Kerangka Kerja AWS Well-Architected

Pilar Pelestarian Lingkungan

Pilar Pelestarian Lingkungan: Kerangka Kerja AWS Well-Architected

Table of Contents

Pilar Pelestarian Lingkungan - Kerangka Kerja AWS Well-Architected

Tanggal publikasi: 3 Oktober 2023 (Revisi Dokumen)

Laporan resmi ini berfokus pada pilar pelestarian lingkungan Kerangka Kerja Amazon Web Services (AWS) Well-Architected. Laporan ini memberikan prinsip desain, panduan operasional, praktik terbaik, potensi kompensasi, dan rencana peningkatan yang dapat Anda gunakan untuk memenuhi target pelestarian lingkungan untuk beban kerja AWS Anda.

Pengantar

AWS Well-Architected Framework membantu Anda mengetahui kelebihan dan kekurangan keputusan yang Anda ambil saat membangun beban kerja di AWS. Kerangka Kerja ini membantu Anda mempelajari praktik terbaik arsitektur untuk mendesain dan mengoperasikan beban kerja yang aman, andal, efisien, hemat biaya, dan ramah lingkungan di AWS Cloud. Kerangka Kerja ini menyediakan cara untuk secara terus menerus menilai arsitektur Anda berdasarkan praktik terbaik dan mengidentifikasi area yang perlu diperbaiki. Dengan memiliki beban kerja yang dirancang dengan baik, kemampuan Anda untuk mendukung hasil bisnis dapat meningkat pesat.

Kerangka Kerja ini disokong oleh enam pilar:

- Keunggulan operasional
- Keamanan
- Keandalan
- Efisiensi kinerja
- Optimasi biaya
- Pelestarian Lingkungan

Dokumen ini berfokus pada pilar keenam, yakni menitikberatkan pada cakupan pelestarian lingkungan. Dokumen ini dimaksudkan untuk orang-orang yang memiliki peran di bidang teknologi, seperti kepala pejabat teknologi (CTO), arsitek, pengembang, dan anggota tim operasi.

Setelah membaca dokumen ini, Anda akan memahami saran dan strategi AWS terkini untuk digunakan ketika merancang arsitektur cloud dengan mempertimbangkan pelestarian lingkungan.

Dengan mengadopsi praktik-praktik dalam laporan ini, Anda dapat membangun arsitektur yang memaksimalkan efisiensi dan mengurangi limbah.

Pelestarian lingkungan cloud

Bidang pelestarian lingkungan membahas tentang dampak jangka panjang aktivitas bisnis Anda terhadap lingkungan, ekonomi, dan masyarakat. Komisi Dunia untuk Lingkungan dan Pembangunan PBB mendefinisikan pembangunan berkelanjutan sebagai “pembangunan yang memenuhi kebutuhan saat ini tanpa mengorbankan kemampuan generasi mendatang untuk memenuhi kebutuhan mereka sendiri.” Bisnis atau organisasi Anda dapat memberikan dampak negatif terhadap lingkungan seperti emisi karbon langsung atau tidak langsung, limbah yang tidak dapat didaur ulang, dan kerusakan pada sumber daya bersama seperti air bersih.

Saat membangun beban kerja cloud, praktik pelestarian lingkungannya adalah memahami dampak layanan yang digunakan, menghitung dampak melalui seluruh siklus hidup beban kerja, dan menerapkan prinsip-prinsip desain dan praktik terbaik untuk mengurangi dampak-dampak ini. Dokumen ini berfokus pada dampak lingkungan, terutama konsumsi dan efisiensi energi, karena ini merupakan pendorong penting bagi arsitek untuk melandasi tindakan langsung untuk mengurangi penggunaan sumber daya.

Saat berfokus pada dampak lingkungan, Anda harus memahami bagaimana dampak ini umumnya dipertimbangkan serta dampak susulan terhadap pengukuran emisi organisasi Anda sendiri. Greenhouse Gas Protocol (Protokol Gas Rumah Kaca) mengatur emisi karbon ke dalam cakupan-cakupan berikut ini, beserta contoh-contoh emisi untuk tiap-tiap cakupan yang relevan bagi penyedia cloud seperti AWS:

- Cakupan 1: Semua emisi langsung dari aktivitas suatu organisasi atau di bawah kendalinya. Contohnya adalah pembakaran gas oleh generator cadangan pusat data.
- Cakupan 2: Emisi tidak langsung dari listrik yang dibeli dan digunakan untuk menghidupkan pusat data dan fasilitas lain. Contohnya adalah emisi dari pembangkit daya komersial.
- Cakupan 3: Semua emisi tidak langsung lainnya dari aktivitas suatu organisasi dari sumber yang tidak dikendalikannya. Contoh terkait AWS antara lain emisi yang berhubungan dengan pembangunan pusat data, dan produksi serta pengangkutan perangkat keras IT yang di-deploy di pusat data.

Dari sudut pandang pelanggan AWS, emisi dari beban kerja Anda yang berjalan di AWS dianggap sebagai emisi tidak langsung, dan bagian dari emisi Cakupan 3 Anda. Tiap-tiap beban kerja yang di-deploy menghasilkan sebagian dari total emisi AWS dari tiap-tiap cakupan sebelumnya. Jumlah yang sebenarnya berbeda-beda untuk setiap beban kerja dan bergantung pada sejumlah faktor termasuk

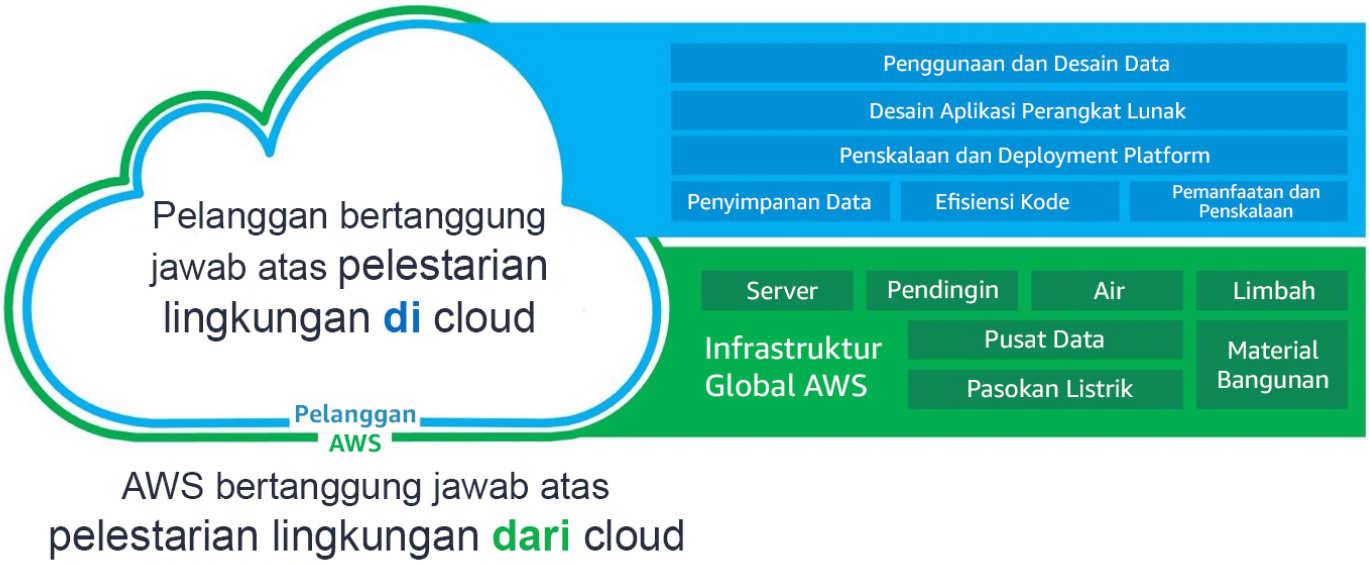
layanan AWS yang digunakan, energi yang dipakai oleh layanan tersebut, intensitas karbon jaringan listrik yang mengaliri pusat data AWS tempatnya berjalan, dan pengadaan energi terbarukan oleh AWS.

