait:

- Amazon QuickSight - Menggunakan KPI
- Amazon CloudWatch - Menggunakan Metrik
- Membangun Dasbor
- Cara melacak KPI pengoptimalan biaya Anda dengan Dasbor KPI

OPS09-BP02 Mengomunikasikan status dan tren untuk memastikan visibilitas beroperasi

Anda perlu mengetahui keadaan operasi Anda dan arah trennya untuk mengidentifikasi kapan hasil mungkin berisiko, apakah pekerjaan tambahan dapat didukung, atau efek perubahan terhadap tim Anda. Selama peristiwa operasi, halaman status yang dapat dijadikan acuan oleh pengguna dan tim operasi untuk mendapatkan informasi dapat mengurangi tekanan pada saluran komunikasi dan menyebarkan informasi secara proaktif.

Hasil yang diinginkan:

- Pemimpin operasi memiliki wawasan sekilas untuk melihat volume panggilan seperti apa yang sedang dioperasikan oleh tim mereka dan upaya apa yang mungkin sedang dilakukan, seperti deployment.
- Peringatan disebarkan kepada pemangku kepentingan dan komunitas pengguna ketika terjadi dampak terhadap operasi normal.
- Kepemimpinan dan pemangku kepentingan organisasi dapat memeriksa halaman status sebagai respons terhadap peringatan atau dampak, dan memperoleh informasi seputar peristiwa operasional, seperti titik kontak, informasi tiket, dan perkiraan waktu pemulihan.
- Laporan tersedia bagi para pemimpin dan pemangku kepentingan lainnya untuk menunjukkan statistik operasi seperti volume panggilan selama periode waktu tertentu, skor kepuasan pengguna, jumlah tiket tertunda, dan usia mereka.

Antipola umum:

- Terdapat beban kerja yang tidak aktif, sehingga sebuah layanan menjadi tidak tersedia. Volume panggilan melonjak karena para pengguna ingin mengetahui apa yang terjadi. Manajer menambah volume tersebut dengan permintaan informasinya tentang siapa yang mengurusi masalah. Berbagai tim operasi melipatgandakan upaya untuk melakukan penyelidikan.
- Keinginan untuk kemampuan baru menyebabkan beberapa personel dialihkan ke upaya rekayasa. Tidak ada pengisian ulang yang disediakan, dan waktu penyelesaian masalah melonjak. Informasi ini tidak ditangkap, dan pimpinan baru menyadari hal ini setelah beberapa minggu dan pengguna menyampaikan ketidakpuasan.

Manfaat menjalankan praktik terbaik ini: Selama peristiwa operasional yang berdampak pada bisnis, banyak waktu dan tenaga yang bisa terbuang untuk meminta informasi dari berbagai tim yang sedang berusaha memahami situasinya. Dengan membuat halaman status dan dasbor yang disebarluaskan, pemangku kepentingan dapat dengan cepat memperoleh informasi seperti apakah ada masalah yang terdeteksi, siapa yang memimpin penanganan masalah tersebut, atau kapan operasi diperkirakan akan kembali normal. Dengan begitu, anggota tim terhindar dari membuang-buang waktu untuk mengomunikasikan status kepada orang lain dan lebih berkonsentrasi menangani masalah.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Buat dasbor yang menunjukkan metrik utama saat ini untuk tim operasi Anda, dan buat dasbor tersebut mudah diakses oleh pemimpin operasi serta manajemen.

Buat halaman status yang dapat diperbarui dengan cepat untuk menunjukkan apabila insiden atau peristiwa sedang berlangsung, siapa yang bertanggung jawab, dan siapa yang mengoordinasikan respons. Bagikan langkah atau solusi apa pun yang harus dipertimbangkan pengguna di halaman ini, dan sebarkan luaskan lokasinya. Imbau pengguna untuk memeriksa lokasi ini terlebih dahulu ketika mereka dihadapkan dengan masalah yang tidak diketahui.

Kumpulkan dan sediakan laporan yang menunjukkan kondisi operasi dari waktu ke waktu, dan distribusikan hal ini kepada para pemimpin dan pengambil keputusan untuk menggambarkan pekerjaan operasi beserta tantangan dan kebutuhan.

Bagikan kepada tim metrik dan laporan yang paling mencerminkan sasaran dan KPI dan bagian yang paling menerima pengaruhnya dalam mendorong perubahan. Luangkan waktu khusus untuk aktivitas ini untuk meningkatkan pentingnya operasi di dalam tim dan antartim.

Sumber daya

Dokumen terkait:

- Mengukur Kemajuan
- Membangun dasbor untuk visibilitas operasi

Solusi terkait:

- Operasi Data

OPS09-BP03 Meninjau metrik operasi dan memprioritaskan perbaikan

Menyisihkan waktu dan sumber daya khusus untuk meninjau keadaan operasi memastikan bahwa pelayanan lini bisnis sehari-hari tetap menjadi prioritas. Kumpulkan para pemimpin operasi dan pemangku kepentingan untuk secara rutin meninjau metrik, menegaskan kembali atau memodifikasi sasaran dan tujuan, dan memprioritaskan perbaikan.

Hasil yang diinginkan:

- Para pemimpin operasi dan staf secara rutin bertemu untuk meninjau metrik selama periode pelaporan tertentu. Tantangan dikomunikasikan, keberhasilan dirayakan, dan pelajaran yang dipetik dibagikan.
- Pemangku kepentingan dan pemimpin bisnis secara rutin diberi pengarahan tentang keadaan operasi dan diminta memberikan masukan mengenai sasaran, KPI, dan inisiatif masa depan. Kompromi antara pelayanan, operasi, dan pemeliharaan dibahas dan dimasukkan ke dalam konteks.

Antipola umum:

- Sebuah produk baru diluncurkan, tetapi tim operasi Tingkat 1 dan Tingkat 2 tidak cukup terlatih untuk mendukung atau diberikan staf tambahan. Metrik yang menunjukkan penurunan waktu resolusi tiket dan peningkatan volume insiden tidak terlihat oleh para pemimpin. Tindakan diambil beberapa minggu kemudian ketika jumlah langganan mulai turun karena ketidakpuasan pengguna yang beralih ke platform lain.
- Proses manual untuk melakukan pemeliharaan pada beban kerja telah berlangsung sejak lama. Meskipun sudah ada keinginan untuk melakukan otomatisasi, prioritas yang diberikan rendah mengingat rendahnya signifikansi sistem. Namun seiring waktu, sistem menjadi makin penting dan sekarang proses manual ini menyita sebagian besar waktu operasional. Tidak ada sumber daya yang dijadwalkan untuk menyediakan peningkatan peralatan untuk operasi, sehingga menyebabkan kelelahan pada staf saat beban kerja meningkat. Pimpinan menyadari hal ini setelah ada laporan bahwa para staf beralih ke kompetitor.

Manfaat menjalankan praktik terbaik ini: Beberapa organisasi kesulitan untuk mengalokasikan waktu dan perhatian yang sama untuk pemberian layanan dan produk atau penawaran baru. Ketika ini terjadi, lini bisnis dapat menderita karena tingkat layanan yang diharapkan perlahan-lahan memburuk. Alasannya adalah operasi tidak berubah dan berkembang sesuai dengan perkembangan bisnis, dan bisa segera tertinggal. Tanpa peninjauan rutin pada wawasan yang dikumpulkan oleh operasi, risiko terhadap bisnis mungkin baru terlihat ketika semua sudah terlambat. Dengan pengalokasian waktu untuk meninjau metrik dan prosedur baik di antara staf operasi maupun dengan pimpinan, peran penting yang dimiliki oleh operasi terus dapat dilihat, dan risiko dapat diidentifikasi jauh sebelum mencapai tingkat kritis. Tim operasi mendapatkan wawasan yang lebih baik tentang perubahan dan inisiatif bisnis yang akan datang, sehingga upaya proaktif dapat dilakukan. Visibilitas kepemimpinan ke dalam metrik operasi menunjukkan peran yang dimiliki oleh tim operasional dalam kepuasan pelanggan, baik internal maupun eksternal, dan memungkinkan mereka mempertimbangkan pilihan prioritas dengan lebih baik, atau memastikan bahwa operasional

memiliki waktu dan sumber daya untuk berubah dan berkembang seiring munculnya inisiatif bisnis dan beban kerja baru.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Luangkan waktu khusus untuk meninjau metrik operasi antara pemangku kepentingan dan tim operasional dan meninjau data laporan. Masukkan laporan-laporan tersebut ke dalam konteks tujuan dan sasaran organisasi untuk menentukan apakah semuanya terpenuhi. Identifikasikan sumber ambiguitas di mana sasaran tidak jelas, atau di mana mungkin ada ketidaksesuaian antara apa yang diminta dan apa yang diberikan.

Identifikasikan di mana waktu, personel, dan alat dapat membantu mencapai hasil operasi. Tentukan KPI mana yang akan menerima dampaknya dan target kesuksesan apa yang harus dimiliki. Tinjau ulang secara rutin untuk memastikan operasi memiliki sumber daya yang memadai untuk mendukung lini bisnis.

Sumber daya

Dokumen terkait:

- Amazon Athena
- Metrik Amazon CloudWatch dan referensi dimensi
- Amazon QuickSight
- AWS Glue
- AWS Glue Data Catalog
- Kumpulkan metrik dan log dari Instans Amazon EC2 serta server on-premise dengan Agen Amazon CloudWatch
- Menggunakan metrik Amazon CloudWatch

Merespons peristiwa

Anda harus mengantisipasi peristiwa operasional, baik yang terencana (seperti promo penjualan, deployment, dan uji kegagalan) dan yang tidak terencana (seperti lonjakan pemanfaatan dan kegagalan komponen). Anda harus menggunakan runbook dan buku pedoman yang Anda miliki

untuk menghadirkan hasil yang konsisten ketika merespons pemberitahuan. Pemberitahuan yang ditetapkan harus dimiliki oleh sebuah peran atau tim yang bertanggung jawab atas respons dan eskalasi. Anda juga perlu mengetahui dampak komponen sistem Anda terhadap bisnis dan gunakan untuk menargetkan upaya saat diperlukan. Anda harus melakukan analisis akar masalah (RCA) setelah peristiwa, lalu mencegah kembali terjadinya kegagalan atau mendokumentasikan pemecahan masalah.

AWS menyederhanakan respons peristiwa Anda dengan menyediakan alat-alat yang mendukung semua aspek beban kerja dan operasi Anda dalam bentuk kode. Alat-alat ini memungkinkan Anda untuk membuat skrip respons terhadap peristiwa operasional dan memulai inisiasi skrip tersebut ketika merespons data pemantauan.

Di AWS, Anda dapat mempercepat waktu pemulihan dengan mengganti komponen yang gagal dengan versi komponen yang diketahui baik, alih-alih mencoba untuk memperbaikinya. Lalu Anda dapat menjalankan analisis terhadap sumber daya yang gagal tersebut di luar jaringan.

