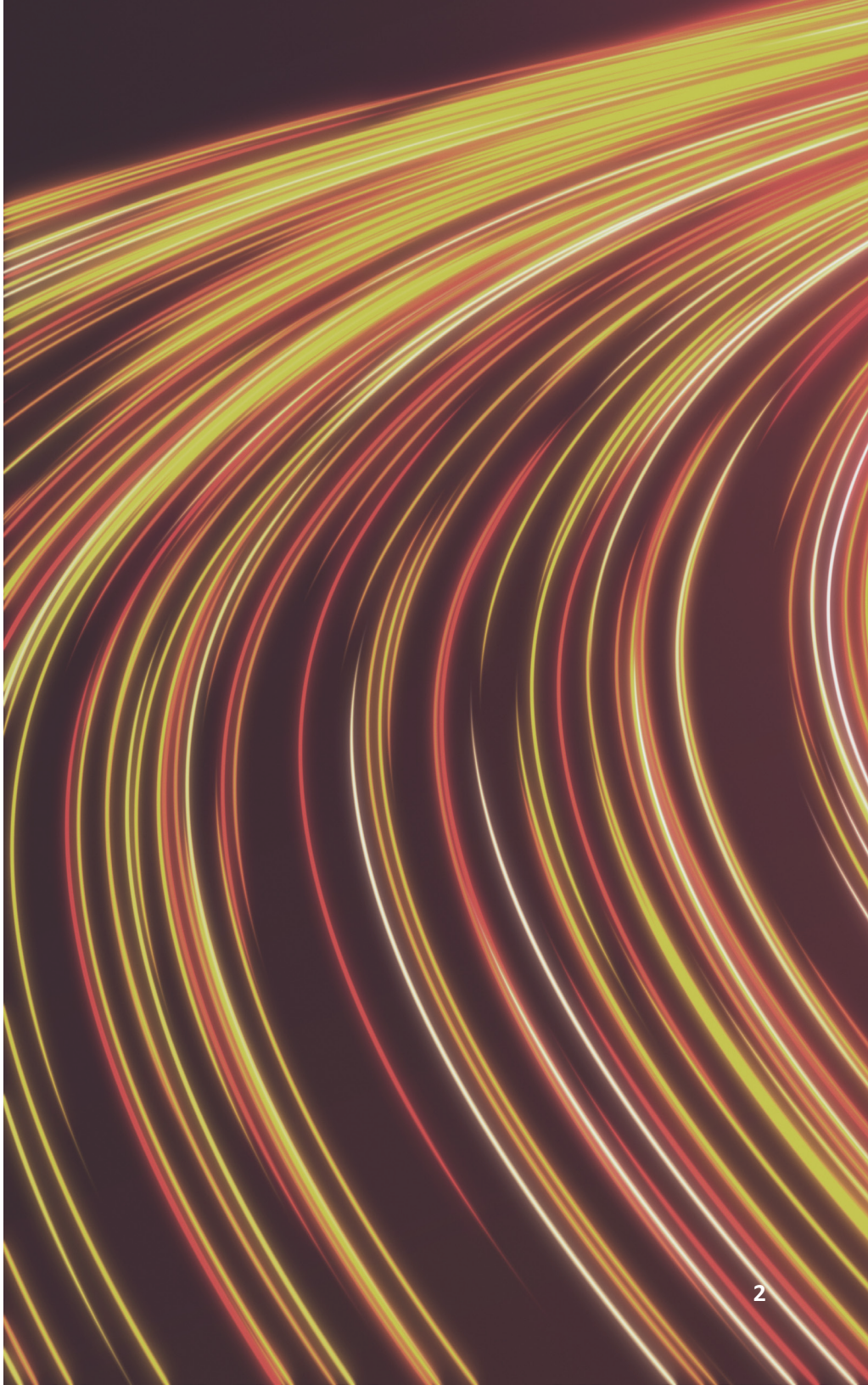

aws modern apps
Mengadopsi model
Dev+Ops Modern

Dev+Ops modern memungkinkan organisasi memenuhi kebutuhan yang terus berkembang dari pelanggan, secara lebih cepat dan lebih konsisten, dengan menutup kesenjangan antara fungsi pengembangan dan operasional.

DevOps telah mengubah dunia. Amazon dan internet mengubah cara perangkat lunak dikirimkan, yang tadinya berupa disk dalam boks yang dibeli di toko, kini tersedia secara digital sebagai layanan yang dibangun di AWS. Namun DevOps-lah yang mengubah siklus pembaruan perangkat lunak tersebut dari hitungan bulanan atau tahunan menjadi hitungan hari, serta mempererat hubungan antara *Dev+Ops*.

Perusahaan dari semua skala dan bentuk telah menyingkirkan siklus manajemen produk model ***waterfall*** dan lebih memilih ketangkasan dan DevOps demi inovasi yang lebih cepat, keamanan, performa, dan ketahanan yang lebih baik, serta developer dan pelanggan yang lebih puas. Meski begitu, DevOps dengan semua kelebihannya tidaklah sempurna. DevOps sudah hadir sebelum pengembangan dan ***hosting*** aplikasi di ***cloud*** menjadi praktik standar, dan meski fungsi ini sudah ada sejak lama, penafsirannya bisa berbeda-beda antara satu organisasi dan organisasi lainnya. Misalnya, banyak organisasi yang masih menganggap DevOps sebagai tim khusus, dan dalam kasus lainnya, fungsi ini ditafsirkan sebagai developer yang melakukan semua pekerjaan operasional. Seiring berkembangnya teknologi, begitu pun



halnya dengan definisi kita atas DevOps. Kami menyebut pendekatan ini sebagai ***Dev+Ops*** Modern, dan pendekatan ini berpusat pada upaya mendekatkan developer dan operasi yang berbagi tugas operasional seperti kepatuhan, observabilitas, ketahanan, dan infrastruktur secara lebih awal ke dalam proses pengembangan dan menyempurnakannya dengan AI/ML.