Dokumen ini terlebih dahulu menjelaskan model tanggung jawab bersama untuk pelestarian lingkungan, lalu menyediakan praktik terbaik arsitektur sehingga Anda dapat meminimalkan dampak beban kerja Anda dengan mengurangi total sumber daya yang diperlukan beban kerja untuk berjalan di pusat data AWS.

Model tanggung jawab bersama

Pelestarian lingkungan adalah tanggung jawab bersama antara pelanggan dan AWS.

- AWS bertanggung jawab untuk mengoptimalkan pelestarian lingkungan dari cloud - dengan menyediakan infrastruktur bersama yang efisien, penatagunaan air, dan pengadaan energi terbarukan.
- Pelanggan bertanggung jawab untuk pelestarian lingkungan di cloud - dengan mengoptimalkan pemanfaatan beban kerja dan sumber daya, dan meminimalkan total sumber daya yang diperlukan untuk di-deploy untuk beban kerja Anda.



Model tanggung jawab bersama

Pelestarian lingkungan dari cloud

Penyedia cloud memiliki jejak karbon yang lebih rendah dan lebih hemat energi daripada alternatif on-premise biasa karena penyedia cloud berinvestasi dalam teknologi energi dan pendingin yang efisien, mengoperasikan kumpulan server yang hemat energi, dan mencapai tingkat pemanfaatan server yang tinggi. Beban kerja cloud mengurangi dampak dengan cara memanfaatkan sumber daya bersama, seperti jaringan, daya, pendingin, dan fasilitas fisik. Anda dapat memigrasikan beban kerja cloud Anda ke teknologi yang lebih efisien jika tersedia dan menggunakan layanan berbasis cloud untuk mentransformasikan beban kerja Anda demi pelestarian lingkungan yang lebih baik.

Sumber daya

- Peluang Penurunan Karbon dengan Beralih ke Amazon Web Services
- AWS menghadirkan solusi pelestarian lingkungan

Pelestarian lingkungan di cloud

Pelestarian lingkungan di cloud adalah sebuah upaya berkelanjutan yang difokuskan terutama pada pengurangan dan efisiensi energi di semua komponen beban kerja dengan mencapai manfaat maksimum dari sumber daya yang disediakan dan meminimalkan total sumber daya yang diperlukan. Upaya ini bisa mencakup pemilihan bahasa pemrograman yang efisien di awal, adopsi algoritme modern, penggunaan teknik penyimpanan data yang efisien, men-deploy ke infrastruktur komputasi yang efisien dan terukur dengan tepat, serta meminimalkan kebutuhan perangkat keras pengguna akhir berdaya tinggi.

Pelestarian lingkungan melalui cloud

Selain meminimalkan dampak beban kerja yang telah Anda deploy, Anda dapat menggunakan AWS Cloud untuk menjalankan beban kerja yang dirancang untuk mendukung tantangan pelestarian lingkungan Anda yang lebih luas. Contoh tantangan ini antara lain pengurangan emisi karbon, penurunan konsumsi energi, daur ulang air, atau pengurangan limbah di area lain dalam bisnis atau organisasi Anda.

Pelestarian lingkungan melalui cloud adalah ketika Anda menggunakan teknologi AWS untuk menyelesaikan tantangan pelesatarian lingkungan yang lebih luas. Sebagai contoh, Anda dapat menggunakan layanan machine learning seperti Amazon Monitron untuk mendeteksi perilaku tidak normal dalam mesin-mesin industri. Dengan data deteksi ini, Anda dapat melakukan pemeliharaan

preventif untuk mengurangi risiko insiden lingkungan yang disebabkan oleh kegagalan peralatan yang tidak terduga dan memastikan mesin dapat terus beroperasi dengan efisiensi optimal.

Prinsip desain untuk pelestarian lingkungan di cloud

Terapkan prinsip-prinsip desain ini saat merancang beban kerja cloud Anda guna memaksimalkan pelestarian lingkungan dan meminimalkan dampak.

- Pahami dampak Anda: Ukur dampak beban kerja cloud Anda dan buat model dampak beban kerja Anda untuk masa mendatang. Sertakan semua sumber dampak, termasuk dampak akibat penggunaan produk Anda oleh pelanggan, serta dampak yang muncul dari penonaktifan dan penghentian produk. Bandingkan output produktif dengan total dampak beban kerja cloud Anda dengan meninjau sumber daya dan emisi yang diperlukan per unit kerja. Gunakan data ini untuk membuat indikator kinerja utama (KPI), evaluasi cara-cara untuk meningkatkan produktivitas sambil mengurangi dampak, dan perkirakan dampak perubahan yang diajukan seiring waktu.
- Tetapkan tujuan pelestarian lingkungan: Untuk tiap-tiap beban kerja cloud, tetapkan tujuan pelestarian lingkungan jangka panjang seperti mengurangi sumber daya komputasi dan penyimpanan yang diperlukan per transaksi. Modelkan laba atas investasi peningkatan pelestarian lingkungan untuk beban kerja yang ada, dan beri pemilik sumber daya yang mereka perlukan untuk berinvestasi dalam tujuan pelestarian lingkungan. Rencanakan pertumbuhan, dan rancang beban kerja Anda agar pertumbuhan menghasilkan penurunan intensitas dampak yang terukur berdasarkan unit yang tepat, seperti per pengguna atau per transaksi. Tujuan ini membantu Anda mendukung tujuan pelestarian lingkungan yang lebih luas untuk bisnis atau organisasi Anda, mengidentifikasi regresi, dan memprioritaskan area-area dengan potensi perbaikan.
- Maksimalkan pemanfaatan: Sesuaikan ukuran beban kerja dan implementasikan desain yang efisien untuk memastikan pemanfaatan yang tinggi dan memaksimalkan efisiensi energi untuk perangkat keras yang mendasari. Dua host yang berjalan dengan pemanfaatan 30% memiliki efisiensi yang lebih rendah daripada satu host dengan pemanfaatan 60% dikarenakan konsumsi daya dasar per host. Pada saat yang sama, singkirkan atau minimalkan sumber daya, pemrosesan, dan penyimpanan yang tidak aktif untuk mengurangi total energi yang diperlukan untuk menjalankan beban kerja Anda.
- Antisipasi dan adopsi penawaran perangkat keras dan perangkat lunak baru yang lebih efisien: Dukung peningkatan hulu yang dibuat oleh partner dan pemasok Anda untuk membantu Anda mengurangi dampak beban kerja cloud Anda. Terus pantau dan evaluasi penawaran perangkat keras dan perangkat lunak baru yang lebih efisien. Rancang fleksibilitas untuk memungkinkan pengadopsian teknologi efisien baru secara cepat.

- Gunakan layanan terkelola: Berbagi layanan di seluruh basis pelanggan yang luas dapat membantu memaksimalkan pemanfaatan sumber daya, sehingga mengurangi jumlah infrastruktur yang diperlukan untuk mendukung beban kerja cloud. Sebagai contoh, pelanggan dapat berbagi dampak komponen pusat data yang sama seperti daya dan jaringan dengan memigrasikan beban kerja ke AWS Cloud dan mengadopsi layanan terkelola, seperti AWS Fargate untuk kontainer nirserver, tempat AWS beroperasi pada skala besar dan bertanggung jawab atas operasi efisien mereka. Gunakan layanan terkelola yang dapat membantu meminimalkan dampak Anda, seperti memindahkan data yang jarang diakses ke penyimpanan dingin secara otomatis dengan konfigurasi Amazon S3 Lifecycle atau Amazon EC2 Auto Scaling untuk menyesuaikan kapasitas guna memenuhi permintaan.
- Kurangi dampak hilir beban kerja cloud Anda: Kurangi jumlah energi atau sumber daya yang diperlukan untuk menggunakan layanan Anda. Kurangi atau singkirkan kebutuhan pelanggan untuk meningkatkan perangkat mereka untuk menggunakan layanan Anda. Uji menggunakan device farm untuk memahami dampak yang diperkirakan dan uji dengan pelanggan untuk memahami dampak riil dari penggunaan layanan Anda.