Praktik terbaik

- OPS10-BP01 Menggunakan proses untuk manajemen peristiwa, insiden, dan masalah
- OPS10-BP02 Menjalankan proses untuk setiap peringatan
- OPS10-BP03 Memprioritaskan kejadian operasional berdasarkan dampaknya terhadap bisnis
- OPS10-BP04 Tetapkan jalur eskalasi
- OPS10-BP05 Membuat rencana komunikasi pelanggan untuk gangguan
- OPS10-BP06 Mengomunikasikan status melalui dasbor
- OPS10-BP07 Otomatiskan respons terhadap peristiwa

OPS10-BP01 Menggunakan proses untuk manajemen peristiwa, insiden, dan masalah

Organisasi Anda memiliki proses untuk menangani peristiwa, insiden, dan masalah. Peristiwa adalah hal-hal yang terjadi dalam beban kerja Anda, tetapi mungkin tidak memerlukan intervensi. Insiden adalah peristiwa yang memerlukan intervensi. Masalah adalah peristiwa berulang yang memerlukan intervensi atau tidak dapat diselesaikan. Anda memerlukan proses untuk mengurangi dampak peristiwa ini pada bisnis Anda dan memastikan bahwa Anda merespons dengan tepat.

Ketika insiden dan masalah terjadi pada beban kerja Anda, Anda memerlukan proses untuk menanganinya. Bagaimana Anda akan mengomunikasikan status peristiwa dengan pemangku

kepentingan? Siapa yang mengawasi pelaksanaan respons? Apa alat yang Anda gunakan untuk memitigasi peristiwa? Ini adalah contoh dari beberapa pertanyaan yang perlu Anda jawab untuk memiliki proses respons yang solid.

Proses harus didokumentasikan di lokasi sentral dan tersedia bagi siapa saja yang terlibat dalam beban kerja Anda. Jika Anda tidak memiliki wiki atau penyimpanan dokumen sentral, repositori kontrol versi dapat digunakan. Anda akan terus memperbarui rencana ini seiring berkembangnya proses Anda.

Masalah merupakan kandidat untuk otomatisasi. Peristiwa ini mengambil waktu Anda yang seharusnya dihabiskan untuk berinovasi. Mulailah dengan membangun proses berulang untuk memitigasi masalah. Seiring waktu, fokuslah untuk mengotomatiskan mitigasi atau memperbaiki masalah mendasar. Tindakan ini akan membebaskan waktu yang kemudian dapat dihabiskan untuk melakukan peningkatan dalam beban kerja Anda.

Hasil yang diinginkan: Organisasi Anda memiliki proses untuk menangani peristiwa, insiden, dan masalah. Proses ini didokumentasikan dan disimpan di lokasi sentral. Dokumentasinya akan diperbarui seiring proses ini berubah.

Antipola umum:

- Sebuah insiden terjadi pada akhir pekan dan teknisi yang berjaga tidak tahu harus melakukan tindakan apa.
- Seorang pelanggan mengirimi Anda email bahwa aplikasi Anda tidak beroperasi. Anda melakukan booting ulang server untuk memperbaikinya. Hal ini sering terjadi.
- Ada insiden yang mengharuskan banyak tim bekerja secara independen untuk mencoba menyelesaikannya.
- Deployment terjadi dalam beban kerja Anda tanpa didokumentasikan.

Manfaat menjalankan praktik terbaik ini:

- Anda memiliki jejak audit peristiwa dalam beban kerja Anda.
- Waktu Anda untuk pulih dari insiden berkurang.
- Anggota tim dapat menyelesaikan insiden dan masalah secara konsisten.
- Ada upaya yang lebih terkonsolidasi ketika menyelidiki sebuah insiden.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Menerapkan praktik terbaik ini berarti Anda melacak peristiwa beban kerja. Anda memiliki proses untuk menangani insiden dan masalah. Proses ini didokumentasikan, dibagikan, dan sering diperbarui. Masalah diidentifikasi, diprioritaskan, dan diperbaiki.

Contoh pelanggan

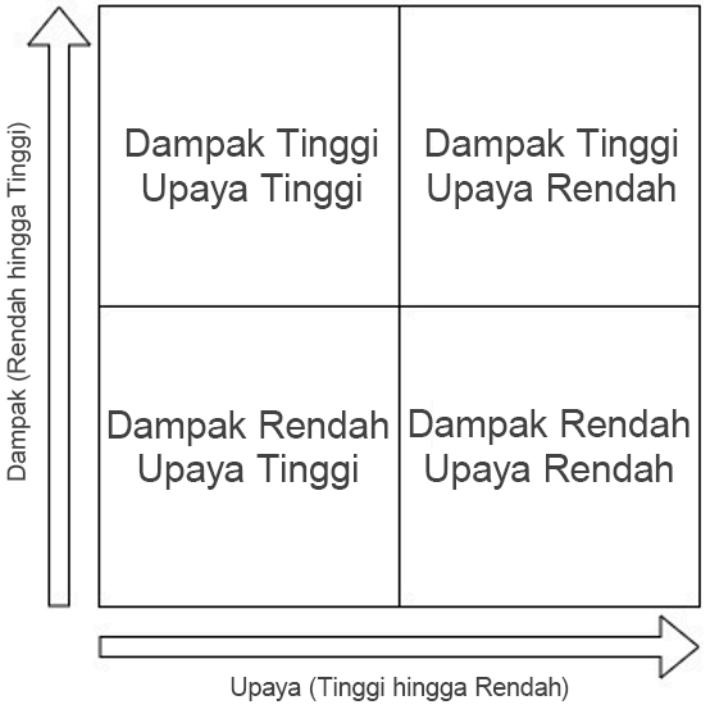
AnyCompany Retail mengkhususkan sebuah bagian dari wiki internal mereka untuk proses penanganan manajemen peristiwa, insiden, dan masalah. Semua peristiwa dikirim ke Amazon EventBridge. Masalah diidentifikasi sebagai OpsItems di AWS Systems Manager OpsCenter dan diprioritaskan untuk diperbaiki, sehingga mengurangi tenaga kerja yang tidak terdiferensiasi. Seiring proses ini berubah, dokumentasinya diperbarui di wiki internal mereka. Mereka menggunakan AWS Systems Manager Incident Manager untuk mengelola insiden dan mengoordinasikan upaya mitigasi.

Langkah implementasi

1. Peristiwa
   - Lacak peristiwa yang terjadi dalam beban kerja Anda, meskipun tidak diperlukan intervensi manusia.
   - Bekerja sama dengan pemangku kepentingan beban kerja untuk mengembangkan daftar peristiwa yang harus dilacak. Beberapa contohnya adalah deployment yang diselesaikan atau patching yang berhasil.
   - Anda dapat menggunakan layanan seperti Amazon EventBridge atau Amazon Simple Notification Service untuk menghasilkan peristiwa kustom untuk pelacakan.
2. Insiden
   - Mulailah dengan mendefinisikan rencana komunikasi untuk insiden. Pemangku kepentingan mana yang harus diinformasikan? Bagaimana Anda akan terus menginformasikan mereka? Siapa yang mengawasi upaya koordinasi? Kami merekomendasikan untuk membuat saluran obrolan internal untuk komunikasi dan koordinasi.
   - Tentukan jalur eskalasi untuk tim yang mendukung beban kerja Anda, terutama jika tim ini tidak memiliki rotasi jaga. Berdasarkan tingkat dukungan Anda, Anda juga dapat mengajukan kasus ke AWS Support.
   - Buat buku playbook untuk menyelidiki insiden. Playbook Ini harus berisi rencana komunikasi dan langkah penyelidikan yang mendetail. Sertakan tindakan memeriksa AWS Health Dashboard dalam penyelidikan Anda.

- Dokumentasikan rencana respons insiden Anda. Komunikasikan rencana manajemen insiden agar pelanggan internal dan eksternal memahami aturan pelibatan dan apa yang diharapkan dari mereka. Latih anggota tim Anda tentang cara menggunakannya.
- Pelanggan dapat menggunakan Incident Manager untuk mengatur dan mengelola rencana respons insiden mereka.
- Pelanggan Enterprise Support dapat meminta Lokakarya Manajemen Insiden dari Manajer Akun Teknis mereka. Lokakarya berpemandu ini akan menguji rencana respons insiden yang ada dan membantu Anda mengidentifikasi area yang perlu ditingkatkan.

3. Masalah
   - Masalah harus diidentifikasi dan dilacak dalam sistem ITSM Anda.
   - Identifikasi semua masalah yang diketahui dan prioritaskan berdasarkan tingkat upaya perbaikan dan dampak pada beban kerja.



   - Selesaikan masalah yang berdampak tinggi dan memerlukan tingkat upaya yang rendah terlebih dahulu. Setelah masalah tersebut diselesaikan, lanjutkan ke masalah yang termasuk dalam kuadran upaya rendah berdampak rendah.
   - Anda dapat menggunakan Systems Manager OpsCenter untuk mengidentifikasi masalah ini, menyediakan runbook yang sesuai, dan melacaknya.

Tingkat upaya untuk rencana implementasi: Sedang. Anda memerlukan proses dan alat untuk menerapkan praktik terbaik ini. Dokumentasikan proses Anda dan sediakan dokumentasi ini untuk siapa saja yang terkait dengan beban kerja. Perbarui dokumentasi ini secara rutin. Anda memiliki proses untuk mengelola dan memitigasi atau memperbaiki masalah.

Sumber daya

Praktik terbaik terkait:

- OPS07-BP03 Menggunakan runbook untuk menjalankan prosedur: Masalah yang diketahui memerlukan runbook terkait agar upaya mitigasinya konsisten.
- OPS07-BP04 Menggunakan buku panduan untuk menyelidiki masalah: Insiden harus diselidiki menggunakan playbook.
- OPS11-BP02 Menjalankan analisis setelah insiden: Selalu lakukan pemeriksaan pascainsiden setelah Anda pulih dari suatu insiden.

Dokumen terkait:

- Atlassian - Manajemen insiden di era DevOps
- Panduan Respons Insiden Keamanan AWS
- Manajemen Insiden di Era DevOps dan SRE
- PagerDuty - Apa itu Manajemen Insiden?