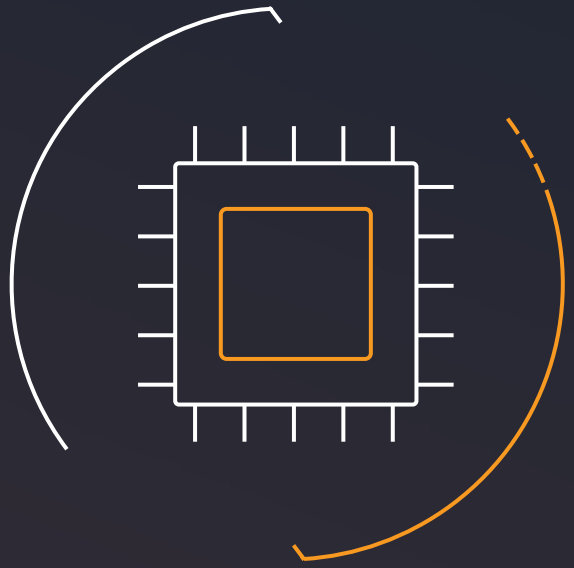
Laporan State of DevOps menyatakan bahwa perusahaan yang mengadopsi DevOps dan mengubah frekuensi ***deployment*** dari mingguan/bulanan menjadi setiap jam/hari mendapati waktu tunggu yang lebih baik, dari hitungan bulan menjadi hari, serta tingkat kegagalan yang menurun, dari 46–60 persen menjadi 0–15 persen. Bagi banyak pelanggan, tolok ukur keberhasilan di ***cloud*** tidak hanya adopsi infrastruktur yang sesuai permintaan dan dapat diskalakan, tetapi juga setidaknya bagaimana mereka mentransformasikan praktik pengembangan dan operasinya. Kini, pelanggan seperti Coca-Cola Argentina, 3M, Lululemon Athletica, dan The Washington Post telah bertransisi ke pendekatan ***Dev+Ops*** Modern, yang didukung oleh layanan ***Dev+Ops*** AWS untuk makin mendekatkan tim pengembangan dan tim operasional mereka agar bisa mengoptimalkan manfaat DevOps.

"Menggunakan AWS, deployment yang tadinya memerlukan enam pekan kini menjadi satu per pekan, dan dalam waktu dekat, kami bisa melakukan lebih dari satu deployment per hari."[1]

—Rick Austin, Manajer (*Manager*) Advanced Technology, 3M Health Information Systems

"Berkat templat AWS CloudFormation dan AWS CodePipeline, kami bisa membangun akun produksi baru dalam hitungan menit saja, bukan dua hari. Itu artinya kami bisa meluncurkan proyek berskala kecil yang bisa disiapkan secara hemat biaya sekaligus hemat waktu. Dengan ketangkasan tersebut, kami bisa bereksperimen dan mendapatkan solusi terbaik alih-alih mandek di sumber daya yang kami miliki saja. Menggunakan AWS, kami dapat mendorong berbagai aplikasi dan fitur baru lebih cepat dari sebelumnya."[2]

—Sam Keen, Direktur Arsitektur Produk (*Director of Product Architecture*), Lululemon Athletica



[1] *https://aws.amazon.com/solutions/case-studies/3M-health-information-systems/*

[2] *https://aws.amazon.com/solutions/case-studies/lululemon-athletica/*

Tim dengan praktik DevOps yang lebih berkembang

77%

mengembalikan layanan setelah insiden dalam waktu kurang dari satu hari

60%

sepenuhnya memulihkan kerentanan keamanan dalam waktu kurang dari satu hari

3x

lebih efektif dalam manajemen perubahan

aws modern apps

Banyak pelanggan belum mengadopsi ***Dev+Ops*** tetapi berkeinginan untuk menggunakannya demi mencari solusi yang dapat membantu mereka mengirim dengan lebih cepat dan kesalahan yang lebih sedikit. Pelanggan lain sudah memulai perjalanan ***Dev+Ops*** mereka, tetapi kesulitan dalam mencapai level kecepatan dan kesuksesan yang mereka harapkan. Dalam banyak kasus, masalah yang dihadapi pelanggan adalah mereka menganggap hal ini terlalu sulit.

1. Memulainya adalah hal yang sangat sulit. Ini tampak seperti sesuatu yang mengecilkan hati. Ada banyak hal yang perlu dipelajari dan perlu dilakukan untuk menerapkan perubahan.
2. Solusi industri yang ada saat ini tidak mempertimbangkan kebutuhan korporasi secara memadai: keamanan, kepatuhan, ketersediaan tinggi, kontrol akses, dll. Hasilnya, tim DevOps menghabiskan banyak waktu untuk membangun solusi, dan solusi ini ujung-ujungnya akan diganti.
3. Banyak solusi industri yang tidak melakukan hal yang sebagaimana mestinya untuk mempermudah ***Dev+Ops*** bagi tim DevOps. Misalnya, banyak solusi yang sifatnya reaktif, bukan proaktif. Mereka memberi tahu pelanggan apabila ada sesuatu yang tidak berjalan sebagaimana mestinya alih-alih mencegah terjadinya masalah tersebut dengan peringatan sebelumnya dan saran terbaik untuk melanjutkan.

Mengubah budaya adalah bagian tersulit dari ***Dev+Ops*** Modern. Perlu kepemimpinan, waktu, dan komitmen agar semua orang bisa berangkat dari pemahaman yang sama. Seiring developer mulai berpartisipasi dalam lebih banyak tanggung jawab yang dulunya dikelola oleh tim operasi, semua harus mengemban tanggung jawab bersama. Berikut ini adalah sekumpulan cara kerja, baik secara kultural maupun perilaku, yang, ketika diadopsi, memberikan mekanisme untuk membangun organisasi ***Dev+Ops*** yang berperforma tinggi.

***eBook* ini membahas prinsip-prinsip panduan yang diidentifikasi AWS dalam organisasi *Dev+Ops* berperforma tinggi:**

- Praktikkan akuntabilitas
- Ukur diri sendiri
- Lakukan peningkatan bertahap
- Otomatiskan segalanya
- Kodifikasi di mana pun memungkinkan
- Terapkan keamanan ganda
- Standardisasikan berbagai alat