Proses peningkatan

Proses peningkatan arsitektur mencakup pemahaman tentang apa yang Anda miliki dan apa yang dapat Anda lakukan untuk melakukan peningkatan, menyeleksi target peningkatan, menguji peningkatan, mengadopsi peningkatan yang berhasil, menghitung keberhasilan, dan membagikan pelajaran yang Anda dapatkan sehingga dapat direplikasi di tempat lain, lalu mengulangi siklus ini.

Tujuan peningkatan Anda antara lain:

- Untuk menghilangkan pemborosan, pemanfaatan yang rendah, dan sumber daya yang tidak aktif atau tidak digunakan
- Untuk memaksimalkan nilai dari sumber daya yang Anda pakai

> **Note**
>
> Gunakan semua sumber daya yang Anda sediakan, dan selesaikan tugas yang sama dengan sumber daya seminimal mungkin.

Pada tahap-tahap awal optimalisasi, terlebih dahulu singkirkan area dengan pemborosan atau pemanfaatan yang rendah, lalu beralih ke optimalisasi yang lebih tertarget yang sesuai dengan beban kerja khusus Anda.

Pantau perubahan pada pemakaian sumber daya dari waktu ke waktu. Identifikasi perubahan terkumpul yang mengakibatkan peningkatan pemakaian sumber daya yang tidak efisien atau terlalu besar. Tentukan kebutuhan peningkatan untuk menangani perubahan pada pemakaian dan implementasikan peningkatan yang Anda prioritaskan.

Langkah-langkah berikut ini dirancang sebagai proses iteratif yang mengevaluasi, memprioritaskan, menguji, dan melakukan deployment peningkatan untuk beban kerja cloud yang berfokus pada pelestarian lingkungan.

1. Identifikasi target peningkatan: Tinjau beban kerja Anda dengan mengacu praktik terbaik untuk pelestarian lingkungan yang disebutkan dalam dokumen ini, dan identifikasi target-target peningkatan.
2. Evaluasi peningkatan khusus: Evaluasi perubahan-perubahan khusus untuk mengetahui potensi peningkatan, biaya yang diperkirakan, dan risiko bisnis.

3. Prioritaskan dan rencanakan peningkatan: Prioritaskan perubahan yang menawarkan peningkatan terbesar dengan biaya dan risiko terkecil, dan buat rencana pengujian dan implementasi.
4. Uji dan validasikan peningkatan:: Implementasikan perubahan di lingkungan pengujian untuk memvalidasi potensi peningkatannya.
5. Terapkan perubahan ke produksi: Implementasikan perubahan di lingkungan produksi.
6. Ukur hasil dan replikasi keberhasilan: Cari peluang untuk mereplikasi keberhasilan di beban kerja, dan batalkan perubahan dengan hasil yang tidak dapat diterima.

Contoh skenario

Contoh skenario berikut ini akan disebutkan di bagian lain dokumen ini untuk menggambarkan tiap-tiap langkah proses peningkatan.

Perusahaan Anda memiliki beban kerja yang menjalankan manipulasi gambar kompleks di instans Amazon EC2 dan menyimpan file asli dan yang telah dimodifikasi agar dapat diakses pengguna. Aktivitas pemrosesannya memerlukan CPU besar, dan file output-nya amat sangat besar.

Identifikasi target peningkatan

Pahami praktik terbaik yang dapat membantu Anda mencapai tujuan pelestarian lingkungan Anda. Anda dapat menemukan deskripsi mendetail tentang praktik-praktik terbaik ini serta rekomendasi untuk peningkatan di bagian mendatang dokumen ini.

Tinjau beban kerja Anda serta sumber daya yang digunakan. Identifikasi area penting seperti deployment yang besar dan sumber daya yang sering digunakan. Evaluasi area-area penting ini untuk mengetahui peluang peningkatan pemanfaatan yang efektif pada sumber daya Anda dan untuk mengurangi sumber daya total yang diperlukan untuk mencapai hasil bisnis Anda.

Tinjau beban kerja Anda dengan mengacu pada praktik terbaik, dan identifikasi target-target peningkatan.

Dengan menerapkan langkah ini ke Contoh skenario, Anda mengidentifikasi praktik-praktik terbaik berikut ini sebagai potensi target peningkatan:

- Gunakan perangkat keras dalam jumlah minim untuk memenuhi kebutuhan Anda
- Gunakan teknologi yang mendukung pola akses dan penyimpanan data

Sumber daya

- Mengoptimalkan Infrastruktur AWS untuk Pelestarian Lingkungan, Bagian I: Komputasi
- Mengoptimalkan Infrastruktur AWS untuk Pelestarian Lingkungan, Bagian II: Penyimpanan
- Mengoptimalkan Infrastruktur AWS untuk Pelestarian Lingkungan, Bagian III: Jaringan

Evaluasi peningkatan khusus

Pahami sumber daya yang disediakan oleh beban kerja Anda untuk menyelesaikan unit kerja. Evaluasi potensi peningkatan, dan perkirakan potensi dampaknya, biaya untuk mengimplementasikannya, dan risiko-risiko terkait.

Untuk mengukur peningkatan dari waktu ke waktu, terlebih dahulu pahami apa yang telah Anda sediakan di AWS dan bagaimana sumber daya tersebut digunakan.

Mulailah dengan gambaran umum penuh tentang penggunaan AWS Anda, dan gunakan Laporan Biaya dan Penggunaan AWS untuk membantu mengidentifikasi area-area penting. Gunakan sampel kode AWS ini untuk membantu Anda meninjau dan menganalisis laporan Anda dengan bantuan Amazon Athena.

Metrik proksi

Ketika mengevaluasi perubahan tertentu, Anda juga harus mengevaluasi metrik apa saja yang paling mewakili efek perubahan tersebut pada sumber daya terkait. Metrik-metrik ini disebut metrik proksi. Pilih metrik proksi yang paling mencerminkan tipe peningkatan yang sedang Anda evaluasi serta sumber daya yang ditargetkan oleh peningkatan. Metrik-metrik ini mungkin berkembang dari waktu ke waktu.

Sumber daya yang disediakan untuk mendukung beban kerja Anda mencakup sumber daya komputasi, penyimpanan, dan jaringan. Evaluasi sumber daya yang disediakan menggunakan metrik-metrik proksi Anda untuk melihat bagaimana sumber daya tersebut digunakan.

Gunakan metrik proksi Anda untuk mengukur sumber daya yang disediakan untuk mencapai hasil bisnis.

Sumber Daya	Contoh metrik proksi	Tujuan peningkatan
Komputasi	Menit vCPU	Memaksimalkan pemanfaatan sumber daya yang disediakan
Penyimpanan	GB yang disediakan	Mengurangi total yang disediakan
Jaringan	GB yang ditransfer atau paket yang ditransfer	Mengurangi total yang ditransfer dan jarak yang ditransfer

Metrik bisnis

Pilih metrik bisnis untuk menghitung pencapaian hasil bisnis. Metrik bisnis Anda harus mencerminkan nilai yang disediakan oleh beban kerja Anda, misalnya jumlah pengguna aktif dalam waktu yang sama, panggilan API yang dilayani, atau jumlah transaksi yang diselesaikan. Metrik-metrik ini mungkin berkembang dari waktu ke waktu. Hati-hatilah ketika mengevaluasi metrik bisnis berbasis keuangan, karena inkonsistensi pada nilai transaksi akan menjadikan perbandingan tidak valid.