Video terkait:

- AWS re:Invent 2020: Manajemen insiden di organisasi terdistribusi
- AWS re:Invent 2021 - Membangun aplikasi generasi baru dengan arsitektur berbasis peristiwa
- AWS Mendukung Anda | Latihan Diskusi Menjelajahi Manajemen Insiden
- AWS Systems Manager Incident Manager - Lokakarya Virtual AWS
- AWS What's Next bersama Incident Manager | Acara AWS

Contoh terkait:

- Lokakarya Alat Manajemen dan Tata Kelola AWS - OpsCenter
- Layanan Proaktif AWS – Lokakarya Manajemen Insiden

- Membangun aplikasi berbasis peristiwa dengan Amazon EventBridge
- Membangun arsitektur berbasis peristiwa di AWS

Layanan terkait:

- Amazon EventBridge
- Amazon SNS
- AWS Health Dashboard
- AWS Systems Manager Incident Manager
- AWS Systems Manager OpsCenter

OPS10-BP02 Menjalankan proses untuk setiap peringatan

Tetapkan respons (runbook atau buku pedoman) dengan baik, dengan pemilik yang teridentifikasi secara khusus, untuk peristiwa apa pun yang diatur peringatannya. Ini memastikan respons yang efektif dan cepat terhadap peristiwa operasi dan mencegah peristiwa yang dapat ditindaklanjuti dihalangi oleh notifikasi yang kurang bernilai.

Antipola umum:

- Sistem pemantauan memberikan aliran koneksi yang disetujui bersama dengan pesan lainnya. Volume pesan sangat besar sehingga Anda melewatkan pesan kesalahan berkala yang perlu diintervensi.
- Anda menerima peringatan bahwa situs web terhenti. Tidak ada proses yang ditentukan jika hal seperti ini terjadi. Anda dipaksa untuk melakukan tindakan ad hoc untuk mendiagnosis dan menyelesaikan masalah. Mengembangkan proses ini seiring berjalannya waktu akan memperpanjang waktu pemulihan.

Manfaat menerapkan praktik terbaik ini: Dengan memperingatkan hanya ketika tindakan diperlukan, Anda mencegah peringatan bernilai rendah menutupi peringatan bernilai tinggi. Dengan memiliki proses untuk setiap peringatan yang dapat ditindaklanjuti, Anda mengaktifkan respons yang konsisten dan cepat terhadap peristiwa di lingkungan Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Proses per peringatan: Peristiwa apa pun yang Anda aktifkan peringatannya harus memiliki respons (runbook atau buku pedoman) yang jelas dengan pemilik yang teridentifikasi secara khusus (misalnya, individu, tim, atau peran) yang bertanggung jawab atas penyelesaian yang berhasil. Kinerja respons dapat diotomatiskan atau dilakukan oleh tim lain tetapi pemiliknya bertanggung jawab untuk memastikan proses memberikan hasil yang diharapkan. Dengan memiliki proses ini, Anda memastikan respons yang efektif dan cepat terhadap peristiwa operasi dan mencegah peristiwa yang dapat ditindaklanjuti dihalangi oleh notifikasi yang kurang bernilai. Misalnya, penskalaan otomatis dapat diterapkan untuk menskalakan front end web, tetapi tim operasi mungkin bertanggung jawab untuk memastikan bahwa aturan dan batas penskalaan otomatis sesuai untuk kebutuhan beban kerja.

Sumber daya

Dokumen terkait:

- Fitur Amazon CloudWatch
- Apa itu Amazon CloudWatch Events?

Video terkait:

- Build a Monitoring Plan

OPS10-BP03 Memprioritaskan kejadian operasional berdasarkan dampaknya terhadap bisnis

Ketika ada beberapa kejadian yang memerlukan intervensi, pastikan untuk mengatasi kejadian yang paling signifikan terhadap bisnis terlebih dahulu. Dampak dapat termasuk kematian atau cedera fisik, kerugian finansial, atau rusaknya reputasi dan kepercayaan.

Antipola umum:

- Anda menerima permintaan dukungan untuk menambahkan konfigurasi printer bagi pengguna. Saat sedang menangani masalah tersebut, Anda menerima permintaan dukungan yang menyatakan bahwa situs retail terhenti. Setelah menyelesaikan konfigurasi pencetak untuk pengguna, Anda mulai menangani masalah yang dialami situs web.

- Anda menerima pemberitahuan bahwa sistem pembayaran dan situs web retail Anda terhenti. Anda tidak tahu mana masalah yang harus diprioritaskan.

Manfaat menerapkan praktik terbaik ini: Dengan memprioritaskan insiden yang dampaknya paling besar terhadap bisnis, Anda dapat menetapkan manajemen untuk dampak tersebut.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Prioritaskan peristiwa operasional berdasarkan dampaknya terhadap bisnis: Ketika ada beberapa kejadian yang memerlukan intervensi, atasi kejadian yang paling signifikan terhadap bisnis terlebih dahulu. Dampak dapat termasuk kematian atau cedera fisik, kerugian finansial, atau rusaknya reputasi atau kepercayaan.

OPS10-BP04 Tetapkan jalur eskalasi

Tetapkan jalur eskalasi di runbook dan playbook Anda, termasuk apa yang memicu eskalasi, dan prosedur untuk eskalasi. Secara spesifik identifikasi pemilik untuk setiap tindakan guna memastikan respons yang efektif dan tepat waktu terhadap peristiwa operasi.

Identifikasi ketika keputusan manusia diperlukan sebelum tindakan diambil. Bekerja samalah dengan pengambil keputusan untuk mengambil keputusan tersebut lebih awal, dan untuk mendapatkan terlebih dulu persetujuan atas tindakan, sehingga MTTR tidak menjadi lebih lama karena menunggu respons.

Antipola umum:

- Situs retail Anda tidak berfungsi. Anda tidak memahami runbook untuk memulihkan situs itu. Anda mulai menelepon kolega dengan harapan seseorang akan dapat membantu Anda.
- Anda menerima kasus permintaan dukungan untuk aplikasi yang tidak dapat dijangkau. Anda tidak memiliki izin untuk administrasi sistem. Anda tidak tahu siapa yang memilikinya. Anda berusaha menghubungi pemilik sistem yang membuka kasus tersebut dan tidak mendapatkan respons. Anda tidak memiliki kontak untuk sistem dan kolega Anda tidak tahu.

Manfaat menerapkan praktik terbaik ini: Dengan menetapkan eskalasi, pemicu untuk eskalasi, dan prosedur untuk eskalasi, Anda memungkinkan penambahan sumber daya secara sistematis ke insiden dengan tingkat yang sesuai untuk dampaknya.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Tetapkan jalur eskalasi: Tetapkan jalur eskalasi di runbook dan playbook Anda, termasuk apa yang memicu eskalasi, dan prosedur untuk eskalasi. Contohnya, eskalasi masalah dari rekayasawan dukungan ke rekayasawan dukungan senior ketika runbook tidak dapat menyelesaikan masalah, atau ketika jangka waktu yang ditetapkan sebelumnya telah lewat. Contoh lain dari jalur eskalasi yang benar adalah dari rekayasawan dukungan senior ke tim pengembangan untuk beban kerja ketika playbook tidak dapat mengidentifikasi jalur ke perbaikan, atau ketika jangka waktu yang ditetapkan sebelumnya telah lewat. Secara spesifik identifikasi pemilik untuk setiap tindakan guna memastikan respons yang efektif dan tepat waktu terhadap peristiwa operasi. Eskalasi dapat mencakup pihak ketiga. Contohnya, penyedia konektivitas jaringan atau vendor perangkat lunak. Eskalasi dapat mencakup pengambil keputusan resmi yang diidentifikasi untuk sistem yang terkena dampak.

OPS10-BP05 Membuat rencana komunikasi pelanggan untuk gangguan

Buat dan uji rencana komunikasi tentang gangguan sistem yang dapat Anda andalkan agar pelanggan dan pemangku kepentingan Anda selalu mendapatkan informasi selama terjadi gangguan. Komunikasikan langsung ke pengguna Anda baik ketika layanan yang mereka gunakan terkena dampaknya, dan ketika layanan kembali normal.

Hasil yang diinginkan:

- Anda memiliki rencana komunikasi untuk situasi yang berbeda-beda, dari pemeliharaan terjadwal hingga kegagalan besar tak terduga, termasuk pemberlakuan rencana pemulihan bencana.
- Dalam komunikasi Anda, Anda memberikan informasi yang jelas dan transparan tentang masalah sistem untuk membantu pelanggan menghindari meragukan performa sistem mereka.
- Anda menggunakan halaman status dan pesan kesalahan kustom untuk mengurangi lonjakan permintaan di pusat bantuan dan menjaga agar pengguna tetap mendapatkan informasi.
- Rencana komunikasi diuji secara teratur untuk memverifikasi bahwa rencana tersebut memiliki performa sesuai yang dimaksud ketika gangguan yang sesungguhnya terjadi.

Antipola umum:

- Gangguan beban kerja terjadi tetapi Anda tidak memiliki rencana komunikasi. Pengguna membuat sistem pengajuan masalah Anda kewalahan karena terlalu banyak permintaan akibat tidak adanya informasi tentang gangguan.
- Anda mengirimkan pemberitahuan lewat email kepada pengguna selama gangguan berlangsung. Pemberitahuan tersebut tidak berisi jangka waktu untuk pemulihan layanan sehingga pengguna tidak dapat membuat rencana untuk mengatasi gangguan.
- Terdapat rencana komunikasi untuk gangguan tetapi tidak pernah diuji. Gangguan terjadi dan rencana komunikasi gagal karena langkah yang sangat penting terlewatkan, dan hal tersebut seharusnya dapat diketahui dalam pengujian.
- Selama gangguan, Anda mengirimkan kepada pengguna pemberitahuan yang disertai terlalu banyak informasi teknis mendetail dan informasi dalam NDA AWS Anda.

Manfaat menjalankan praktik terbaik ini:

- Mempertahankan komunikasi selama gangguan memastikan pelanggan diberi visibilitas tentang progres masalah dan perkiraan waktu resolusinya.
- Mengembangkan rencana komunikasi yang jelas akan memverifikasi bahwa pelanggan dan pengguna akhir Anda mendapatkan informasi sehingga mereka dapat mengambil langkah tambahan yang diperlukan untuk memitigasi dampak gangguan.
- Dengan komunikasi yang tepat dan peningkatan kesadaran akan gangguan terencana dan tidak terencana, Anda dapat meningkatkan tingkat kepuasan pelanggan, membatasi reaksi yang tidak diinginkan, dan mendorong retensi pelanggan.
- Komunikasi gangguan secara tepat waktu dan transparan meningkatkan keyakinan dan menjalin kepercayaan yang diperlukan untuk memelihara hubungan antara Anda dan pelanggan.
- Strategi komunikasi yang terbukti selama gangguan atau krisis akan mengurangi spekulasi dan gosip yang dapat menghalangi kemampuan Anda untuk pulih.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

Rencana komunikasi yang selalu memberikan informasi kepada pelanggan Anda selama gangguan bersifat holistik dan meliputi beberapa antarmuka, termasuk halaman kesalahan yang dilihat pelanggan, pesan kesalahan API kustom, spanduk status sistem, dan halaman status kondisi. Jika sistem Anda mencakup pengguna terdaftar, Anda dapat berkomunikasi melalui saluran pesan seperti

email, SMS, atau notifikasi push untuk mengirimkan konten pesan yang dipersonalisasi ke pelanggan Anda.