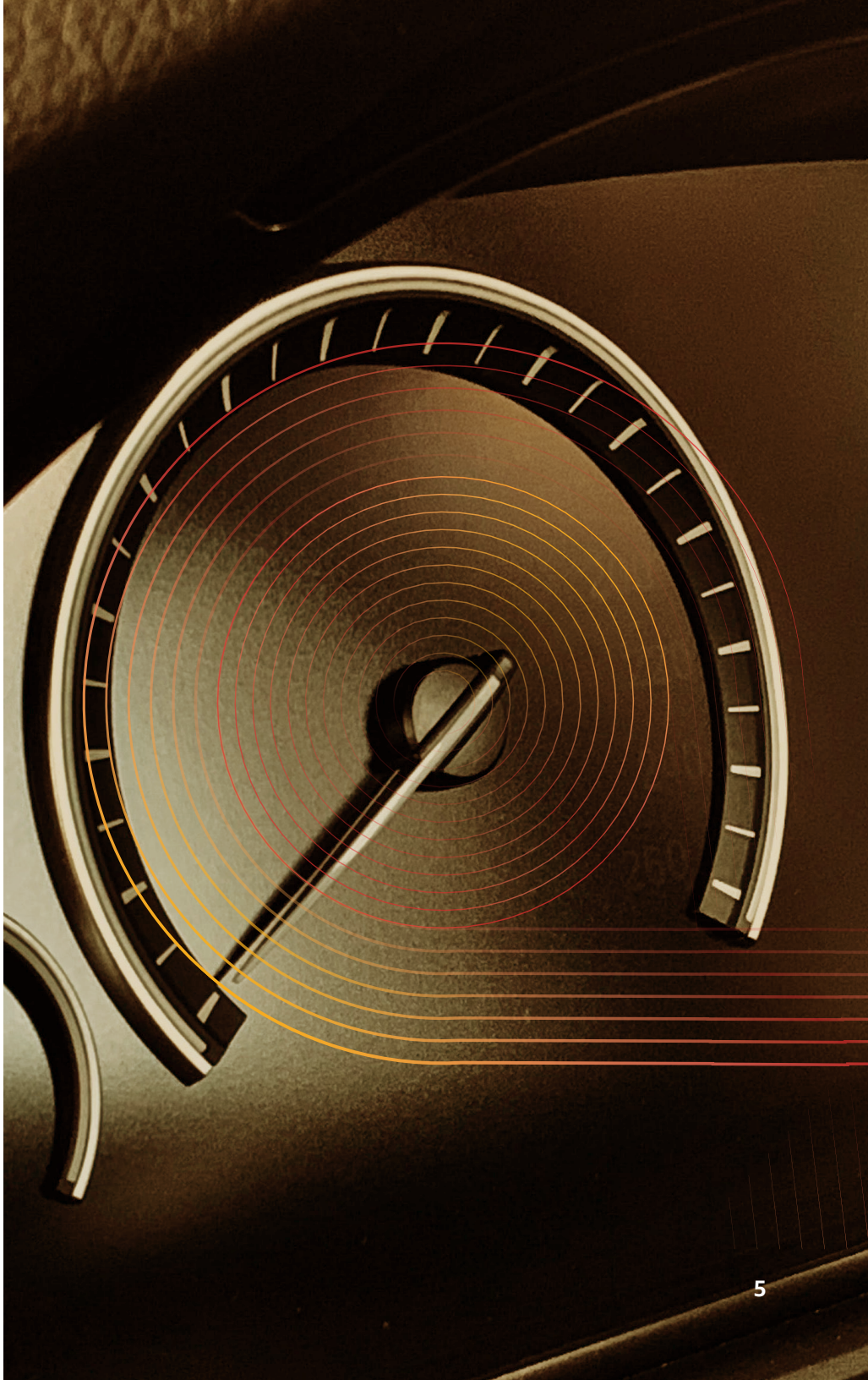
Praktikkan akuntabilitas

Sasaran ***Dev+Ops*** mungkin kecepatan dalam pengiriman perangkat lunak, tetapi niat awalnya adalah untuk mempererat tim pengembangan dan operasional agar mereka bisa berkolaborasi dalam memecahkan masalah. Struktur organisasi memang berubah-ubah seiring waktu, tetapi solusi umumnya adalah fokus pada budaya akuntabilitas. Tim dan individu bisa memiliki spesialisasi dan area keahlian khusus, tetapi mereka harus memiliki akuntabilitas yang sama dalam mengatasi dan memecahkan masalah.

Ukur diri sendiri

Sasaran utama ***Dev+Ops*** adalah kecepatan yang lebih tinggi. Namun tim sering kali tidak tahu harus memulai dari mana. Anda bisa mulai dengan mengukur diri sendiri. Ini artinya mengukur waktu yang saat ini diperlukan untuk merilis. Bagian apa dalam proses pengiriman perangkat lunak Anda yang memakan waktu paling lama, dan apa alasannya? Apakah bisa dipersingkat? Apa metrik performa lain yang penting untuk kesuksesan Anda? Apakah ada area dari proses Anda yang menjadi penghalang antara tim pengembangan dan operasional? Mulailah dengan mengukur input dan hasil dari metrik tersebut, dan Anda akan menemukan fokus bidang serta mendemonstrasikan hasil yang membangun kepercayaan diri dalam tim Anda.

- Waktu rilis
- Hambatan dan penghalang alur
- Kecepatan rilis



Lakukan peningkatan bertahap

Menjadi organisasi ***Dev+Ops*** berperforma tinggi tidak diraih hanya dalam hitungan hari atau minggu. Penting untuk mengidentifikasi beberapa prinsip panduan, lalu memulai dari hal kecil, membuat peningkatan yang stabil dan bertahap. Anda juga tidak akan dapat mempercepat proses perangkat lunak jika Anda masih terpaku pada perubahan dan peluncuran besar. Manusia lebih mudah untuk memproses perubahan kecil dan bertahap dengan sedikit hal baru untuk dipahami, daripada perubahan besar, luas, dan kompleks.

Otomatiskan segalanya

Tujuan dari ***Dev+Ops*** adalah agar segalanya tetap berjalan secara kontinu. Otomatisasi meminimalkan waktu dan kesalahan terkait campur tangan manusia. Ada banyak cara untuk menambahkan otomatisasi ke alur kerja ***Dev+Ops*** Anda, tetapi yang paling dasar adalah dengan menambahkan alur CI/CD ***Dev+Ops*** dengan ***rollback***, ***deployment***, dan uji otomatis.

Kodifikasi di mana pun memungkinkan

Bagian lain dari trik otomatisasi dan ***Dev+Ops*** adalah mengodifikasi hal-hal seperti infrastruktur dan kebijakan Anda, sehingga dapat dipahami oleh manusia dan mesin. Kasus penggunaan paling dasarnya adalah infrastruktur sebagai ***code*** (IaC), yang mengotomatiskan penyediaan infrastruktur ***cloud*** seperti yang Anda tentukan, memberikan rancangan untuk membantu Anda memantau perubahan dan membagikan praktik terbaik di organisasi. Contoh lain adalah kebijakan sebagai kode, yang memungkinkan tim pusat untuk menulis aturan yang diterapkan dengan otomatisasi.

File infrastruktur sebagai code mencakup:

- **Komputasi**
- **Penyimpanan**
- **Identitas, Akses, Keamanan**
- **Sumber daya aplikasi**

Terapkan keamanan ganda

Penerapan praktik terbaik dipastikan dengan mendorong dan melaksanakannya. Mendorong praktik terbaik dapat dilakukan dengan menggunakan templat praktik terbaik yang dibagikan di organisasi, biasanya dengan infrastruktur sebagai code (IaC). Melaksanakan praktik terbaik dapat dilakukan melalui proses seperti kebijakan sebagai kode. Namun batas perlindungan ganda lainnya adalah menggunakan aturan dan kecerdasan buatan untuk menemukan potensi kesalahan sebelum berdampak pada pengguna dan memberikan panduan kepada tim ***Dev+Ops*** tentang cara menerapkan praktik terbaik.