Indikator kinerja utama

Menggunakan rumus berikut ini, bagi sumber daya yang disediakan dengan hasil bisnis yang dicapai untuk menentukan sumber daya yang disediakan per unit kerja.

$$\textit{Sumber daya yang disediakan per unit kerja} = \frac{\textit{Metrik proksi untuk sumber daya yang disediakan}}{\textit{Metrik bisnis untuk hasil}}$$

Rumus KPI

Gunakan sumber daya Anda per unit kerja sebagai KPI Anda. Buat garis acuan berdasarkan sumber daya yang disediakan sebagai dasar perbandingan.

Sumber Daya	Contoh KPI	Tujuan peningkatan
Komputasi	menit vCPU per transaksi	Memaksimalkan pemanfaatan sumber daya yang disediakan
Penyimpanan	GB per transaksi	Mengurangi total yang disediakan
Jaringan	GB yang ditransfer per transaksi atau paket yang ditransfer per transaksi	Mengurangi total yang ditransfer dan jarak yang ditransfer

Menghitung peningkatan

Hitung peningkatan sebagai penurunan kuantitatif pada sumber daya yang disediakan (seperti yang ditunjukkan oleh metrik proksi Anda) serta perubahan persentase dari sumber daya acuan Anda yang disediakan per unit kerja.

Sumber Daya	Contoh KPI	Tujuan peningkatan
Komputasi	% penurunan menit vCPU per transaksi	Maksimalkan pemanfaatan
Penyimpanan	% penurunan GB per transaksi	Mengurangi total yang disediakan
Jaringan	% penurunan GB yang ditransfer per transaksi atau paket yang ditransfer per transaksi	Mengurangi total yang ditransfer dan jarak yang ditransfer

Evaluasi peningkatan

Evaluasi potensi peningkatan berdasarkan manfaat bersih yang diharapkan. Evaluasi waktu, biaya, dan tingkat upaya untuk mengimplementasikan dan memelihara, serta risiko bisnis seperti dampak yang tidak terduga.

Peningkatan yang ditargetkan sering mewakili kompromi antar tipe sumber daya yang dipakai. Sebagai contoh, untuk mengurangi pemakaian komputasi, Anda dapat menyimpan hasil, atau untuk membatasi data yang ditransfer, Anda dapat memproses data sebelum mengirimkan hasilnya kepada klien. Kompromi-kompromi ini dibahas di detail tambahan yang akan datang.

Sertakan persyaratan nonfungsional ketika mengevaluasi risiko untuk beban kerja Anda, termasuk keamanan, keandalan, efisiensi kinerja, optimasi biaya, dan dampak peningkatan terhadap kemampuan Anda untuk mengoperasikan beban kerja.

Dengan menerapkan langkah ini ke Contoh skenario, Anda mengevaluasi target peningkatan dengan hasil-hasil berikut ini:

Praktik terbaik	Peningkatan yang ditargetkan	Potensi	Biaya	Risiko
Gunakan perangkat keras dalam jumlah minim untuk memenuhi kebutuhan Anda	Implementasikan penskalaan prediktif untuk mengurang i periode pemanfaatan yang rendah	Sedang	Rendah	Rendah
Gunakan teknologi yang mendukung pola akses dan penyimpanan data	Implementasikan mekanisme kompresi yang lebih efektif untuk mengurangi total penyimpanan dan waktu untuk mencapainya	Tinggi	Rendah	Rendah

Implementasi penjadwalan prediktif dapat mengurangi jam vCPU yang digunakan oleh instans dengan pemanfaatan rendah atau yang tidak digunakan yang menyediakan manfaat sedang dibandingkan mekanisme penskalaan yang ada dengan estimasi penurunan sumber daya yang digunakan sebesar 11%. Biaya yang terlibat jumlahnya rendah dan mencakup konfigurasi sumber

daya cloud dan operasi penskalaan prediktif untuk Amazon EC2 Auto Scaling. Risikonya adalah kinerja yang dibatasi ketika dilakukan penskalaan horizontal untuk merespons permintaan yang melampaui prediksi.

Implementasi kompresi yang lebih efektif akan memiliki dampak signifikan dengan penurunan yang besar dalam hal ukuran file di semua gambar asli dan manipulasi Anda, dengan estimasi penurunan kebutuhan penyimpanan sebesar 25% di lingkungan produksi. Implementasi algoritme baru adalah pengganti yang mudah dengan sedikit risiko yang terlibat.

Prioritaskan dan rencanakan peningkatan

Prioritaskan peningkatan yang Anda identifikasi berdasarkan dampak paling besar yang diantisipasi dengan biaya paling rendah serta risiko yang dapat diterima.

Putuskan peningkatan yang harus difokuskan di awal, dan sertakan peningkatan tersebut dalam perencanaan sumber daya dan roadmap pengembangan Anda.

Dengan menerapkan langkah ini ke Contoh skenario, Anda memprioritaskan target peningkatan sebagai berikut:

Prioritas	Peningkatan	Potensi	Biaya	Risiko
1	Implementasikan mekanisme kompresi yang lebih efektif	Tinggi	Rendah	Rendah
2	Implementasikan penskalaan prediktif	Sedang	Rendah	Rendah

Potensi yang tinggi dan biaya serta risiko yang rendah pada pembaruan kompresi file menjadikannya target bernilai tinggi untuk perusahaan Anda dan prioritas dibandingkan implementasi penskalaan prediktif. Anda menentukan bahwa implementasi penskalaan prediktif dengan potensi dampaknya yang sedang dan biaya serta risikonya yang rendah harus menjadi peningkatan prioritas setelah kompresi file selesai.

Anda menugaskan seorang anggota tim untuk mengimplementasikan kompresi file yang telah ditingkatkan dan menambahkan penskalaan prediktif ke backlog Anda.

Uji dan validasikan peningkatan

Lakukan pengujian kecil dengan investasi yang minim untuk mengurangi risiko upaya skala besar.

Implementasikan salinan beban kerja yang representatif dalam lingkungan pengujian Anda untuk membatasi biaya dan risiko dalam melakukan pengujian dan validasi. Lakukan rangkaian transaksi uji yang ditetapkan sebelumnya, ukur sumber daya yang disediakan, dan tentukan sumber daya yang digunakan per unit kerja untuk membuat garis acuan pengujian.

Implementasikan peningkatan target Anda di lingkungan pengujian dan ulangi pengujian menggunakan metodologi yang sama dengan kondisi yang sama. Lalu ukur sumber daya yang disediakan serta sumber daya yang digunakan per unit kerja dengan peningkatan yang telah diterapkan.

Hitung perubahan persentase dari garis acuan sumber daya yang disediakan per unit kerja, dan tentukan penurunan kuantitatif yang diharapkan pada sumber daya yang disediakan di lingkungan produksi Anda. Bandingkan nilai-nilai ini dengan nilai-nilai yang diperkirakan. Tentukan apakah hasilnya adalah peningkatan dengan level yang dapat diterima. Evaluasi apakah kompromi pada sumber daya tambahan yang dipakai menjadikan manfaat bersih dari peningkatan tersebut tidak dapat diterima.

Tentukan apakah peningkatan tersebut berhasil dan apakah sumber daya harus diinvestasikan dalam implementasi perubahan di lingkungan produksi. Jika setelah dievaluasi perubahan dianggap tidak berhasil, arahkan sumber daya Anda untuk menguji dan memvalidasi target berikutnya dan lanjutkan siklus peningkatan Anda.