Alat komunikasi pelanggan

Sebagai garis pertahanan pertama, aplikasi seluler dan web harus memberikan pesan kesalahan yang ramah dan informatif selama gangguan serta memiliki kemampuan untuk mengarahkan ulang lalu lintas ke halaman status. Amazon CloudFront adalah jaringan pengiriman konten (CDN) terkelola penuh yang disertai kemampuan untuk membuat dan menghadirkan konten kesalahan kustom. Halaman kesalahan kustom di CloudFront merupakan lapisan pertama yang bagus untuk pesan pelanggan terkait gangguan di tingkat komponen. CloudFront dapat juga menyederhanakan pengelolaan dan pengaktifan halaman status untuk menangkap semua permintaan selama gangguan terencana atau tidak terencana.

Pesan kesalahan API kustom dapat membantu mendeteksi dan mengurangi dampak ketika gangguan terisolasi ke layanan terpisah. Amazon API Gateway memungkinkan Anda mengonfigurasi respons kustom untuk API REST Anda. Hal ini memungkinkan Anda memberikan pesan yang jelas dan bermakna kepada konsumen API ketika API Gateway tidak dapat menjangkau layanan backend. Pesan kustom juga dapat digunakan untuk mendukung konten spanduk gangguan dan pemberitahuan ketika fitur sistem tertentu mengalami penurunan kualitas akibat gangguan di tingkat layanan.

Pesan langsung adalah pesan pelanggan jenis paling personal. Amazon Pinpoint adalah layanan terkelola untuk komunikasi multi-saluran yang dapat diskalakan. Amazon Pinpoint memungkinkan Anda membangun kampanye yang dapat menyiarkan pesan secara luas ke seluruh pelanggan yang terkena dampak melalui SMS, email, pesan suara, notifikasi push, atau saluran kustom yang Anda tentukan. Ketika Anda mengelola pesan dengan Amazon Pinpoint, kampanye pesan dibuat dengan baik, dapat diuji, dan dapat diterapkan secara cerdas pada segmen pelanggan yang ditarget. Setelah dibuat, kampanye dapat dijadwalkan atau dipicu oleh peristiwa dan kampanya dapat diuji dengan mudah.

Contoh pelanggan

Ketika beban kerja terganggu, AnyCompany Retail mengirimkan pemberitahuan lewat email ke pengguna mereka. Email tersebut menerangkan fungsionalitas bisnis apa yang terganggu dan memberikan perkiraan yang realistis tentang kapan layanan akan pulih. Selain itu, mereka memiliki halaman status yang menunjukkan informasi dalam waktu nyata tentang kondisi beban kerja mereka. Rencana komunikasi diuji dalam lingkungan pengembangan dua kali per tahun untuk memvalidasi keefektifannya.

Langkah implementasi

1. Tentukan saluran komunikasi untuk strategi pesan Anda. Pertimbangkan aspek arsitektur dari aplikasi Anda dan tentukan strategi terbaik untuk memberikan umpan balik kepada pelanggan. Hal ini dapat mencakup satu atau lebih strategi panduan yang dijelaskan, termasuk halaman status dan kesalahan, respons kesalahan API kustom, atau pesan langsung.
2. Desain halaman status untuk aplikasi Anda. Jika Anda telah menentukan bahwa halaman kesalahan kustom atau status sesuai untuk pelanggan Anda, Anda harus mendesain konten dan pesan Anda untuk halaman tersebut. Halaman kesalahan menjelaskan kepada pengguna mengapa aplikasi tidak tersedia, kapan aplikasi mungkin tersedia lagi, dan apa yang dapat mereka lakukan sementara ini. Jika aplikasi Anda menggunakan Amazon CloudFront Anda dapat menampilkan respons kesalahan kustom atau menggunakan Lambda di Edge untuk menerjemahkan kesalahan dan menulis ulang konten halaman. CloudFront juga memungkinkan penukaran destinasi dari konten aplikasi Anda ke asal konten Amazon S3 statis yang berisi halaman status gangguan atau pemeliharaan Anda.
3. Desain status kesalahan API yang benar untuk layanan Anda. Pesan kesalahan yang dihasilkan oleh API Gateway ketika layanan backend tidak dapat dicapainya, serta pengecualian tingkat layanan, mungkin tidak berisi pesan yang ramah dan sesuai untuk ditampilkan ke pengguna akhir. Tanpa harus membuat perubahan kode pada layanan backend Anda, Anda dapat mengonfigurasi API Gateway respons kesalahan kustom untuk memetakan kode respons HTTP ke pesan kesalahan API yang dikurasi.
4. Desain pesan dari perspektif bisnis sehingga pesan relevan untuk pengguna akhir sistem Anda dan tidak berisi informasi teknis mendetail. Pertimbangkan audiensi Anda dan sesuaikan pesan Anda. Contohnya, Anda mungkin mengarahkan pengguna internal ke proses manual atau solusi alternatif yang memanfaatkan sistem pengganti. Pengguna eksternal dapat diminta untuk menunggu sampai sistem pulih, atau berlangganan pengiriman pembaruan informasi untuk menerima pemberitahuan segera setelah sistem pulih. Tentukan pesan yang disetujui untuk beberapa skenario, termasuk gangguan tak terduga, pemeliharaan terencana, dan kegagalan sistem parsial di mana fitur tertentu mungkin mengalami penurunan kualitas atau tidak tersedia.
5. Buat sebagai templat dan otomatiskan pesan Anda untuk pelanggan. Setelah Anda membuat konten pesan, Anda dapat menggunakan Amazon Pinpoint atau alat lain untuk mengotomatiskan kampanye pesan Anda. Dengan Amazon Pinpoint Anda dapat membuat segmen pelanggan target untuk pengguna spesifik yang terpengaruh dan mengubah pesan menjadi templat. Tinjau tutorial Amazon Pinpoint untuk memahami cara membuat kampanye pesan.
6. Hindari kemampuan pesan dengan penggabungan erat di sistem yang dilihat pelanggan Anda. Strategi pesan Anda tidak boleh memiliki dependensi keras pada layanan atau penyimpanan data

sistem untuk memverifikasi bahwa Anda bisa sukses mengirimkan pesan ketika Anda mengalami gangguan. Pertimbangkan untuk membuat kemampuan mengirimkan pesan dari lebih dari satu Zona Ketersediaan atau Wilayah untuk ketersediaan pesan. Jika Anda menggunakan layanan AWS untuk mengirimkan pesan, manfaatkan operasi bidang data daripada operasi bidang kendali untuk memunculkan pesan Anda.

Tingkat upaya untuk rencana implementasi: Tinggi. Mengembangkan rencana komunikasi, dan mekanisme untuk mengirimkannya, dapat memerlukan upaya yang cukup besar.

Sumber daya

Praktik terbaik terkait:

- OPS07-BP03 Menggunakan runbook untuk menjalankan prosedur - Rencana komunikasi Anda harus memiliki runbook yang terkait dengannya sehingga personel Anda tahu cara merespons.
- OPS11-BP02 Menjalankan analisis setelah insiden - Setelah gangguan, lakukan analisis pasca-insiden guna mengidentifikasi mekanisme untuk mencegah gangguan lain.

Dokumen terkait:

- Pola Penanganan Kesalahan di Amazon API Gateway dan AWS Lambda
- Respons Amazon API Gateway

Contoh terkait:

- Dasbor AWS Health
- Ringkasan Peristiwa Layanan AWS di Wilayah Virginia Utara (US-EAST-1)

Layanan terkait:

- AWS Support
- Perjanjian Pelanggan AWS
- Amazon CloudFront
- Amazon API Gateway
- Amazon Pinpoint
- Amazon S3

OPS10-BP06 Mengomunikasikan status melalui dasbor

Menyediakan dasbor yang disesuaikan untuk audiens target mereka (misalnya, tim teknis internal, pimpinan, dan pelanggan) guna mengomunikasikan status operasi bisnis saat ini dan memberikan metrik kepentingan.

Anda dapat membuat dasbor menggunakan Dasbor Amazon CloudWatch dengan halaman beranda yang dapat disesuaikan di konsol CloudWatch. Dengan layanan kecerdasan bisnis seperti Amazon QuickSight Anda dapat membuat dan memublikasikan dasbor interaktif yang menampilkan kondisi operasional dan beban kerja Anda (misalnya, tingkat pesanan, pengguna terhubung, dan waktu transaksi). Buat Dasbor yang memberikan tampilan tingkat bisnis dan sistem mengenai metrik Anda.

Antipola umum:

- Atas permintaan, Anda menjalankan laporan tentang pemanfaatan aplikasi Anda saat ini untuk manajemen.
- Selama insiden, Anda dihubungi setiap dua puluh menit oleh pemilik sistem yang ingin mengetahui apakah insiden sudah teratasi.

Manfaat menerapkan praktik terbaik ini: Dengan membuat dasbor, Anda mengaktifkan akses layanan mandiri untuk pelanggan Anda agar mereka mengetahui jika mereka harus melakukan suatu tindakan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Komunikasikan status melalui dasbor: Sediakan dasbor yang disesuaikan untuk audiens target mereka (misalnya, tim teknis internal, pimpinan, dan pelanggan) guna mengomunikasikan status operasi bisnis saat ini dan menyediakan metrik kepentingan. Menyediakan opsi layanan mandiri untuk informasi status dapat mengurangi disrupsi permintaan penanganan status dari tim operator lapangan. Contohnya termasuk dasbor Amazon CloudWatch dan AWS Health Dashboard.
  - Dasbor CloudWatch membuat dan menggunakan tampilan metrik yang disesuaikan

Sumber daya

Dokumen terkait:

- Amazon QuickSight
- Dasbor CloudWatch membuat dan menggunakan tampilan metrik yang disesuaikan

OPS10-BP07 Otomatiskan respons terhadap peristiwa

Otomatiskan respons terhadap peristiwa untuk mengurangi kesalahan yang disebabkan oleh proses manual, dan untuk memastikan respons yang konsisten dan tepat waktu.

Ada sejumlah cara untuk mengotomatiskan tindakan runbook dan playbook di AWS. Untuk merespons peristiwa dari perubahan keadaan di sumber daya AWS Anda, atau dari peristiwa kustom Anda sendiri, Anda harus membuat aturan CloudWatch Events untuk memicu respons melalui target CloudWatch (contohnya, fungsi Lambda, topik Amazon Simple Notification Service (Amazon SNS), tugas Amazon ECS, dan Otomatisasi AWS Systems Manager).

Untuk merespons metrik yang melampaui ambang batas untuk sumber daya (contohnya, waktu tunggu), Anda harus membuat alarm CloudWatch untuk melakukan satu atau lebih tindakan menggunakan tindakan CloudWatch Events, tindakan Auto Scaling, atau untuk mengirimkan notifikasi ke topik Amazon SNS. Jika Anda harus melakukan tindakan kustom untuk merespons alarm, panggil Lambda melalui notifikasi Amazon SNS. Gunakan Amazon SNS untuk mempublikasikan notifikasi peristiwa dan pesan eskalasi agar orang selalu tahu.