Standardisasikan berbagai alat

Menggunakan alat terstandardisasi memungkinkan organisasi untuk melakukan penskalaan. Menulis kebijakan dan memastikan praktik terbaik jauh lebih sulit, bahkan bisa saja tidak mungkin, untuk diterapkan di sekumpulan alat yang berbeda. Menggunakan alat dari vendor yang telah dirancang untuk bekerja sama dapat membantu, tetapi semakin banyak standardisasi akan lebih menghemat waktu dan mengurangi kesalahan.

Mengapa AWS?

AWS berada dalam posisi unik untuk membantu pelanggan mempercepat transformasi ***cloud*** mereka dengan menyediakan serangkaian pelatihan, alat, dan praktik ***Modern Dev+Ops*** inti. Dengan AWS, pelanggan dapat menikmati manfaat ***cloud*** lebih cepat dan mempercepat inovasi mereka.

1

Amazon adalah pelopor dalam *Dev+Ops* dan inovasi

Amazon telah memelopori banyak praktik terbaik yang umum di ***Dev+Ops*** dan telah berpengalaman lebih dari satu dekade melayani pelanggan berperforma tinggi untuk memberikan solusi yang sesuai dengan kebutuhan mereka. Layanan AWS menggabungkan praktik terbaik Amazon secara ***default***.

2

AWS memiliki portofolio terluas dan terdalam dari layanan *Modern Dev+Ops* yang terintegrasi

Dari pemantauan dan layanan inti CI/CD hingga infrastruktur sebagai ***code*** (IaC) dan layanan terbaru yang menggunakan ***big data*** serta AI untuk memberikan saran dan rekomendasi proaktif, AWS menyediakan semua layanan yang Anda perlukan untuk membangun organisasi atau tim ***Dev+Ops*** berperforma tinggi.

3

Ketersediaan penting, tetapi keamanan lebih penting

Layanan AWS ***Dev+Ops*** dibangun dengan praktik terbaik keamanan secara ***default***. Misalnya, alur CI/CD secara ***default*** diisolasi per proyek, sehingga mempermudah kontrol izin hanya untuk yang memerlukannya. Selain itu, semua layanan AWS memiliki ketersediaan tinggi, sehingga tim bisa menjadi produktif.

4

Dibangun untuk developer dan organisasi

Banyak alat developer yang mengesampingkan kebutuhan organisasi. AWS berfokus pada kebutuhan developer dan organisasi, serta memudahkan penerapan kontrol tata kelola dan kepatuhan agar tim developer dapat bekerja dengan cepat dan aman.

Rencana tindakan

Bagaimana cari menerapkan ***Modern Dev+Ops*** di AWS? Berikut ini adalah rekomendasi praktik terbaik yang telah digunakan organisasi dari semua bentuk dan ukuran untuk menerapkan prinsip panduan di atas:

GitOps

GitOps adalah suatu pendekatan yang mana infrastruktur sebagai ***code*** (IaC) di-***host*** di repositori git dan mengikuti proses permintaan penggabungan yang sama seperti kode perangkat lunak aplikasi. GitOps dirancang untuk menghilangkan perubahan ***out-of-band*** pada infrastruktur aplikasi. Infrastruktur sebagai ***code*** (IaC) adalah praktik penentuan infrastruktur aplikasi Anda dalam format kode, sering kali dengan templat atau file, dan mengotomatiskan penyediaan infrastruktur agar sesuai dengan ketentuan file tersebut. Dengan GitOps, file ketentuan infrastruktur di-***host*** di repositori ***git*** dan setiap tindakan mengikuti proses permintaan gabungan seperti kode perangkat lunak aplikasi dan mengalir melalui integrasi berkelanjutan dan otomatisasi pengiriman berkelanjutan (CI/CD). Dalam kasus GitOps, file infrastruktur diuji dan penyediaan infrastruktur diotomatiskan dengan ***rollback*** ke status sebelumnya jika terjadi kesalahan.

AWS menyediakan solusi GitOps lengkap yang merupakan bagian dari AWS CloudFormation. AWS CloudFormation memungkinkan developer dan tim operasi menentukan, menyediakan, dan mengelola file infrastruktur sebagai ***code*** di konsol dengan bahasa infrastruktur umum seperti YAML atau JSON atau dengan AWS Cloud Development Kit (AWS CDK) dengan mudah. AWS CDK memungkinkan developer menentukan templat CloudFormation dengan bahasa pemrograman umum seperti Python, TypeScript, C#, Java, dan JavaScript. CloudFormation memiliki integrasi terbaik dengan AWS CodePipeline, yang memungkinkan alur kerja CI/CD otomatis dan terintegrasi dengan layanan kontrol versi populer seperti AWS CodeCommit, GitLab, Bitbucket, dan GitHub. Saat file ketentuan infrastruktur baru dibuat, file tersebut di-***host*** di lingkungan dengan versi terkontrol. Saat templat CloudFormation diubah, otomatisasi CI/CD terpicu, dan developer dapat men-***deploy*** atau me-***rollback***/meneruskan perubahan mereka dengan mudah.

"Seiring dengan perkembangan Rivian yang begitu pesat, kami memerlukan sistem yang dapat diskalakan dalam jumlah besar. Perubahan yang biasanya memakan waktu lima hari, kini terjadi hanya dalam hitungan menit."[3]