% Penurunan sumber daya yang disediakan per unit kerja	Penurunan kuantitatif sumber daya yang disediakan	Tindakan
Memenuhi ekspektasi	Memenuhi ekspektasi	Lanjutkan peningkatan
Tidak memenuhi ekspektasi	Memenuhi ekspektasi	Lanjutkan peningkatan
Memenuhi ekspektasi	Tidak memenuhi ekspektasi	Cari peningkatan alternatif
Tidak memenuhi ekspektasi	Tidak memenuhi ekspektasi	Cari peningkatan alternatif

Dengan menerapkan langkah ini ke Contoh skenario, Anda melakukan pengujian untuk memvalidasi keberhasilan.

Setelah Anda melakukan pengujian pada algoritme kompresi yang ditingkatkan, penurunan persentase pada sumber daya yang disediakan per unit kerja (penyimpanan yang diperlukan untuk gambar asli dan gambar modifikasi) memenuhi ekspektasi dengan rata-rata penurunan 30% pada penyimpanan yang disediakan dan penambahan beban komputasi dalam jumlah sangat kecil.

Anda menentukan bahwa sumber daya komputasi tambahan yang diperlukan untuk menerapkan algoritme kompresi yang ditingkatkan ke file yang ada di lingkungan produksi tidaklah signifikan dibandingkan dengan penurunan penyimpanan yang dicapai. Anda mengonfirmasi keberhasilan penurunan kuantitatif pada sumber daya yang diperlukan (TB penyimpanan), dan peningkatan tersebut disetujui untuk deployment produksi.

Terapkan perubahan ke produksi

Implementasikan peningkatan yang telah diuji, divalidasi, dan disetujui ke produksi. Implementasikan menggunakan deployment terbatas, konfirmasikan fungsionalitas beban kerja Anda, uji penurunan riil pada sumber daya yang disediakan dan sumber daya yang digunakan per unit kerja di dalam deployment terbatas, dan periksa apakah ada konsekuensi yang tidak diinginkan dari perubahan tersebut. Lanjutkan deployment penuh setelah pengujian berhasil.

Batalkan perubahan jika pengujian gagal atau Anda mengalami konsekuensi yang tidak diinginkan dari perubahan Anda pada level yang tidak dapat diterima.

Dengan menerapkan langkah ini ke Contoh skenario, Anda mengambil tindakan-tindakan berikut ini.

Anda mengimplementasikan perubahan di produksi menggunakan deployment terbatas melalui metodologi deployment biru-hijau. Pengujian fungsionalitas pada instans yang baru di-deploy berhasil. Anda melihat penurunan rata-rata 26% pada penyimpanan yang disediakan untuk file gambar asli dan gambar manipulasi. Anda tidak melihat bukti kenaikan beban komputasi ketika mengompresi file-file baru.

Anda melihat penurunan tidak terduga pada waktu proses untuk mengompresi file gambar, dan Anda mengaitkan hal ini dengan kode yang sangat dioptimalkan untuk algoritme kompresi baru.

Anda melanjutkan deployment versi baru secara penuh.

Ukur hasil dan replikasi keberhasilan

Ukur hasil dan replikasi keberhasilan dengan cara-cara berikut ini:

- Ukur peningkatan awal pada sumber daya yang disediakan per unit kerja dan penurunan kuantitatif pada sumber daya yang disediakan.
- Bandingkan hitungan awal dan hasil pengujian dengan pengukuran produksi Anda. Identifikasi faktor-faktor yang mungkin menjadi penyebab selisih tersebut, dan perbarui metodologi hitungan dan pengujian Anda jika perlu.
- Tentukan keberhasilan, serta tingkat keberhasilan, dan bagikan hasilnya kepada pemangku kepentingan.
- Jika Anda harus membatalkan perubahan dikarenakan pengujian yang gagal atau konsekuensi negatif yang tidak diinginkan dari perubahan tersebut, identifikasi faktor-faktor penyebabnya. Lakukan iterasi apabila memungkinkan, atau evaluasi pendekatan baru untuk mencapai tujuan-tujuan perubahan.
- Ambil pelajaran yang Anda dapatkan, buat standar, dan terapkan peningkatan yang berhasil ke sistem-sistem lain yang bisa menerima manfaat yang serupa. Rekam dan bagikan metodologi Anda, artefak terkait, dan manfaat bersih kepada seluruh tim dan organisasi sehingga orang lain dapat mengadopsi standar Anda dan mengulang keberhasilan Anda.
- Pantau sumber daya yang disediakan per unit kerja dan lacak perubahan serta total dampak dari waktu ke waktu. Perubahan pada beban kerja Anda, atau cara pelanggan Anda memakai beban kerja Anda, dapat memengaruhi efektivitas peningkatan Anda. Evaluasi ulang peluang peningkatan jika Anda melihat penurunan jangka pendek yang besar pada efektivitas peningkatan Anda atau penurunan efektivitas yang terakumulasi dari waktu ke waktu.
- Hitung manfaat bersih dari peningkatan Anda dari waktu ke waktu (termasuk manfaat yang diterima oleh tim lain yang menerapkan peningkatan Anda, jika ada) untuk menunjukkan laba atas investasi dari aktivitas peningkatan Anda.

Dengan menerapkan langkah ini ke Contoh skenario, Anda mengukur hasil-hasil berikut ini.

Beban kerja Anda menunjukkan peningkatan awal berupa penurunan kebutuhan penyimpanan sebesar 23% setelah melakukan deployment dan menerapkan algoritme kompresi baru ke file-file gambar yang ada.

Nilai yang diukur sebagian besar sesuai dengan perkiraan awal (25%), dan selisih signifikan yang dibandingkan dengan pengujian (30%) ditentukan menjadi hasil file gambar yang digunakan dalam

pengujian tidak mewakili file gambar yang ada di produksi. Anda memodifikasi set gambar pengujian agar lebih mencerminkan gambar di produksi.

Peningkatan dianggap sepenuhnya berhasil. Total penurunan penyimpanan yang disediakan 2% lebih sedikit daripada yang diperkirakan yakni 25%, tetapi 23% tetaplah peningkatan besar dalam hal dampak pelestarian lingkungan, dan disertai dengan penghematan biaya yang setara.

Satu-satunya konsekuensi yang tidak diinginkan dari perubahan ini adalah penurunan waktu proses untuk melakukan kompresi dan penurunan setara pada vCPU yang dipakai. Semua peningkatan ini dianggap sebagai hasil dari kode yang sangat dioptimalkan.

Anda membuat sebuah proyek sumber terbuka internal di mana Anda membagikan kode, artefak terkait, panduan cara mengimplementasikan perubahan, dan hasil dari implementasi Anda. Proyek sumber terbuka internal ini memudahkan tim Anda dalam mengadopsi kode tersebut untuk semua kasus penggunaan penyimpanan file tetap mereka. Tim Anda mengadopsi peningkatan ini sebagai standar. Manfaat sekunder dari proyek sumber terbuka internal ini adalah setiap orang yang mengadopsi solusi tersebut mendapatkan manfaat peningkatan pada solusi, dan siapa pun dapat menyumbangkan peningkatan ke proyek ini.

Anda memublikasikan keberhasilan Anda dan membagikan proyek sumber terbuka Anda kepada seluruh organisasi. Setiap tim yang mengadopsi solusi ini mereplikasi manfaatnya dengan investasi yang minim dan menjadi tambahan pada manfaat bersih yang diterima dari investasi Anda. Anda memublikasikan data ini sebagai kisah keberhasilan yang berkelanjutan.

Anda melanjutkan pemantauan dampak peningkatan dari waktu ke waktu dan akan membuat perubahan pada proyek sumber terbuka internal Anda jika diperlukan.