AWS juga mendukung sistem pihak ketiga melalui API dan SDK layanan AWS. Ada sejumlah alat pemantauan yang disediakan oleh Partner AWS dan pihak ketiga yang memungkinkan pemantauan, notifikasi, dan respons. Beberapa alat ini antara lain New Relic, Splunk, Loggly, SumoLogic, dan Datadog.

Anda harus selalu menyediakan prosedur manual yang sangat penting untuk digunakan ketika prosedur otomatis gagal

Antipola umum:

- Developer memeriksa kodenya. Peristiwa ini bisa saja digunakan untuk mulai membangun kemudian melakukan pengujian tetapi tidak ada yang terjadi.
- Aplikasi Anda mencatat kesalahan spesifik sebelum berhenti berfungsi. Prosedur untuk memulai ulang aplikasi dipahami dengan baik dan dapat diberi skrip. Anda dapat menggunakan log event untuk memanggil skrip dan memulai ulang aplikasi. Tetapi, ketika kesalahan terjadi pada hari Minggu jam 3 pagi, Anda dibangunkan karena Anda adalah sumber daya yang siap dipanggil untuk memperbaiki sistem tersebut.

Manfaat menerapkan praktik terbaik ini: Dengan menggunakan respons otomatis terhadap peristiwa, Anda mengurangi waktu untuk merespons dan membatasi timbulnya kesalahan akibat aktivitas manual.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

- Otomatiskan respons terhadap peristiwa: Otomatiskan respons terhadap peristiwa untuk mengurangi kesalahan yang disebabkan oleh proses manual, dan untuk memastikan respons yang konsisten dan tepat waktu.
  - Apa itu Amazon CloudWatch Events?
  - Membuat aturan CloudWatch Events yang memicu peristiwa
  - Membuat aturan CloudWatch Events yang memicu AWS panggilan API menggunakan AWS CloudTrail
  - Contoh peristiwa CloudWatch Events dari layanan yang didukung

Sumber daya

Dokumen terkait:

- Amazon CloudWatch Fitur
- Contoh peristiwa CloudWatch Events dari layanan yang didukung
- Membuat aturan CloudWatch Events yang memicu AWS panggilan API menggunakan AWS CloudTrail
- Membuat aturan CloudWatch Events yang memicu peristiwa
- Apa itu Amazon CloudWatch Events?

Video terkait:

- Buat Rencana Pemantauan

Contoh terkait:

Kembangkan

Perkembangan merupakan siklus berkelanjutan dari peningkatan yang terus-menerus. Implementasikan perubahan kecil secara bertahap berdasarkan pengalaman yang dipelajari dari aktivitas operasional sebelumnya, serta evaluasi peningkatan yang sukses dicapai.

Agar dapat terus mengembangkan operasi, Anda harus:

Topik

Belajar, berdiskusi, dan melakukan perbaikan

Menyisihkan waktu secara rutin untuk melakukan analisis aktivitas operasi, analisis kegagalan, bereksperimen, dan melakukan perbaikan, adalah hal yang penting. Ketika terjadi kegagalan, jadikan kegagalan itu sebagai pelajaran untuk tim Anda dan komunitas rekayasawan secara umum. Anda harus menganalisis kegagalan untuk mengambil pelajaran dan merencanakan perbaikan. Anda perlu meninjau hasil pengamatan Anda bersama tim lain untuk memvalidasi wawasan tersebut.

Praktik terbaik

OPS11-BP01 Miliki proses untuk peningkatan berkelanjutan

Evaluasi beban kerja Anda berdasarkan praktik terbaik arsitektur internal dan eksternal Lakukan peninjauan beban kerja setidaknya satu kali setahun. Prioritaskan peluang perbaikan ke dalam jadwal pengembangan perangkat lunak Anda.

Hasil yang diinginkan:

- Anda menganalisis beban kerja berdasarkan praktik terbaik arsitektur setidaknya satu kali setahun.
- Peluang perbaikan memperoleh prioritas yang setara dalam proses pengembangan perangkat lunak Anda.

Antipola umum:

- Anda belum menjalankan peninjauan arsitektur pada beban kerja Anda sejak deployment beberapa tahun lalu.
- Peluang perbaikan menerima prioritas yang lebih rendah dan tetap berada di backlog.
- Tidak ada standar untuk mengimplementasikan modifikasi terhadap praktik terbaik untuk organisasi.

Manfaat menjalankan praktik terbaik ini:

- Beban kerja Anda selalu dimutakhirkan dengan praktik terbaik arsitektur.
- Mengembangkan beban kerja dilakukan secara cermat.
- Anda dapat memanfaatkan praktik terbaik organisasi untuk meningkatkan semua beban kerja.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Minimal satu kali dalam setahun, Anda melakukan peninjauan arsitektur beban kerja. Menggunakan praktik terbaik internal dan eksternal, evaluasi beban kerja Anda dan identifikasi peluang perbaikan. Prioritaskan peluang perbaikan ke dalam jadwal pengembangan perangkat lunak Anda.

Contoh pelanggan

Semua beban kerja di AnyCompany Retail menjalani proses peninjauan arsitektur tahunan. Mereka mengembangkan daftar periksa praktik terbaik mereka sendiri yang berlaku untuk semua beban

kerja. Menggunakan fitur Lensa Kustom AWS Well-Architected Tool, mereka melakukan peninjauan menggunakan alat dan lensa praktik terbaik kustom mereka. Peluang perbaikan yang dihasilkan dari peninjauan diberikan prioritas dalam sprint perangkat lunak mereka.

Langkah implementasi

1. Lakukan peninjauan arsitektur berkala pada beban kerja produksi Anda setidaknya satu kali dalam setahun. Gunakan standar arsitektur terdokumentasi yang menyertakan praktik terbaik khusus AWS.
   a. Kami menyarankan Anda menggunakan standar yang ditetapkan secara internal untuk peninjauan ini. Jika Anda tidak memiliki standar internal, kami menyarankan Anda menggunakan Kerangka Kerja AWS Well-Architected.
   b. Anda dapat menggunakan AWS Well-Architected Tool untuk membuat Lensa Kustom praktik terbaik internal Anda dan melakukan peninjauan arsitektur Anda.
   c. Pelanggan dapat menghubungi Arsitek Solusi AWS mereka untuk melakukan Peninjauan Kerangka Kerja Well-Architected terpandu pada beban kerja mereka.
2. Prioritaskan peluang perbaikan yang diidentifikasi selama peninjauan ke dalam proses pengembangan perangkat lunak Anda.

Tingkat upaya untuk rencana implementasi: Rendah Anda dapat menggunakan Kerangka Kerja AWS Well-Architected untuk melakukan peninjauan arsitektur tahunan Anda.

Sumber daya

Praktik Terbaik Terkait:

- OPS11-BP02 Menjalankan analisis setelah insiden - Analisis pascainsiden adalah penghasil item perbaikan lainnya. Masukkan pelajaran yang diperoleh ke dalam daftar praktik terbaik arsitektur internal Anda.
- OPS11-BP08 Mendokumentasikan dan membagikan pelajaran yang didapatkan - Bagikan praktik terbaik arsitektur internal yang sedang Anda kembangkan kepada seluruh organisasi Anda.

Dokumen terkait:

- AWS Well-Architected Tool - Lensa kustom
- Laporan Resmi AWS Well-Architected - Proses peninjauan

- Kustomisasi Peninjauan Well-Architected menggunakan Lensa Kustom dan AWS Well-Architected Tool
- Mengimplementasikan siklus hidup Lensa Kustom AWS Well-Architected di dalam organisasi Anda

Video terkait:

- Lab Well-Architected - Level 100: Lensa Kustom di AWS Well-Architected Tool

Contoh terkait:

- AWS Well-Architected Tool

OPS11-BP02 Menjalankan analisis setelah insiden

Tinjau peristiwa yang memengaruhi pelanggan, dan identifikasi faktor yang berkontribusi serta tindakan pencegahannya. Gunakan informasi ini untuk mengembangkan mitigasi guna meminimalkan atau mencegah kemungkinan terjadi lagi. Kembangkan prosedur untuk respons efektif dan cepat. Komunikasikan faktor yang berkontribusi dan tindakan korektif yang diperlukan, yang disesuaikan dengan audiens target.

Antipola umum:

- Anda mengelola server aplikasi. Kira-kira setiap 23 jam 55 menit, semua sesi aktif Anda dihapus. Anda berupaya mengidentifikasi masalah yang terjadi di server aplikasi Anda. Anda menduga bahwa ini mungkin masalah jaringan, tetapi tidak dapat memperoleh bantuan dari tim jaringan karena mereka terlalu sibuk. Anda tidak menetapkan proses di awal yang dapat Anda jadikan panduan untuk mendapatkan dukungan dan mengumpulkan informasi yang dibutuhkan guna mengetahui masalah yang sedang terjadi.
- Anda mengalami kehilangan data di dalam beban kerja Anda. Hal ini baru pertama kali terjadi dan penyebabnya belum jelas. Anda menganggap bahwa kejadian ini tidak penting karena Anda dapat membuat ulang data. Kehilangan data makin sering terjadi dan memengaruhi pelanggan Anda. Hal ini juga menambah beban operasional Anda karena harus memulihkan data yang hilang.

Manfaat menerapkan praktik terbaik ini: Dengan proses yang telah ditetapkan di awal untuk menentukan komponen, kondisi, tindakan, dan kejadian yang berkontribusi terhadap insiden, Anda dapat mengidentifikasi peluang untuk pengembangan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Tinggi

Panduan implementasi

- Gunakan sebuah proses untuk mengetahui faktor yang berkontribusi: Tinjau semua insiden yang memengaruhi pelanggan. Buat sebuah proses untuk mengidentifikasi dan mendokumentasi faktor yang berkontribusi terhadap insiden agar Anda dapat mengembangkan mitigasi untuk membatasi atau mencegah kejadian serupa serta mengembangkan prosedur untuk merespons dengan cepat dan efektif. Komunikasikan penyebab utama sebagaimana diperlukan, yang disesuaikan dengan audiens target.

OPS11-BP03 Mengimplementasikan loop umpan balik

Loop umpan balik menyediakan wawasan yang dapat ditindaklanjuti yang mendorong pengambilan keputusan. Masukkan loop umpan balik ke dalam prosedur dan beban kerja Anda. Ini membantu Anda mengidentifikasi permasalahan dan area yang memerlukan perbaikan. Loop umpan balik juga memvalidasi investasi yang dilakukan dalam upaya perbaikan. Loop umpan balik ini adalah landasan untuk meningkatkan beban kerja Anda secara berkelanjutan.