—Surendra Balu, Kepala Teknis (*Technical Lead*)
3DExperience, Rivian

[3] *https://aws.amazon.com/solutions/case-studies/rivian-case-study/*

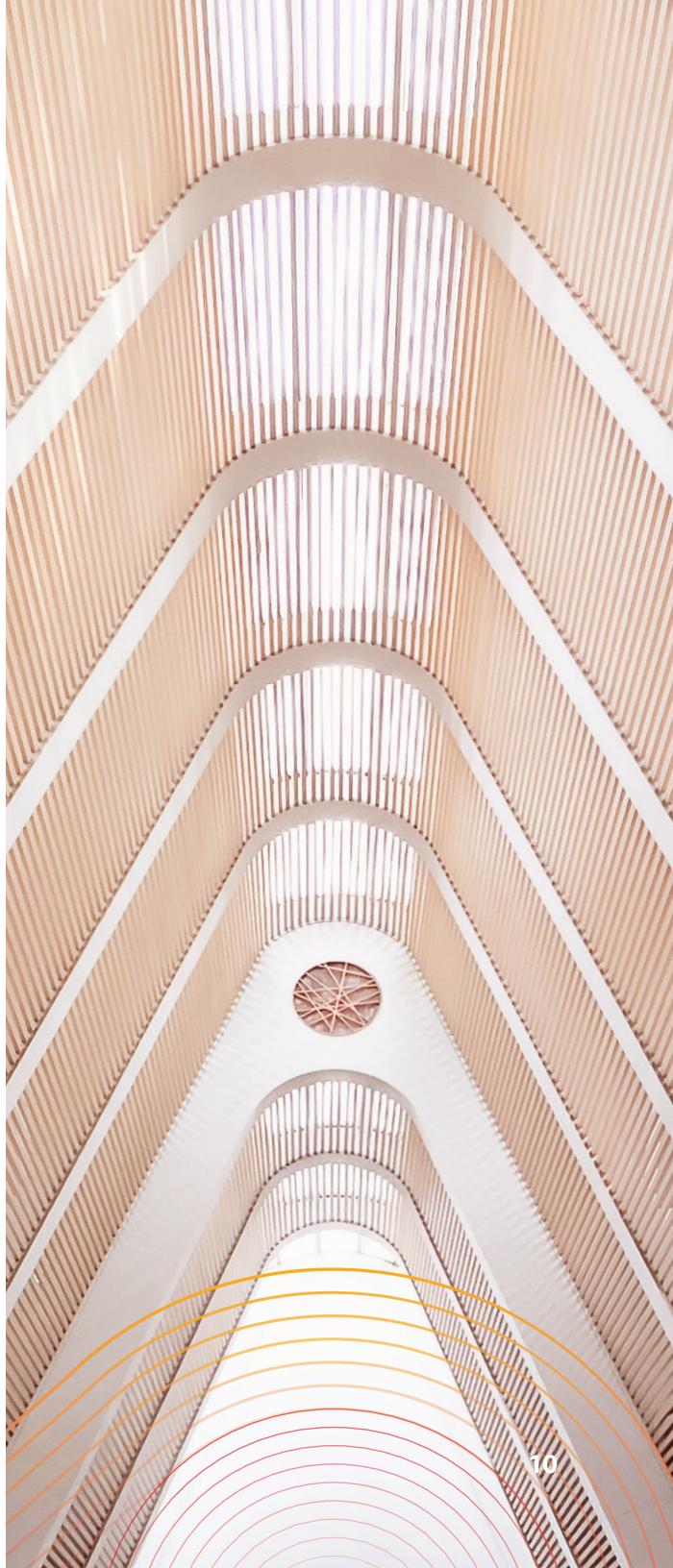
Integrasi berkelanjutan (CI) dan pengiriman berkelanjutan (DI)

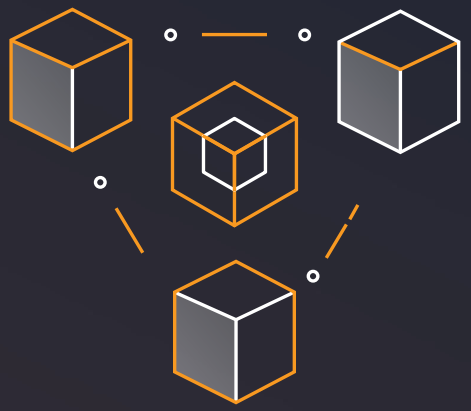
CI/CD adalah praktik terbaik dasar dari otomatisasi persiapan perangkat lunak untuk peluncuran menggunakan alur yang mencakup pembangunan, pengujian, dan ***deployment***. CI/CD membantu tim untuk berkembang lebih cepat dan mengurangi kesalahan dengan menghilangkan proses manual yang rawan kesalahan dan kebutuhan untuk mengawasi peluncuran perangkat lunak. Untuk pelanggan dengan waktu peluncuran lebih lama, CI/CD mengatasi masalah ini dengan melakukan perubahan sistem secara berkelanjutan. Rekayasawan membuat pengujian yang mendeteksi potensi kesalahan atau pelanggaran dalam kode dan menolak pembaruan ke tim. Jika ***bug*** dikirimkan ke produksi, tim dapat mengotomatiskan ***rollback*** ke versi stabil dan mempertahankan waktu aktif.

AWS menyediakan solusi terkelola penuh untuk membangun CI/CD di perusahaan rintisan atau korporasi. AWS CodeBuild, AWS CodeArtifact, AWS CodePipeline, dan AWS CodeDeploy adalah serangkaian layanan ***Dev+Ops*** terhubung yang dibuat khusus untuk membantu pelanggan mengimplementasikan CI/CD. Layanan tersebut mengambil pelajaran dari alat CI/CD internal Amazon dan mendorong praktik terbaik secara ***default***. Misalnya, dengan mengisolasi setiap proyek dalam satu alur CodePipeline, pelanggan yang memprioritaskan keamanan dapat dengan mudah mengunci izin hanya untuk mereka yang seharusnya memiliki akses ke proyek, jadi tidak perlu mengkhawatirkan kerentanan di server pembuatan bersama. Selain itu, AWS memudahkan CI/CD dan memerlukan keahlian yang lebih sedikit. Layanan seperti Amazon CodeGuru Reviewer dapat membantu dalam fase ***pull-request*** dari siklus peluncuran perangkat lunak dengan menambah tinjauan kode dengan ***machine learning*** yang membantu mengidentifikasi masalah penting, kerentanan keamanan, dan ***bug*** yang sulit ditemukan selama pengembangan. Developer menerima panduan dan dapat memperbaiki kode dengan menyelesaikan masalah sebelum fitur baru di-***deploy*** ke produksi. Selain itu, AWS Proton adalah platform layanan yang dapat digunakan oleh tim rekayasawan untuk melakukan pra-konfigurasi berbagai alat yang diperlukan untuk penyediaan infrastruktur, ***deployment*** kode, pemantauan, dan pembaruan untuk tim pengembangan mereka.

"Dengan AWS CodeBuild, pembangunan aplikasi kami sekarang memerlukan waktu sekitar 10 menit; saat masih menggunakan Jenkins, dulu bisa mencapai 1 jam. Untuk mendapatkan performa yang sama, Jenkins empat kali lebih mahal karena kami perlu menjalankan 50 instans Jenkins agar pembangunan dapat selesai secepat itu."

—Bryan Kane, Teknisi Senior (*Senior Engineer*), Coursera





AIOps

AIOps adalah perubahan menuju otomatisasi yang lebih banyak dan mekanisme yang lebih proaktif yang memungkinkan tim untuk berinovasi lebih cepat dengan percaya diri. Dirancang untuk mengurangi jumlah pengetahuan yang diperlukan oleh developer dan menambah pengalaman mereka dengan memanfaatkan AI dalam alur kerja ***Dev+Ops***, AIOps dapat memberikan wawasan yang berguna sebelum masalah muncul, membantu tim menjadi proaktif, menerapkan praktik terbaik secara ***default***, dan berinovasi lebih cepat. Misalnya, developer dapat mendeteksi penyimpangan praktik terbaik, masalah konkurensi, dan ***bug*** pengodean umum lainnya sebelum mereka sempat memengaruhi sistem dan mendapatkan rekomendasi yang dapat ditindaklanjuti yang akan meningkatkan kinerja sistem.