Pelestarian lingkungan sebagai persyaratan nonfungsional

Penambahan pelestarian lingkungan ke daftar persyaratan bisnis dapat menghasilkan solusi-solusi yang lebih hemat biaya. Fokus pada pemerolehan lebih banyak nilai dari sumber daya yang Anda gunakan dan penggunaan sumber daya yang lebih sedikit secara langsung menghasilkan penghematan biaya di AWS karena Anda hanya membayar sesuai yang Anda gunakan.

Memenuhi target pelestarian lingkungan mungkin tidak memerlukan kompromi setara di satu atau beberapa metrik tradisional seperti waktu aktif, ketersediaan, atau waktu respons. Anda dapat mencapai hasil yang besar dalam hal pelestarian lingkungan tanpa ada dampak yang terukur pada tingkat layanan. Apabila kompromi kecil diperlukan, peningkatan pelestarian lingkungan yang didapatkan dari kompromi tersebut dapat mengungguli perubahan kualitas layanan.

Dorong anggota tim Anda untuk terus bereksperimen dengan peningkatan pelestarian lingkungan saat mereka mengembangkan persyaratan fungsional. Tim juga harus menyematkan metrik proksi saat menetapkan tujuan untuk memastikan bahwa mereka mengevaluasi intensitas sumber daya saat mengembangkan beban kerja.

Berikut ini adalah contoh kompromi yang dapat mengurangi sumber daya cloud yang Anda pakai:

Menyesuaikan kualitas hasil: Anda dapat mengorbankan Kualitas Hasil (QoR) demi pengurangan intensitas beban kerja dengan penghitungan perkiraan. Praktik penghitungan perkiraan mencari peluang pemanfaatan celah antara apa yang dibutuhkan pelanggan dan apa yang sebenarnya Anda buat. Misalnya, jika Anda menempatkan data Anda di sebuah set struktur data, Anda dapat menggunakan operator ORDER BY di SQL untuk menghilangkan pemrosesan yang tidak perlu, sehingga dapat menghemat sumber daya sambil tetap menyediakan jawaban yang dapat diterima.

Menyesuaikan waktu respons: Jawaban dengan waktu respons yang lebih lambat dapat mengurangi karbon dengan meminimalkan biaya tambahan bersama. Karena memproses secara khusus, tugas-tugas sementara dapat mendatangkan biaya tambahan di awal. Kelompokkan dan proses tugas-tugas dalam batch, bukan membayar biaya tambahan setiap kali tugas muncul. Pemrosesan batch mengorbankan waktu respons yang lebih cepat demi pengurangan biaya tambahan bersama untuk memunculkan instans, mengunduh kode sumber, dan menjalankan proses.

Menyesuaikan ketersediaan: Dengan AWS, Anda dapat menambahkan redundansi dan memenuhi target ketersediaan tinggi hanya dengan beberapa kali klik. Anda dapat meningkatkan redundansi melalui teknik seperti stabilitas statis dengan menyediakan sumber daya tidak aktif yang selalu menyebabkan pemanfaatan yang lebih rendah. Evaluasi kebutuhan bisnis saat menetapkan target.

Kompromi yang relatif kecil dalam hal ketersediaan dapat mendatangkan peningkatan yang jauh lebih besar dalam hal pemanfaatan. Misalnya, pola arsitektur stabilitas statis melibatkan pengadaan kapasitas failover tidak aktif agar dapat langsung mengambil beban setelah kesalahan komponen. Pelonggaran persyaratan ketersediaan dapat menghilangkan kebutuhan kapasitas daring tidak aktif dengan menyediakan waktu otomatisasi untuk melakukan deployment sumber daya pengganti. Penambahan kapasitas failover sesuai permintaan dapat mendorong pemanfaatan yang lebih tinggi secara keseluruhan tanpa mengganggu bisnis selama operasi normal dan memiliki manfaat sekunder berupa penurunan biaya.

Praktik terbaik untuk pelestarian lingkungan di cloud

Optimalkan penempatan beban kerja, dan optimalkan arsitektur untuk permintaan, perangkat lunak, data, perangkat keras, dan proses untuk meningkatkan efisiensi energi. Masing-masing area ini merepresentasikan peluang untuk menerapkan praktik terbaik guna mengurangi dampak beban kerja cloud Anda terhadap pelestarian lingkungan dengan memaksimalkan pemanfaatan, dan meminimalkan limbah serta total sumber daya yang di-deploy dan diberdayakan untuk mendukung beban kerja Anda.

Topik

Pemilihan wilayah

Pilihan Wilayah untuk beban kerja Anda sangat memengaruhi KPI-nya, termasuk kinerja, biaya, dan jejak karbon. Untuk meningkatkan semua KPI ini secara efektif, sebaiknya pilih Wilayah untuk beban kerja Anda berdasarkan persyaratan bisnis dan tujuan pelestarian lingkungan Anda.

Praktik terbaik

SUS01-BP01 Memilih Wilayah berdasarkan persyaratan bisnis dan tujuan keberlanjutan

Pilih Wilayah untuk beban kerja Anda berdasarkan persyaratan bisnis dan tujuan keberlanjutan Anda untuk mengoptimalkan KPI, termasuk kinerja, biaya, dan jejak karbon.

Antipola umum:

- Anda memilih Wilayah beban kerja berdasarkan lokasi Anda sendiri.
- Anda menggabungkan semua sumber daya beban kerja ke dalam satu lokasi geografis.

Manfaat menjalankan praktik terbaik ini: Menempatkan beban kerja dekat dengan proyek-proyek energi terbarukan Amazon atau Wilayah dengan intensitas karbon terpublikasi yang rendah dapat membantu menurunkan jejak karbon beban kerja cloud.

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

Panduan implementasi

AWS Cloud adalah jaringan Wilayah dan titik kehadiran (PoP) yang terus-menerus berekspansi, dengan infrastruktur jaringan global yang menghubungkan semuanya. Pilihan Wilayah untuk beban kerja Anda sangat memengaruhi KPI-nya, termasuk kinerja, biaya, dan jejak karbon. Untuk meningkatkan semua KPI ini secara efektif, Anda harus memilih Wilayah untuk beban kerja Anda berdasarkan persyaratan bisnis dan tujuan keberlanjutan Anda.

Langkah implementasi

- Ikuti langkah-langkah ini untuk mengevaluasi dan merangkum Wilayah potensial untuk beban kerja Anda berdasarkan persyaratan bisnis Anda termasuk kepatuhan, fitur yang tersedia, biaya, dan latensi:
  - Konfirmasikan bahwa Wilayah-Wilayah tersebut mematuhi persyaratan peraturan setempat.
  - Gunakan Daftar Layanan Wilayah AWS untuk memeriksa apakah Wilayah memiliki layanan dan fitur yang Anda perlukan untuk menjalankan beban kerja Anda.
  - Hitung biaya beban kerja pada setiap Wilayah menggunakan AWS Pricing Calculator.
  - Uji latensi jaringan antara lokasi pengguna akhir Anda dan setiap Wilayah AWS.
- Pilih Wilayah di dekat proyek-proyek energi terbarukan Amazon dan Wilayah dengan jaringan energi yang memiliki intensitas karbon terpublikasi yang lebih rendah daripada lokasi (atau Wilayah) lain.
  - Identifikasi pedoman keberlanjutan yang relevan untuk melacak dan membandingkan emisi karbon dari tahun ke tahun berdasarkan Protokol Gas Rumah Kaca (metode berbasis pasar dan berbasis lokasi).
  - Pilih wilayah berdasarkan metode yang Anda gunakan untuk melacak emisi karbon. Untuk detail selengkapnya tentang memilih Wilayah berdasarkan pedoman keberlanjutan, lihat Cara memilih Wilayah untuk beban kerja Anda berdasarkan tujuan keberlanjutan.