Loop umpan balik dibagi ke dalam dua kategori: umpan balik langsung dan analisis retrospektif. Umpan balik langsung (immediate feedback) dikumpulkan melalui peninjauan kinerja dan hasil dari aktivitas operasi. Umpan balik ini berasal dari anggota tim, pelanggan, atau output otomatis dari aktivitas. Umpan balik langsung diterima dari hal-hal seperti pengujian A/B dan pengiriman fitur baru, dan ini penting untuk gagal cepat (fail fast).

Analisis retrospektif dilakukan secara rutin untuk menangkap umpan balik dari peninjauan metrik dan hasil operasional dari waktu ke waktu. Retrospektif ini terjadi pada akhir sprint, secara terjadwal, atau setelah perilisan atau peristiwa besar. Tipe loop umpan balik ini memvalidasi investasi dalam operasi atau beban kerja Anda. Loop umpan balik ini membantu Anda mengukur keberhasilan dan memvalidasi strategi Anda.

Hasil yang diinginkan: Anda menggunakan umpan balik langsung dan analisis retrospektif untuk mendorong perbaikan. Terdapat mekanisme untuk mendapatkan umpan balik pengguna dan anggota tim. Analisis retrospektif digunakan untuk mengidentifikasi tren-tren yang mendorong perbaikan.

Antipola umum:

- Anda meluncurkan fitur baru tetapi tidak ada cara untuk menerima umpan balik pelanggan tentangnya.

- Setelah berinvestasi dalam perbaikan operasi, Anda tidak melakukan analisis retrospektif untuk memvalidasinya.
- Anda mengumpulkan umpan balik pelanggan tetapi tidak meninjaunya secara rutin.
- Loop umpan balik mendatangkan item-item tindakan yang diajukan tetapi item-item tersebut tidak disertakan dalam proses pengembangan perangkat lunak.
- Pelanggan tidak menerima umpan balik tentang perbaikan yang mereka ajukan.

Manfaat menjalankan praktik terbaik ini:

- Anda dapat bekerja mundur dari pelanggan untuk mendorong fitur-fitur baru.
- Budaya organisasi Anda dapat merespons perubahan lebih cepat.
- Tren digunakan untuk mengidentifikasi peluang perbaikan.
- Retrospektif memvalidasi investasi yang dilakukan pada beban kerja dan operasi Anda.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Dengan mengimplementasikan praktik terbaik ini, Anda dapat menggunakan umpan balik langsung serta analisis retrospektif. Loop umpan balik ini mendorong perbaikan. Terdapat banyak mekanisme untuk umpan balik langsung, termasuk survei, jajak pendapat pelanggan, atau formulir umpan balik. Organisasi Anda juga menggunakan retrospektif untuk mengidentifikasi peluang perbaikan dan memvalidasi inisiatif.

Contoh pelanggan

AnyCompany Retail membuat sebuah formulir web yang digunakan pelanggan untuk memberikan umpan balik atau melaporkan permasalahan. Selama scrum mingguan, umpan balik pengguna dievaluasi oleh tim pengembangan perangkat lunak. Umpan balik digunakan secara rutin sebagai landasan pengembangan platform mereka. Mereka melakukan analisis retrospektif di akhir setiap sprint untuk mengidentifikasi item yang ingin mereka tingkatkan.

Langkah implementasi

1. Umpan balik langsung
   - Anda memerlukan mekanisme untuk menjangkau umpan balik dari pelanggan dan anggota tim. Aktivitas operasi Anda juga dapat dikonfigurasi untuk menghadirkan umpan balik otomatis.

- Organisasi Anda perlu meninjau umpan balik ini, menentukan hal-hal yang harus ditingkatkan, dan menjadwalkan perbaikan.
- Umpan balik harus ditambahkan ke dalam proses pengembangan perangkat lunak Anda.
- Seiring Anda melakukan perbaikan, sampaikan tindak lanjut kepada pemberi umpan balik.
  - Anda dapat menggunakan AWS Systems Manager OpsCenter untuk membuat dan melacak perbaikan ini dalam bentuk OpsItems.

2. Analisis retrospektif
   - Lakukan retrospektif di akhir siklus pengembangan, pada jadwal yang ditetapkan, atau setelah perilisan besar.
   - Kumpulkan pemangku kepentingan yang terlibat dalam beban kerja untuk rapat retrospektif.
   - Buat tiga kolom di papan tulis atau lembar kerja: Hentikan, Mulai, dan Pertahankan
     - Hentikan adalah untuk apa pun yang Anda ingin tidak dilakukan lagi oleh tim Anda.
     - Mulai adalah gagasan yang ingin mulai Anda lakukan.
     - Pertahankan adalah untuk item-item yang ingin tetap Anda lakukan.
   - Berkelilinglah dan kumpulkan umpan balik dari para pemangku kepentingan.
   - Buat prioritas umpan balik. Tugaskan tindakan dan pemangku kepentingan ke item-item Mulai atau Pertahankan.
   - Tambahkan tindakan ke proses pengembangan perangkat lunak dan komunikasikan pembaruan status ke pemangku kepentingan seiring Anda melakukan perbaikan.

Tingkat upaya untuk rencana implementasi: Sedang. Untuk mengimplementasikan praktik terbaik, Anda memerlukan cara untuk menyerap umpan balik langsung dan menganalisisnya. Selain itu, Anda perlu membangun proses analisis retrospektif.

Sumber daya

Praktik terbaik terkait:

- OPS01-BP01 Mengevaluasi kebutuhan pelanggan eksternal: Loop umpan balik adalah mekanisme untuk mengumpulkan kebutuhan pelanggan eksternal.
- OPS01-BP02 Mengevaluasi kebutuhan pelanggan internal: Pemangku kepentingan internal dapat menggunakan loop umpan balik untuk mengomunikasikan kebutuhan dan persyaratan.
- OPS11-BP02 Menjalankan analisis setelah insiden: Analisis pascainsiden adalah bentuk analisis retrospektif yang penting yang dilakukan setelah insiden.

- OPS11-BP07 Melakukan peninjauan metrik operasi: Peninjauan metrik operasi mengidentifikasi tren dan area perbaikan.

Dokumen terkait:

- 7 Perangkap yang Perlu Dihindari Saat Membangun CCOE
- Playbook Tim Atlassian - Retrospektif
- Definisi Email: Loop Umpan Balik
- Membangun Loop Umpan Balik Berdasarkan Tinjauan AWS Well-Architected Framework
- Metodologi Garasi IBM - Melakukan retrospektif
- Investopedia - Siklus PDCS
- Memaksimalkan Efektivitas Developer oleh Tim Cochran
- Laporan Resmi Peninjauan Kesiapan Operasional (ORR) - Iterasi
- TIL CSI - Continual Service Improvement (Perbaikan Layanan Berkelanjutan)
- Saat Toyota bertemu e-commerce: Bersandar pada Amazon

Video terkait:

- Membangun Loop Umpan Balik Pelanggan yang Efektif

Contoh terkait:

- Astuto - alat umpan balik pelanggan sumber terbuka
- Solusi AWS - QnABot di AWS
- Fider - Platform untuk mengatur umpan balik pelanggan

Layanan terkait:

- AWS Systems Manager OpsCenter

OPS11-BP04 Menjalankan manajemen pengetahuan

Manajemen pengetahuan membantu anggota tim menemukan informasi untuk melakukan pekerjaan mereka. Di dalam organisasi yang mau belajar, informasi dibagikan secara bebas sehingga individu

diberdayakan. Informasi dapat ditemukan atau dicari. Informasi bersifat akurat dan mutakhir. Ada mekanisme untuk membuat informasi baru, memperbarui informasi yang sudah ada, dan mengarsipkan informasi yang kedaluwarsa. Contoh paling umum dari platform manajemen pengetahuan adalah sistem manajemen konten seperti wiki.

Hasil yang diinginkan:

- Anggota tim memiliki akses ke informasi yang akurat secara tepat waktu.
- Informasi dapat dicari.
- Ada mekanisme untuk menambahkan, memperbarui, dan mengarsipkan informasi.

Antipola umum:

- Tidak ada penyimpanan pengetahuan terpusat. Anggota tim mengelola catatan mereka sendiri di mesin mereka secara lokal.
- Anda memiliki wiki yang di-hosting secara mandiri tetapi tidak ada mekanisme untuk mengelola informasi, yang mengakibatkan informasi menjadi kedaluwarsa.
- Seseorang melihat ada informasi yang kurang tetapi tidak ada proses untuk meminta penambahannya ke wiki. Mereka menambahkannya sendiri tetapi mereka melewatkan langkah yang penting, sehingga mengakibatkan terjadinya gangguan.

Manfaat menjalankan praktik terbaik ini:

- Anggota tim diberdayakan karena informasi dibagikan secara bebas.
- Anggota tim baru menjalani masa orientasi dengan lebih cepat karena dokumentasinya mutakhir dan dapat dicari.
- Informasi bersifat tepat waktu, akurat, dan dapat ditindaklanjuti.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Tinggi

Panduan implementasi

Manajemen pengetahuan adalah segi penting dari organisasi yang mau belajar. Untuk memulai, Anda memerlukan tempat penyimpanan terpusat guna menyimpan pengetahuan Anda (contoh yang umum yakni wiki yang di-hosting secara mandiri). Anda harus membuat proses untuk menambahkan, memperbarui, dan mengarsipkan pengetahuan. Buat standar untuk apa yang harus didokumentasikan dan izinkan semua orang memberi kontribusi.

Contoh pelanggan

AnyCompany Retail melakukan hosting Wiki internal tempat semua pengetahuan disimpan. Anggota tim didorong untuk menambahkan pengetahuan seiring pengerjaan tugas mereka sehari-hari. Setiap kuartal sekali, tim lintas fungsi mengevaluasi halaman mana yang paling jarang diperbarui dan menentukan apakah halaman tersebut harus diarsipkan atau diperbarui.

Langkah implementasi

1. Mulai dengan identifikasi sistem manajemen konten tempat pengetahuan akan disimpan. Dapatkan kesepakatan para pemangku kepentingan dari seluruh organisasi Anda.
   a. Jika Anda belum memiliki sistem manajemen konten, pertimbangkan untuk menjalankan wiki yang di-hosting secara mandiri atau menggunakan tempat penyimpanan pengontrolan versi sebagai titik awal.
2. Kembangkan runbook untuk menambahkan, memperbarui, dan mengarsipkan informasi. Didik tim Anda tentang proses-proses ini.
3. Identifikasi pengetahuan apa yang harus disimpan di sistem manajemen konten. Mulai dengan aktivitas harian (runbook dan playbook) yang dilakukan anggota tim. Bekerja samalah dengan para pemangku kepentingan untuk memprioritaskan pengetahuan yang akan ditambahkan.
4. Bekerja samalah dengan para pemangku kepentingan secara berkala untuk mengidentifikasi informasi yang kedaluwarsa dan arsipkan atau mutakhirkan.