Dengan model ***machine learning*** AWS dari unggulan operasional Amazon.com dan AWS selama dua dekade, Amazon CodeGuru Reviewer, layanan alat pengembang yang didukung ML, memudahkan analisis kode untuk memastikan bahwa praktik terbaik keamanan sudah diterapkan di semua tempat. Dengan Amazon CodeGuru Profiler, pelanggan dapat mengidentifikasi metode yang mahal dan tidak efisien dalam menjalankan aplikasi serta memberikan langkah-langkah yang dapat ditindaklanjuti untuk memperbaikinya, sehingga dapat menghemat biaya. Selain itu, dengan Amazon DevOps Guru, pelanggan dapat mengadopsi solusi AIOps untuk meningkatkan performa dan ketersediaan aplikasi. DevOps Guru mengidentifikasi perilaku aplikasi yang tidak wajar dan mendeteksi masalah penting yang dapat menyebabkan potensi pemadaman atau gangguan layanan, serta memberikan rekomendasi untuk memperbaiki masalah tersebut.

"Kami terus mencari cara untuk mengurangi waktu yang diperlukan oleh tim kami untuk menyelesaikan masalah operasional, dan kini kami menggunakan Amazon DevOps Guru serta memanfaatkan wawasannya yang didukung ML untuk membantu mengidentifikasi, menghubungkan, dan memperbaiki masalah operasional dengan cepat. Dengan adanya wawasan Amazon DevOps Guru, tim kami kini dapat menemukan masalah dengan cepat tanpa harus memulai dari awal untuk menemukan akar masalahnya. Tim IT kami telah mengurangi waktu rata-rata untuk pemulihan (MTTR) kami dengan signifikan, dan mempersingkat waktu penyelesaian masalah—sekaligus memastikan pelanggan kami mendapatkan pengalaman pengguna akhir terbaik."

—HCL Technologies

Observabilitas berkelanjutan

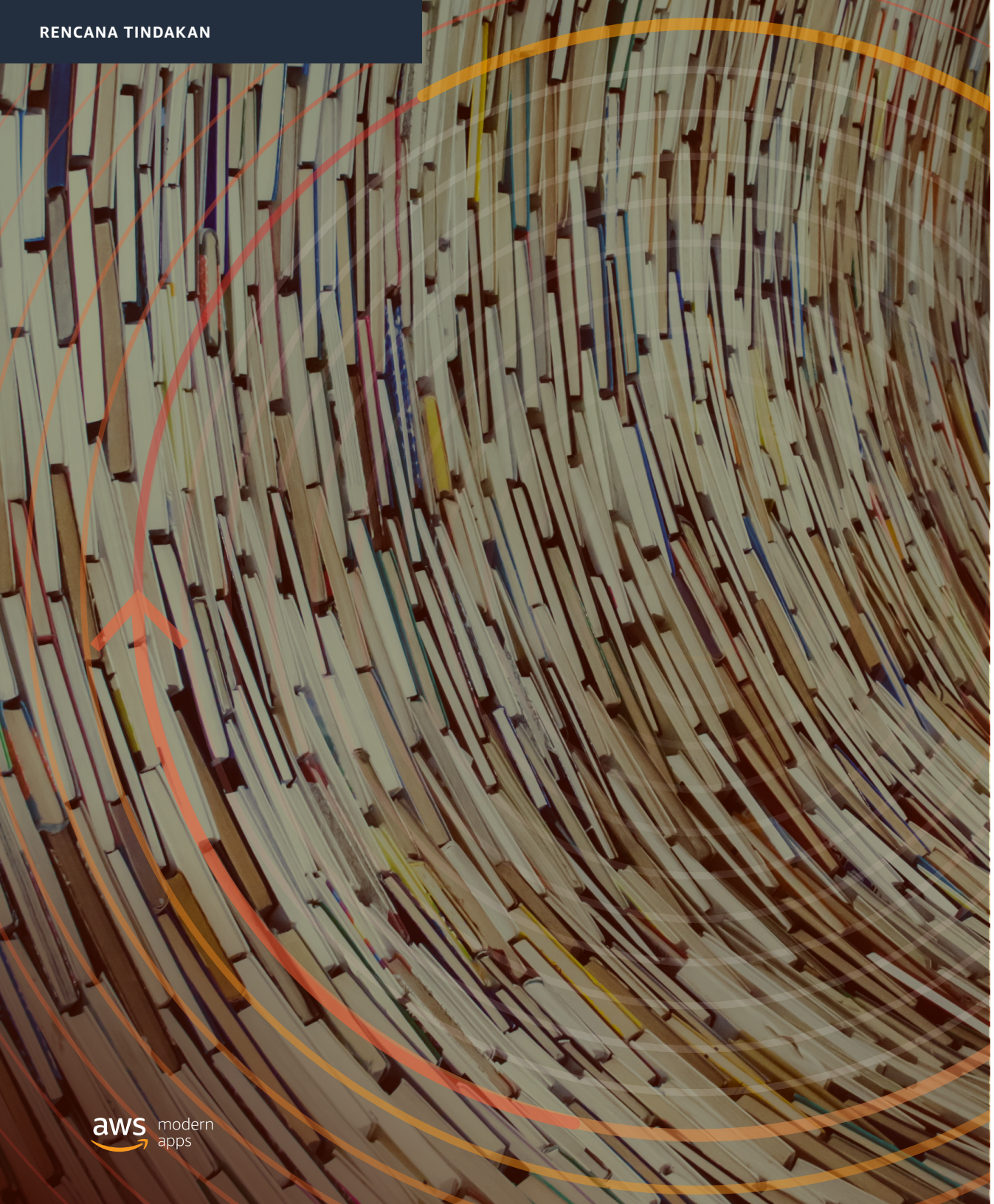
Observabilitas berkelanjutan adalah praktik untuk dapat memahami status sistem Anda pada waktu tertentu. Observabilitas memungkinkan Anda untuk mendeteksi, menyelidiki, dan memperbaiki masalah. Observabilitas berkelanjutan menghadirkan wawasan bagi tim rekayasawan tentang kondisi dan performa aplikasi mereka, sehingga mereka dapat memecahkan serta menyelesaikan masalah yang perlu diperhatikan, dan pada akhirnya akan meningkatkan pengalaman pengguna akhir. Aplikasi dan sumber daya ***cloud*** menghasilkan miliaran metrik, log, serta jejak dalam pengaliran data yang terus-menerus, dan analisis performa di semua aplikasi yang terdistribusi bisa menjadi hal yang cukup menantang. Tingkatkan produktivitas developer dengan kemampuan observabilitas dengan mengidentifikasi dampak pengguna dari sumber mana pun atau menemukan jalur kode yang rusak atau mahal dengan cepat.