Sumber daya

Dokumen terkait:

- Memahami estimasi emisi karbon Anda
- Amazon di Seluruh Dunia
- Metodologi Energi Terbarukan
- Hal-Hal yang Perlu Dipertimbangkan saat Memilih Wilayah untuk Beban Kerja

Video terkait:

- Merancang secara berkelanjutan dan mengurangi jejak karbon AWS Anda

Penyelarasan dengan permintaan

Cara pengguna dan aplikasi menggunakan beban kerja Anda dan sumber daya lainnya dapat membantu Anda mengidentifikasi peningkatan untuk memenuhi tujuan pelestarian lingkungan. Skalakan infrastruktur agar dapat terus sesuai dengan permintaan dan verifikasi bahwa Anda hanya menggunakan sumber daya minimum yang diperlukan untuk mendukung pengguna Anda. Selaraskan tingkat layanan dengan kebutuhan pelanggan. Posisikan sumber daya guna membatasi jaringan yang diperlukan pengguna dan aplikasi untuk memakainya. Singkirkan aset yang tidak digunakan. Bekali anggota tim Anda dengan perangkat yang mendukung kebutuhan mereka dan meminimalkan dampak terhadap pelestarian lingkungan.

Praktik terbaik

- SUS02-BP01 Menskalakan infrastruktur beban kerja secara dinamis
- SUS02-BP02 Menyelaraskan SLA dengan tujuan keberlanjutan
- SUS02-BP03 Menghentikan pembuatan dan pemeliharaan aset yang tak terpakai
- SUS02-BP04 Mengoptimalkan penempatan geografis beban kerja berdasarkan persyaratan jaringannya
- SUS02-BP05 Mengoptimalkan sumber daya anggota tim untuk aktivitas yang dijalankan
- SUS02-BP06 Mengimplementasikan buffering atau throttling untuk meratakan kurva permintaan

SUS02-BP01 Menskalakan infrastruktur beban kerja secara dinamis

Gunakan elastisitas cloud dan skalakan infrastruktur Anda secara dinamis untuk menyesuaikan pasokan sumber daya cloud dengan permintaan dan menghindari kelebihan penyediaan kapasitas di beban kerja Anda.

Antipola umum:

- Anda tidak menskalakan infrastruktur Anda dengan beban pengguna.
- Anda secara manual menskalakan infrastruktur Anda sepanjang waktu.
- Anda membiarkan peningkatan kapasitas setelah peristiwa penskalaan, bukannya menurunkan kembali skala.

Manfaat menerapkan praktik terbaik ini: Mengonfigurasikan dan menguji elastisitas beban kerja akan membantu menyesuaikan pasokan sumber daya cloud dengan permintaan secara efisien dan menghindari kelebihan penyediaan kapasitas. Anda dapat memanfaatkan elastisitas di cloud untuk menskalakan kapasitas secara otomatis selama dan setelah lonjakan permintaan. Hal ini bertujuan untuk memastikan Anda hanya menggunakan jumlah sumber daya yang benar-benar diperlukan untuk memenuhi persyaratan bisnis Anda.

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

Panduan implementasi

Cloud menyediakan fleksibilitas untuk memperluas atau mengurangi sumber daya Anda secara dinamis melalui beragam mekanisme untuk memenuhi perubahan sesuai permintaan. Menyesuaikan pasokan dengan permintaan secara optimal akan memberikan dampak lingkungan terendah untuk beban kerja.

Permintaan dapat bersifat tetap atau bervariasi, sehingga akan memerlukan metrik dan otomatisasi untuk memastikan manajemen permintaan tersebut tidak menyulitkan. Aplikasi dapat diskalakan secara vertikal (naik atau turun) dengan mengubah ukuran instans, secara horizontal (ke dalam atau ke luar) dengan mengubah jumlah instans, atau kombinasi keduanya.

Anda dapat menggunakan sejumlah pendekatan yang berbeda untuk menyesuaikan pasokan sumber daya dengan permintaan.

- Pendekatan pelacakan target: Pantau metrik penskalaan Anda dan tingkatkan atau turunkan kapasitas secara otomatis sesuai kebutuhan.

- Penskalaan prediktif: Lakukan penskalaan dalam mengantisipasi tren harian dan mingguan.
- Pendekatan berbasis jadwal: Tetapkan jadwal penskalaan Anda sendiri sesuai dengan perubahan beban yang dapat diprediksi.
- Penskalaan layanan: Pilih layanan (seperti nirserver) yang diskalakan secara native berdasarkan desain atau sediakan penskalaan otomatis sebagai fitur.

Identifikasi periode penggunaan rendah atau nol dan skalakan sumber daya untuk menghapus kapasitas berlebih dan meningkatkan efisiensi.

Langkah implementasi

- Elastisitas menyesuaikan pasokan sumber daya yang Anda miliki dengan permintaan untuk sumber daya tersebut. Instans, kontainer, dan fungsi menyediakan mekanisme untuk elastisitas, baik dalam kombinasi dengan penskalaan otomatis maupun sebagai fitur layanan. AWS menyediakan serangkaian mekanisme penskalaan otomatis untuk memastikan beban kerja dapat diturunkan skalanya dengan cepat dan mudah selama periode beban pengguna yang rendah. Berikut beberapa contoh mekanisme penskalaan otomatis:

Auto scaling mechanism	Where to use
Amazon EC2 Auto Scaling	Gunakan untuk memastikan Anda memiliki jumlah instans Amazon EC2 yang tepat untuk menangani beban pengguna aplikasi Anda.
Application Auto Scaling	Gunakan untuk secara otomatis menskalakan sumber daya bagi layanan AWS masing-masing di luar Amazon EC2, seperti fungsi Lambda atau layanan Amazon Elastic Container Service (Amazon ECS).
Kubernetes Cluster Autoscaler	Gunakan untuk secara otomatis menskalakan klaster Kubernetes di AWS.

- Penskalaan sering dibahas terkait dengan layanan komputasi seperti instans Amazon EC2 atau fungsi AWS Lambda. Pertimbangkan konfigurasi layanan nonkomputasi seperti unit kapasitas baca dan tulis Amazon DynamoDB atau serpihan (shard) Amazon Kinesis Data Streams untuk disesuaikan dengan permintaan.

- Pastikan bahwa metrik untuk peningkatan atau penurunan skala telah divalidasi terhadap jenis beban kerja yang di-deploy. Jika Anda men-deploy aplikasi transkode video, 100% pemanfaatan CPU adalah hal normal dan tidak boleh menjadi metrik primer Anda. Anda dapat menggunakan metrik yang disesuaikan (seperti pemanfaatan memori) untuk kebijakan penskalaan jika diperlukan. Untuk memilih metrik yang tepat, pertimbangkan panduan berikut untuk Amazon EC2:
  - Metrik harus merupakan metrik pemanfaatan yang valid dan mendeskripsikan tingkat kesibukan suatu instans.
  - Nilai metrik harus meningkat atau menurun secara proporsional dengan jumlah instans dalam grup Auto Scaling.
- Gunakan penskalaan dinamis, bukan penskalaan manual untuk grup Auto Scaling Anda. Selain itu, sebaiknya gunakan kebijakan penskalaan pelacakan target dalam penskalaan dinamis Anda.
- Pastikan deployment beban kerja dapat menangani peristiwa penskalaan ke dalam dan ke luar. Buat skenario pengujian untuk peristiwa penskalaan ke dalam guna memastikan beban kerja berperilaku seperti yang diharapkan dan tidak memengaruhi pengalaman pengguna (seperti kehilangan sesi lekat (sticky session)). Anda dapat menggunakan Riwayat aktivitas untuk memverifikasi aktivitas penskalaan untuk grup Auto Scaling.
- Evaluasi beban kerja Anda untuk pola terprediksi dan secara proaktif skalakan saat Anda mengantisipasi perubahan terencana dan terprediksi dalam permintaan. Dengan penskalaan prediktif, Anda dapat meniadakan kebutuhan untuk menyediakan kapasitas secara berlebih. Untuk detail selengkapnya, lihat Penskalaan Prediktif dengan Amazon EC2 Auto Scaling.