Tingkat upaya untuk rencana implementasi: Sedang. Jika Anda belum memiliki sistem manajemen konten, Anda dapat membuat wiki yang di-hosting secara mandiri atau menggunakan tempat penyimpanan dokumen dengan pengontrolan versi.

Sumber daya

Praktik terbaik terkait:

- OPS11-BP08 Mendokumentasikan dan membagikan pelajaran yang didapatkan - Manajemen pengetahuan memfasilitasi pembagian informasi tentang pelajaran yang didapatkan.

Dokumen terkait:

- Atlassian - Manajemen Pengetahuan

Contoh terkait:

- DokuWiki
- Gollum
- MediaWiki
- Wiki.js

OPS11-BP05 Menetapkan pendorong untuk perbaikan

Identifikasi pendorong untuk perbaikan untuk membantu Anda mengevaluasi dan memprioritaskan peluang.

Di AWS, Anda dapat mengagregasi log semua aktivitas operasi, beban kerja, dan infrastruktur Anda untuk membuat riwayat aktivitas yang mendetail. Kemudian, Anda dapat menggunakan alat-alat AWS untuk menganalisis operasi dan kesehatan beban kerja Anda seiring waktu (misalnya untuk mengidentifikasi tren, mengaitkan peristiwa dan aktivitas dengan hasil, dan membandingkan serta mengkontraskan antarlingkungan dan lintas sistem) untuk mengungkap peluang perbaikan berdasarkan pendorong Anda.

Anda harus menggunakan CloudTrail untuk melacak aktivitas API (melalui AWS Management Console, CLI, SDK, dan API) untuk mengetahui apa yang terjadi di seluruh akun Anda. Lacak aktivitas deployment Alat pengembang AWS Anda dengan CloudTrail dan CloudWatch. Ini akan menambahkan riwayat aktivitas mendetail untuk deployment Anda serta hasilnya ke data log CloudWatch Logs Anda.

Ekspor data log Anda ke Amazon S3 untuk penyimpanan jangka panjang. Menggunakan AWS Glue, Anda menemukan dan mempersiapkan data log Anda di Amazon S3 untuk analitik. Gunakan Amazon Athena, melalui integrasi native-nya dengan AWS Glue, untuk menganalisis data log Anda. Gunakan alat kecerdasan bisnis seperti Amazon QuickSight untuk memvisualisasi, menjelajahi, dan menganalisis data Anda

Antipola umum:

- Anda memiliki skrip yang berfungsi tetapi tidak elegan. Anda menginvestasikan waktu untuk menulis ulang skrip tersebut. Kini skrip tersebut terlihat sangat bagus.
- Perusahaan rintisan Anda sedang mencoba mendapatkan pendanaan lain dari sebuah pemodal ventura. Mereka meminta Anda mendemonstrasikan kepatuhan terhadap PCI DSS. Anda ingin membuat mereka terkesan sehingga Anda mendokumentasikan kepatuhan, tetapi Anda

melewatkan tanggal pengiriman untuk seorang pelanggan dan kehilangan pelanggan tersebut. Ini bukan tindakan yang salah tetapi sekarang Anda bertanya-tanya apakah itu tindakan yang tepat.

Manfaat menjalankan praktik terbaik ini: Dengan menentukan kriteria yang ingin Anda gunakan untuk perbaikan, Anda dapat meminimalkan dampak motivasi berbasis peristiwa atau investasi emosional.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Sedang

Panduan implementasi

- Pahami pendorong perbaikan: Anda sebaiknya hanya melakukan perubahan pada suatu sistem saat hasil yang diinginkan didukung.
  - Kemampuan yang diinginkan: Evaluasi fitur dan kemampuan yang diinginkan saat mengevaluasi peluang untuk perbaikan.
    - Yang Baru dengan AWS
  - Masalah yang tidak dapat diterima: Evaluasi masalah, bug, dan kerentanan yang tidak dapat diterima saat mengevaluasi peluang untuk perbaikan.
    - Buletin Keamanan Terkini AWS
    - AWS Trusted Advisor
  - Persyaratan kepatuhan: Evaluasi pembaruan dan perubahan yang diperlukan untuk mempertahankan kepatuhan terhadap peraturan, kebijakan, atau agar tetap memperoleh dukungan pihak ketiga, saat meninjau peluang untuk perbaikan.
    - Kepatuhan AWS
    - Program Kepatuhan AWS
    - Berita Terbaru Kepatuhan AWS

Sumber daya

Dokumen terkait:

- Amazon Athena
- Amazon QuickSight
- Kepatuhan AWS
- Berita Terbaru Kepatuhan AWS
- Program Kepatuhan AWS

- AWS Glue
- Buletin Keamanan Terkini AWS
- AWS Trusted Advisor
- Ekspor data log Anda ke Amazon S3
- Yang Baru dengan AWS

OPS11-BP06 Memvalidasi wawasan

Tinjau respons dan hasil analisis Anda dengan tim lintas fungsi serta pemilik bisnis. Gunakan tinjauan tersebut untuk menetapkan pemahaman umum, mengidentifikasi dampak tambahan, dan menentukan alur tindakan. Sesuaikan respons sebagaimana mestinya.

Antipola umum:

- Anda menemukan pemanfaatan CPU pada sistem sebesar 95% dan menjadikan hal itu sebagai prioritas untuk menemukan cara mengurangi beban pada sistem. Anda menentukan tindakan terbaik yang perlu dinaikkan skalanya. Sistemnya adalah transkoder dan sistem tersebut diskalakan untuk menjalankan 95% pemanfaatan CPU sepanjang waktu. Pemilik sistem dapat menjelaskan situasinya kepada Anda jika Anda menghubunginya. Waktu Anda terbuang.
- Pemilik sistem menyatakan bahwa sistem mereka bersifat kritis terhadap misi. Sistemnya tidak ditempatkan di lingkungan dengan keamanan tinggi. Untuk meningkatkan keamanan, Anda mengimplementasikan kontrol detektif dan preventif yang diperlukan untuk sistem yang kritis terhadap misi. Anda memberi tahu pemilik sistem bahwa pekerjaannya sudah selesai dan dia akan dikenakan biaya untuk sumber daya tambahan. Dalam diskusi setelah pemberitahuan ini, pemilik sistem mengetahui bahwa ada ketentuan formal untuk sistem yang kritis terhadap misi yang tidak dipenuhi oleh sistem ini.

Manfaat menerapkan praktik terbaik ini: Dengan memvalidasi wawasan bersama pemilik bisnis dan orang yang ahli di bidangnya, Anda dapat menetapkan pemahaman umum dan memandu peningkatan dengan lebih efektif.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Sedang

Panduan implementasi

- Validasikan wawasan: Berinteraksi dengan pemilik bisnis dan orang yang ahli di bidangnya untuk memastikan ada pemahaman dan kesepakatan bersama tentang makna data yang dikumpulkan. Identifikasikan masalah tambahan, dampak potensial, dan tentukan alur tindakan.

OPS11-BP07 Melakukan peninjauan metrik operasi

Lakukan analisis retrospektif rutin terhadap metrik operasi dengan peserta lintas tim dari berbagai area bisnis. Gunakan tinjauan ini untuk mengidentifikasi peluang perbaikan, potensi pilihan tindakan, dan untuk membagikan pelajaran yang diperoleh.

Cari peluang perbaikan di semua lingkungan Anda (misalnya pengembangan, pengujian, dan produksi).

Antipola umum:

- Terdapat promosi ritel penting yang terganggu oleh jadwal pemeliharaan Anda. Bisnis tidak tahu bahwa ada jadwal pemeliharaan standar yang dapat ditunda jika terdapat peristiwa lain yang memengaruhi bisnis.
- Anda mengalami pemadaman berkepanjangan karena menggunakan pustaka bermasalah yang biasa digunakan di organisasi Anda. Sejak saat itu Anda beralih ke pustaka yang andal. Tim-tim lain di organisasi Anda tidak tahu bahwa mereka terpapar risiko. Jika Anda rutin melakukan pertemuan dan meninjau insiden ini, mereka akan mengetahui risiko tersebut.
- Kinerja transkoder Anda terus mengalami penurunan secara bertahap dan memengaruhi tim media. Saat ini kondisinya belum parah. Anda tidak akan memiliki kesempatan untuk tahu sampai kondisinya cukup buruk hingga menyebabkan insiden. Seandainya Anda meninjau metrik operasi dengan tim media, akan ada peluang untuk melakukan perubahan pada metrik, mengenali pengalaman mereka, dan mengatasi masalah.
- Anda tidak meninjau kepuasan Anda terhadap SLA pelanggan. Anda memiliki kecenderungan untuk tidak memenuhi SLA pelanggan. Terdapat denda finansial jika Anda tidak memenuhi SLA pelanggan. Jika rutin melakukan pertemuan untuk meninjau metrik untuk SLA ini, Anda akan memiliki kesempatan untuk mengenali dan menangani masalah.

Manfaat menjalankan praktik terbaik ini: Dengan melakukan pertemuan rutin untuk meninjau metrik operasi, peristiwa, dan insiden, Anda dapat menjaga pemahaman bersama lintas tim, membagikan pelajaran yang didapatkan, dan dapat memprioritaskan serta menargetkan perbaikan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak ditetapkan: Sedang

Panduan implementasi

- Tinjauan metrik operasi: Lakukan analisis retrospektif rutin terhadap metrik operasi dengan peserta lintas tim dari berbagai area bisnis. Libatkan pemangku kepentingan, termasuk tim bisnis, pengembangan, dan operasi, untuk memvalidasi temuan dari umpan balik langsung dan analisis retrospektif, serta untuk membagikan pelajaran yang didapatkan. Gunakan wawasan mereka untuk mengidentifikasi peluang perbaikan dan potensi pilihan tindakan.
  - Amazon CloudWatch
  - Menggunakan metrik Amazon CloudWatch
  - Memublikasikan metrik kustom
  - Metrik Amazon CloudWatch dan referensi dimensi

Sumber daya

Dokumen terkait:

- Amazon CloudWatch
- Metrik Amazon CloudWatch dan referensi dimensi
- Memublikasikan metrik kustom
- Menggunakan metrik Amazon CloudWatch

OPS11-BP08 Mendokumentasikan dan membagikan pelajaran yang didapatkan

Dokumentasikan dan bagikan pelajaran yang didapatkan dari aktivitas operasional sehingga Anda dapat menggunakannya secara internal dan di seluruh tim.

Anda harus membagikan pelajaran yang didapatkan oleh tim Anda guna meningkatkan manfaat di seluruh organisasi Anda. Anda perlu membagikan informasi dan sumber daya untuk mencegah kesalahan yang dapat dihindari dan memudahkan upaya pengembangan. Dengan demikian, Anda dapat fokus menghadirkan fitur-fitur yang diinginkan.