Langkah pertama dalam observabilitas berkelanjutan adalah membangun dasbor observabilitas dengan Amazon CloudWatch (atau Amazon Managed Service for Prometheus, Amazon Managed Service for Grafana, atau Amazon Distro for OpenTelemetry, sesuai preferensi). Dasbor observabilitas membantu Anda untuk memantau aktivitas di layanan ***cloud*** kami. Hal ini adalah sistem yang ditampilkan ke pengguna dan memberikan ringkasan tentang perilaku sistem dengan menampilkan data alarm, pelacakan, log, dan metrik deret waktu. Setelah dasbor dibangun, pelanggan sebaiknya menyetel alarm CloudWatch untuk terus mendapatkan pemberitahuan tentang potensi masalah di lingkungan ***cloud*** mereka.

"Kami menggunakan CloudWatch guna membuat pemberitahuan untuk Amazon Simple Queue Service dan untuk menskalakan tindakan untuk Grup EC2 Auto Scaling. Wawasan ini membantu kami dalam membuat keputusan penskalaan, memungkinkan kami untuk mendapatkan penggunaan sumber daya komputasi AWS paling efisien tanpa memengaruhi pengalaman pelanggan kami. Dengan CloudFormation, kami dapat men-deploy tumpukan logis untuk sumber daya AWS yang digunakan oleh aplikasi yang dipakai oleh pelanggan kami, memungkinkan deployment berulang dan konsisten di berbagai akun AWS. Sebagai hasilnya, kami menemukan cara mudah untuk menyusun tumpukan aplikasi menggunakan konfigurasi sebagai kode, dan tidak bergantung pada konfigurasi manual yang rawan kesalahan dan dapat menjadi inkonsisten."

—Martin Costello, Teknisi Senior (*Senior Engineer*), Just Eat

Fakta:
AWS memantau rata-rata 1 kuadriliun (1.000 triliun) observasi metrik setiap bulan



Bangun dasbor untuk visibilitas operasional

Dalam artikel Amazon Builders' Library ini, Teknisi Utama (***Principal Engineer***) AWS, John O'Shea, menjelaskan bagaimana pendapat tim Amazon DevOps tentang pembangunan dasbor untuk memahami status sistem mereka.

Pelajari selengkapnya

Kepatuhan berkelanjutan

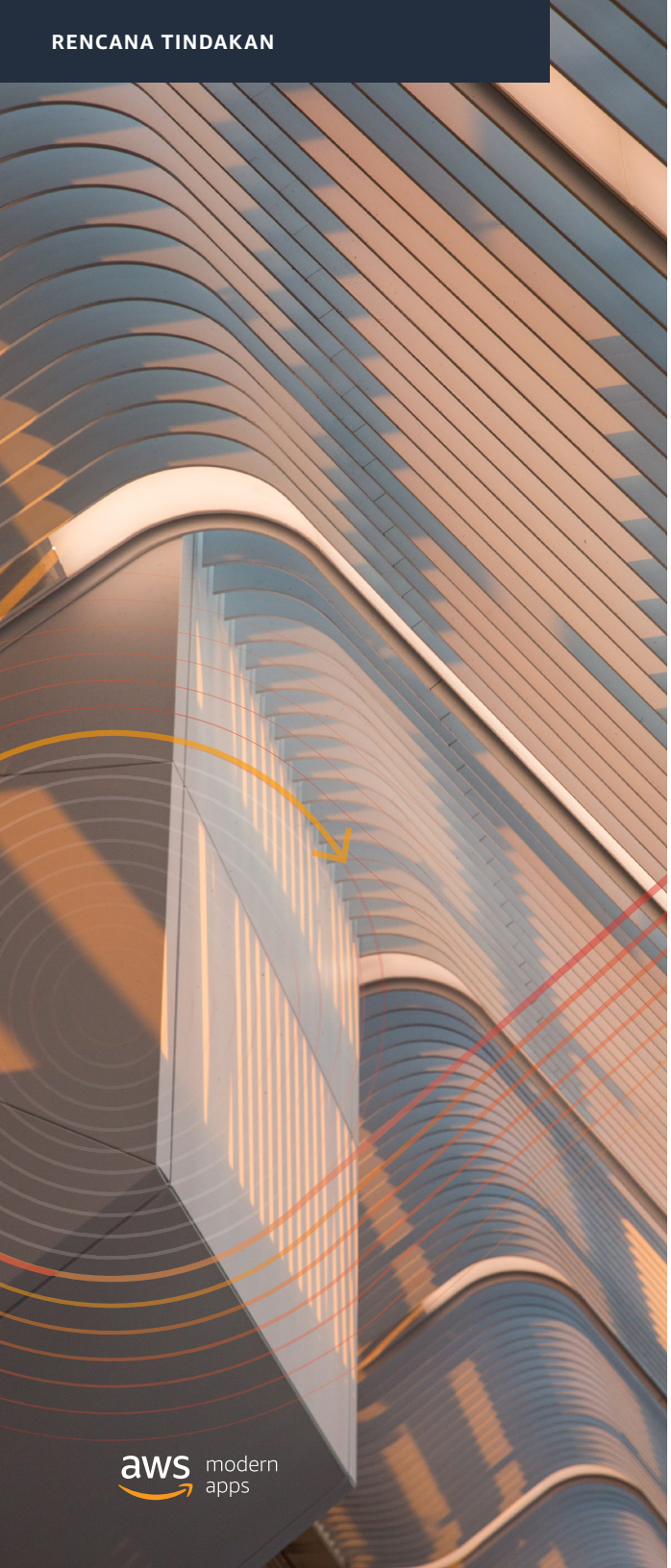
Kepatuhan di ***cloud*** merupakan paradigma baru. Dengan kecepatan dan skala sumber daya ***cloud,*** penggunaan metode tradisional untuk kepatuhan adalah hal yang mustahil. Spreadsheet dan basis data statis tidak dapat menangani ***churn*** sumber daya. Kunci keberhasilannya adalah otomatisasi dan penyederhanaan sebanyak mungkin. AWS menyediakan berbagai layanan untuk membantu pelanggan menyederhanakan kepatuhan ***cloud***. Salah satu cara untuk memahami layanan apa yang digunakan untuk kepatuhan adalah membingkainya dengan ***Three Lines Model*** dari The Institute of Internal Auditors, badan yang diakui secara internasional dalam praktik audit internal. Model ini menetapkan bahwa organisasi mengelola risiko melalui peran dan tanggung jawab IT tertentu bersama tiga lini pertahanan. Lini pertama mengelola risiko, lini kedua mengawasi risiko, dan lini ketiga memberikan jaminan risiko.

AWS menyediakan berbagai layanan yang sesuai dengan setiap lini. Misalnya, untuk layanan pengelolaan risiko di lini pertama, kami memiliki AWS Config, AWS CloudTrail, dan AWS Systems Manager. Layanan ini memungkinkan Anda menentukan dan men-***deploy*** kontrol individual untuk mengelola risiko. Di lini kedua, AWS Security Hub dan Amazon CloudWatch membantu mengawasi risiko untuk seluruh organisasi Anda. Dan AWS Audit Manager adalah alat yang sempurna untuk memberikan jaminan risiko, lini ketiga, dengan memungkinkan pelanggan mengotomatiskan proses pengumpulan bukti kepatuhan untuk audit.

Namun, penggunaan alat ini hanya sebagian dari prosesnya. Pelanggan perlu mengotomatiskan proses dengan kepatuhan berkelanjutan. Untuk mengadopsi kepatuhan berkelanjutan, sebaiknya gunakan konsep kepatuhan sebagai kode. Kepatuhan sebagai kode memungkinkan Anda menentukan elemen ***Three Lines Model*** menggunakan alat seperti AWS CloudFormation. Kemudian Anda dapat menguji dan men-***deploy*** elemen-elemen tersebut menggunakan alur otomatis, seperti AWS CodePipeline. Karena diperlukan kontrol dan bukti baru, elemen-elemen tersebut tidak perlu diterapkan secara manual di setiap akun. Kode diperbarui dalam ketentuan kontrol dan didorong melalui alur. Hasil akhirnya berupa kontrol baru untuk mengelola risiko yang segera ditambahkan sebagai bukti untuk keperluan audit. Manfaat Kepatuhan Berkelanjutan lainnya adalah jaminan nyata bahwa Anda memiliki kontrol dan kumpulan bukti tetap aman tanpa ada yang mengutak-atik. Memanfaatkan ***Three Lines Model*** dan mengotomatiskan proses dengan kepatuhan berkelanjutan adalah pendekatan terbaik untuk menangani kepatuhan dalam skala besar.

"Dengan AWS, kami telah mengubah cara kerja sepenuhnya, dan kini dapat menskalakan tanpa penambahan jumlah karyawan dengan mengotomatiskan proses kepatuhan. Pengelolaan kepatuhan bisa menjadi hal yang rumit, tetapi AWS Config dapat membantu siapa pun, termasuk orang tidak memiliki pengetahuan mendalam tentang kepatuhan, untuk menerapkan kepatuhan dengan templat siap pakai. Dasbor menyediakan informasi kepada pengguna seperti sumber daya yang tidak sesuai, yang dapat digunakan untuk meningkatkan postur keamanan, dan berfungsi sebagai titik awal diskusi dengan seluruh organisasi tentang hal yang perlu diperhatikan dan diperbaiki. Sebagai grup digital yang menyediakan kepemimpinan teknis melalui teknologi cloud untuk tim internal maupun eksternal, penting untuk memiliki visibilitas dan kontrol terpusat atas seluruh postur keamanan organisasi kami. Templat paket kesesuaian Config yang sudah dikemas telah menyederhanakan proses ini, serta merupakan faktor kunci dalam mempercepat migrasi kami secara keseluruhan ke cloud."

—Andrew Clark, Arsitek Solusi Senior (*Senior Solutions Architect*), Baker Tilly Digital



aws modern apps

Bangun platform layanan bersama

Penerapan ***Dev+Ops*** di setiap tim dan organisasi sedikit berbeda. Di banyak organisasi, terdapat tim platform pusat yang membantu mendukung tim pengembangan dan mengurangi beban operasional dengan menstandardisasi keamanan, pengiriman perangkat lunak, pemantauan, dan jaringan. Platform layanan bersama adalah antarmuka layanan mandiri yang dipesan lebih dahulu agar dapat digunakan oleh developer, yang disederhanakan untuk men-***deploy*** kode. Tim pusat memiliki kontrol untuk menentukan standar keamanan, pengiriman perangkat lunak, pemantauan, dan jaringan yang harus digunakan di semua aplikasi yang di-***deploy***. Dengan demikian, developer menjadi lebih produktif dan tim platform memiliki kontrol yang lebih banyak.

AWS menyediakan semua alat yang Anda perlukan untuk membangun platform layanan bersama di AWS. Untuk kasus penggunaan produksi dasar, AWS Copilot CLI memudahkan pembuatan lingkungan pengembangan "***batteries included***" (sangat lengkap) dengan CI/CD multi-akun, grup keamanan, dan pemantauan langsung. Untuk situasi yang lebih kompleks, AWS menyediakan layanan terkelola penuh yang disebut AWS Proton. AWS Proton adalah layanan pengiriman terkelola penuh pertama untuk aplikasi nirserver dan kontainer. Tim platform dapat menggunakan AWS Proton untuk menghubungkan dan mengoordinasikan berbagai alat yang diperlukan untuk penyediaan infrastruktur, ***deployment*** kode, pemantauan, dan pembaruan. AWS Proton mengatasi masalah ini dengan memberi alat yang diperlukan tim platform untuk mengelola kompleksitas ini dan menegakkan standar yang konsisten sekaligus memudahkan developer untuk men-***deploy*** kode menggunakan kontainer dan teknologi nirserver. Jika organisasi Anda memerlukan kontrol lebih untuk membangun platform layanan bersama, banyak organisasi telah berhasil membangun dengan AWS CloudFormation dan AWS Service Catalog untuk membuat dan mengelola katalog layanan IT yang penggunaannya disetujui di AWS. Sama halnya dengan pelanggan yang tim platformnya sudah familier dengan bahasa pemrograman dinamis, mereka menggunakan AWS CDK untuk membangun platform pemberdayaan untuk tim pengembangan mereka.

"Ahli teknologi kami dapat men-deploy pembangunan aplikasi ke produksi dengan mudah menggunakan portal layanan mandiri AWS Service Catalog, tanpa harus membuka tiket. Hasilnya, mereka dapat meluncurkan tumpukan aplikasi baru dalam lima menit, bukan berminggu-minggu seperti pada umumnya. Berkat AWS Service Catalog, kami dapat menerapkan DevOps dan otomatisasi di Wiley."[4]

—Meltem Dincer, Wakil Presiden Kemampuan Platform (*Vice President of Platform Capabilities*), Wiley

[4] *https://aws.amazon.com/solutions/case-studies/wiley/*

Di mana pun tahap perjalanan ***Dev+Ops*** organisasi Anda, AWS menawarkan berbagai solusi yang dirancang untuk menutup kesenjangan antara developer dan tim operasional.

Alat Developer AWS mudah untuk memulai, mencakup kebutuhan korporasi besar seperti keamanan, kepatuhan, ketersediaan tinggi, dan kontrol akses, serta memberikan solusi proaktif kepada pelanggan, yang mengidentifikasi dan mencegah masalah sebelum terjadi dengan peringatan dan saran lanjutan berdasarkan praktik terbaik. Dengan AWS, pelanggan dapat lebih cepat memperoleh manfaat ***cloud*** dan mempercepat inovasi melalui rangkaian inti pelatihan, alat, dan praktik Modern ***Dev+Ops***.

Pelajari selengkapnya tentang DevOps modern ›