Gunakan AWS Identity and Access Management (IAM) untuk menetapkan izin yang memungkinkan akses terkontrol ke sumber daya yang ingin Anda bagikan di dalam dan antarakun. Anda harus

menggunakan repositori AWS CodeCommit terkontrol versi untuk membagikan pustaka aplikasi, prosedur dalam skrip, dokumentasi prosedur, dan dokumentasi sistem lainnya. Bagikan standar komputasi Anda dengan membagikan akses ke AMI Anda dan dengan memberikan otorisasi penggunaan fungsi Lambda Anda di seluruh akun. Anda juga harus membagikan standar infrastruktur Anda dalam bentuk templat AWS CloudFormation.

Melalui API dan SDK AWS, Anda dapat mengintegrasikan alat dan repositori eksternal dan pihak ketiga (seperti GitHub, BitBucket, dan SourceForge). Ketika membagikan hal-hal yang Anda pelajari dan kembangkan, berhati-hatilah untuk menyusun izin guna memastikan integritas repositori yang dibagikan.

Antipola umum:

- Anda mengalami pemadaman berkepanjangan karena Anda menggunakan pustaka bermasalah yang biasa digunakan di organisasi Anda. Sejak saat itu Anda beralih ke pustaka yang andal. Tim-tim lain di organisasi Anda tidak tahu bahwa mereka terpapar risiko. Andai saja Anda mendokumentasikan dan membagikan pengalaman Anda dengan pustaka ini, mereka pasti tahu tentang risiko tersebut.
- Anda mengidentifikasi sebuah masalah di dalam layanan mikro yang digunakan bersama secara internal yang menyebabkan terganggunya sesi. Anda pun memperbarui panggilan Anda ke layanan guna menghindari masalah tersebut. Tim-tim lain di organisasi Anda tidak tahu bahwa mereka terpapar risiko. Andai saja Anda mendokumentasikan dan membagikan pengalaman Anda dengan pustaka ini, mereka pasti tahu tentang risiko tersebut.
- Anda menemukan cara untuk mengurangi secara signifikan persyaratan pemanfaatan CPU untuk salah satu layanan mikro Anda. Anda tidak tahu bahwa tim lain bisa memanfaatkan teknik ini. Andai saja Anda mendokumentasikan dan membagikan pengalaman Anda dengan pustaka ini, mereka pasti memiliki peluang untuk melakukannya.

Manfaat menjalankan praktik terbaik ini: Bagikan pelajaran yang didapatkan untuk mendukung perbaikan dan memaksimalkan manfaat pengalaman.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Rendah

Panduan implementasi

- Dokumentasikan dan bagikan pelajaran yang didapatkan: Miliki prosedur untuk mendokumentasikan pelajaran didapatkan dari aktivitas operasional dan analisis retrospektif agar dapat digunakan oleh tim lain.

- Bagikan pembelajaran: Miliki prosedur untuk membagikan pelajaran yang didapatkan serta artefak terkait ke seluruh tim. Sebagai contoh, bagikan prosedur, panduan, tata kelola, dan praktik terbaik yang telah diperbarui melalui wiki yang dapat diakses. Bagikan skrip, kode, dan pustaka melalui repositori umum.
  - Mendelegasikan akses ke lingkungan AWS Anda
  - Bagikan repositori AWS CodeCommit
  - Otorisasi fungsi AWS Lambda secara mudah
  - Membagikan AMI kepada Akun AWS tertentu
  - Percepat pembagian tempat dengan URL desainer AWS CloudFormation
  - Menggunakan AWS Lambda dengan Amazon SNS

Sumber daya

Dokumen terkait:

- Otorisasi fungsi AWS Lambda secara mudah
- Bagikan repositori AWS CodeCommit
- Membagikan AMI kepada Akun AWS tertentu
- Percepat pembagian tempat dengan URL desainer AWS CloudFormation
- Menggunakan AWS Lambda dengan Amazon SNS

Video terkait:

- Mendelegasikan akses ke lingkungan AWS Anda

OPS11-BP09 Mengalokasikan waktu untuk membuat peningkatan

Dedikasikan waktu dan sumber daya dalam proses Anda untuk memungkinkan peningkatan bertahap yang berkelanjutan.

Di AWS, Anda dapat membuat duplikat lingkungan sementara, menurunkan risiko, usaha, serta biaya eksperimen dan pengujian. Lingkungan duplikat ini dapat digunakan untuk menguji kesimpulan dari analisis dan eksperimen Anda, serta mengembangkan dan menguji peningkatan terencana.

Antipola umum:

- Ada masalah kinerja yang diketahui dalam aplikasi Anda. Ini ditambahkan ke backlog di balik setiap implementasi fitur terencana. Jika peringkat fitur terencana yang ditambahkan tetap konstan, masalah kinerja tidak akan pernah tertangani.
- Untuk mendukung peningkatan berkelanjutan yang disetujui, administrator dan developer menggunakan seluruh waktu tambahan mereka untuk memilih dan mengimplementasikan peningkatan. Tidak ada peningkatan yang diselesaikan.

Manfaat menerapkan praktik terbaik ini: Dengan mendedikasikan waktu dan sumber daya dalam proses, Anda memungkinkan peningkatan bertahap yang berkelanjutan.

Tingkat risiko yang terjadi jika praktik terbaik ini tidak diterapkan: Rendah

Panduan implementasi

- Alokasikan waktu untuk membuat peningkatan: Dedikasikan waktu dan sumber daya dalam proses Anda untuk memungkinkan peningkatan bertahap yang berkelanjutan. Implementasikan perubahan guna meningkatkan dan mengevaluasi hasil untuk menentukan keberhasilan. Jika hasilnya tidak memenuhi tujuan, dan peningkatan masih menjadi prioritas, lakukan tindakan alternatif.

Kesimpulan

Keunggulan operasional adalah upaya berkelanjutan dan iteratif.

Raih keberhasilan organisasi Anda dengan adanya tujuan bersama. Pastikan semua orang memahami bagian masing-masing dalam mencapai hasil bisnis serta dampaknya terhadap kemampuan orang lain untuk sukses. Berikan dukungan kepada anggota tim untuk mendukung hasil bisnis Anda.

Setiap kegagalan dan peristiwa operasional harus dipandang sebagai peluang untuk meningkatkan operasi arsitektur Anda. Dengan memahami kebutuhan beban kerja, menetapkan runbook sejak awal untuk aktivitas rutin, memanfaatkan buku pedoman untuk memandu penyelesaian masalah, menggunakan operasi sebagai fitur kode di AWS, dan menjaga kewaspadaan dalam situasi apa pun, operasi Anda dapat lebih siap dan mampu merespons dengan lebih efektif saat terjadi insiden.

Dengan berfokus pada peningkatan bertahap berdasarkan prioritas yang berubah-ubah, serta belajar dari respons peristiwa dan analisis retrospektif, Anda dapat menyukseskan bisnis dengan meningkatkan efisiensi dan efektivitas aktivitas Anda.

AWS berupaya untuk membantu Anda membangun dan mengoperasikan arsitektur yang memaksimalkan efisiensi saat Anda membangun deployment yang sangat adaptif dan responsif. Untuk meningkatkan keunggulan operasional beban kerja, Anda harus menerapkan praktik terbaik yang dipaparkan dalam tulisan ini.

Kontributor

- Rich Boyd, Pimpinan Pilar Keunggulan Operasional (Operational Excellence Pillar Lead), Well-Architected, Amazon Web Services
- Jon Steele, Arsitek Solusi (Solutions Architect) Well-Architected, Amazon Web Services
- Ryan King, Manajer Program Teknis Senior (Sr. Technical Program Manager), Amazon Web Services
- Chris Kunselman, Konsultan Penasihat (Advisory Consultant), Amazon Web Services
- Peter Mullen, Konsultan Penasihat (Advisory Consultant), Amazon Web Services
- Brian Quinn, Advisory Consultant, Amazon Web Services
- David Stanley, Cloud Operating Model Lead, Amazon Web Services
- Chris Kozlowski, Senior Specialist Technical Account Manager, Dukungan Korporasi, Amazon Web Services
- Alex Livingstone, Principal Specialist Solutions Architect, Operasi Cloud, Amazon Web Services
- Paul Moran, Principal Technologist, Dukungan Korporasi, Amazon Web Services
- Peter Mullen, Advisory Consultant, Layanan Profesional, Amazon Web Services
- Chris Pate, Senior Specialist Technical Account Manager, Dukungan Korporasi, Amazon Web Services
- Arvind Raghunathan, Principal Specialist Technical Account Manager, Dukungan Korporasi, Amazon Web Services
- Ben Mergen, Senior Cost Lead Solutions Architect, Amazon Web Services

Bacaan lebih lanjut

Untuk panduan tambahan, pelajari sumber berikut:

- Kerangka Kerja AWS Well-Architected
- Pusat Arsitektur AWS

Revisi Dokumen

Berlangganan umpan RSS untuk memperoleh pemberitahuan tentang pembaruan laporan resmi ini.

Perubahan	Deskripsi	Tanggal
Pembaruan dan konsolidasi konten utama	Konten telah diperbarui dan dikonsolidasikan di beberapa area praktik terbaik. Dua area praktik terbaik (OPS 04 dan OPS 08) telah ditulis ulang dengan konten dan fokus baru. Praktik terbaik telah diperbarui dan dikonsolidasikan di bidang-bidang berikut: Desain operasi, Memitigasi risiko deployment, dan Memahami kesehatan operasional. Area praktik terbaik OPS 04 telah diperbarui ke Menerapkan observabilitas. Area praktik terbaik OPS 08 telah diperbarui ke Memanfaatkan observabilitas beban kerja.	October 3, 2023
Pembaruan untuk Kerangka Kerja baru	Praktik terbaik diperbarui dengan panduan preskript if dan praktik terbaik baru ditambahkan.	April 10, 2023
Laporan resmi diperbarui	Praktik terbaik diperbarui dengan panduan implementasi baru.	December 15, 2022

Laporan resmi diperbarui	Praktik terbaik diperluas dan rencana pengembangan ditambahkan.	October 20, 2022
Pembaruan kecil	Pembaruan redaksi kecil.	August 8, 2022
Laporan resmi diperbarui	Pembaruan untuk menggambarkan fitur dan layanan AWS baru, dan praktik terbaik terbaru.	February 2, 2022
Pembaruan kecil	Penambahan Pilar Pelestarian Lingkungan ke pengantar.	December 2, 2021
Pembaruan untuk Kerangka Kerja baru	Pembaruan untuk menggambarkan fitur dan layanan AWS baru, dan praktik terbaik terbaru.	July 8, 2020
Laporan resmi diperbarui	Pembaruan untuk menggambarkan fitur dan layanan AWS baru, dan pembaruan referensi.	July 1, 2018
Publikasi awal	Pilar Keunggulan Operasional - AWS Well-Architected Framework dipublikasikan	November 1, 2017