Sumber daya

Dokumen terkait:

- Mulai Menggunakan Amazon EC2 Auto Scaling
- Penskalaan Prediktif untuk EC2, Didukung oleh Machine Learning
- Analisis perilaku pengguna menggunakan Amazon OpenSearch Service, Amazon Data Firehose, dan Kibana
- Apa yang dimaksud dengan Amazon CloudWatch?
- Memantau beban DB dengan Performance Insights di Amazon RDS
- Memperkenalkan Dukungan Native untuk Penskalaan Prediktif dengan Amazon EC2 Auto Scaling
- Memperkenalkan Karpenter - Kubernetes Cluster Autoscaler Sumber Terbuka yang Beperforma Tinggi

- Pendalaman tentang Amazon ECS Cluster Auto Scaling

Video terkait:

- Bangun lingkungan komputasi yang hemat biaya, energi, dan sumber daya
- Komputasi yang lebih baik, lebih cepat, dan lebih murah: Optimisasi biaya Amazon EC2 (CMP202-R1)

Contoh terkait:

- Lab: Contoh Grup Amazon EC2 Auto Scaling
- Lab: Memperkenalkan Penskalaan Otomatis dengan Karpenter

SUS02-BP02 Menyelaraskan SLA dengan tujuan keberlanjutan

Tinjau dan optimalkan perjanjian tingkat layanan (SLA) beban kerja berdasarkan tujuan keberlanjutan Anda untuk meminimalkan sumber daya yang diperlukan untuk mendukung beban kerja Anda sambil terus memenuhi kebutuhan bisnis.

Antipola umum:

- SLA beban kerja tidak diketahui atau ambigu.
- Anda menetapkan SLA hanya demi ketersediaan dan kinerja.
- Anda menggunakan pola desain yang sama (seperti arsitektur Multi-AZ) untuk semua beban kerja Anda.

Manfaat menjalankan praktik terbaik ini: Menyelaraskan SLA dengan tujuan keberlanjutan menghasilkan penggunaan sumber daya yang optimal sambil memenuhi kebutuhan bisnis.

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

Panduan implementasi

SLA menetapkan tingkat layanan yang diharapkan dari beban kerja cloud, seperti waktu respons, ketersediaan, dan retensi data. SLA memengaruhi arsitektur, penggunaan sumber daya, dan dampak lingkungan beban kerja cloud. Secara rutin, tinjau SLA dan buat pilihan kompromi yang secara

signifikan mengurangi penggunaan sumber daya dengan penurunan yang dapat diterima dalam hal tingkat layanan.

Langkah implementasi

- Tetapkan atau desain ulang SLA yang mendukung tujuan keberlanjutan Anda sambil memenuhi persyaratan bisnis Anda, bukan melampauinya.
- Ambil pilihan kompromi yang secara signifikan mengurangi dampak keberlanjutan dengan penurunan yang dapat diterima dalam hal tingkat layanan.
  - Keberlanjutan dan keandalan: Beban kerja dengan ketersediaan tinggi cenderung mengonsumsi lebih banyak sumber daya.
  - Keberlanjutan dan kinerja: Penggunaan lebih banyak sumber daya untuk meningkatkan kinerja dapat mendatangkan dampak lingkungan yang lebih tinggi.
  - Keberlanjutan dan keamanan: Beban kerja yang aman secara berlebihan dapat memiliki dampak lingkungan yang lebih tinggi.
- Gunakan pola desain seperti layanan mikro di AWS yang mengutamakan fungsi-fungsi yang krusial untuk bisnis, dan izinkan tingkat layanan (seperti waktu respons atau tujuan waktu pemulihan) yang lebih rendah untuk fungsi-fungsi nonkritis.

Sumber daya

Dokumen terkait:

- Perjanjian Tingkat Layanan (SLA) AWS
- Pentingnya Perjanjian Tingkat Layanan untuk Penyedia SaaS

Video terkait:

- Menghadirkan arsitektur pelestarian lingkungan dengan performa tinggi
- Bangun lingkungan komputasi yang hemat biaya, energi, dan sumber daya

SUS02-BP03 Menghentikan pembuatan dan pemeliharaan aset yang tak terpakai

Nonaktifkan aset yang tak terpakai di beban kerja Anda untuk mengurangi jumlah sumber daya cloud yang diperlukan untuk mendukung permintaan Anda dan meminimalkan limbah.

Antipola umum:

- Anda tidak menganalisis aplikasi Anda untuk mengetahui aset yang redundan atau tidak diperlukan lagi.
- Anda tidak menyingkirkan aset yang redundan atau tidak diperlukan lagi.

Manfaat menjalankan praktik terbaik ini: Menyingkirkan aset yang tak terpakai membebaskan sumber daya dan meningkatkan efisiensi beban kerja secara keseluruhan.

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

Panduan implementasi

Aset yang tak terpakai mengonsumsi sumber daya cloud seperti ruang penyimpanan dan daya komputasi. Dengan mengidentifikasi dan mengeliminasi aset-aset ini, Anda dapat membebaskan berbagai sumber daya ini, sehingga arsitektur cloud akan lebih efisien. Lakukan analisis aset aplikasi secara teratur, yakni aset seperti laporan pra-kompilasi, set data, gambar statis, dan pola akses aset untuk mengidentifikasi redundansi, pemanfaatan yang terlalu rendah, dan potensi target penonaktifan. Singkirkan aset redundan tersebut untuk mengurangi limbah sumber daya di beban kerja Anda.

Langkah implementasi

- Gunakan alat pemantauan untuk mengidentifikasi aset statis yang tidak diperlukan lagi.
- Sebelum menyingkirkan aset apa pun, evaluasi dampak penyingkirannya di arsitektur.
- Kembangkan rencana dan singkirkan aset yang tidak diperlukan lagi.
- Gabungkan aset tumpang tindih yang dihasilkan untuk menghindari redundansi pemrosesan.
- Perbarui aplikasi Anda hingga tidak lagi membuat dan menyimpan aset yang tidak diperlukan.
- Arahkan pihak ketiga untuk berhenti memproduksi dan menyimpan aset yang dikelola atas nama Anda yang tidak diperlukan lagi.
- Arahkan pihak ketiga untuk menggabungkan aset redundan yang dibuat atas nama Anda.
- Tinjau secara teratur beban kerja Anda untuk mengidentifikasi dan menyingkirkan aset yang tak terpakai.

Sumber daya

Dokumen terkait:

- Mengoptimalkan Infrastruktur AWS Anda untuk Pelestarian Lingkungan, Bagian II: Penyimpanan
- Bagaimana cara menghentikan sumber daya aktif yang tidak saya butuhkan lagi di Akun AWS saya?

Video terkait:

- Bagaimana cara memeriksa kemudian menyingkirkan sumber daya aktif yang tidak saya butuhkan lagi di Akun AWS saya?

SUS02-BP04 Mengoptimalkan penempatan geografis beban kerja berdasarkan persyaratan jaringannya

Pilih layanan dan lokasi cloud untuk beban kerja Anda yang mengurangi jarak yang harus ditempuh lalu lintas jaringan dan menurunkan total sumber daya jaringan yang diperlukan untuk mendukung beban kerja Anda.

Antipola umum:

- Anda memilih Wilayah beban kerja berdasarkan lokasi Anda sendiri.
- Anda menggabungkan semua sumber daya beban kerja ke dalam satu lokasi geografis.
- Semua lalu lintas mengalir melalui pusat data Anda.

Manfaat menjalankan praktik terbaik ini: Menempatkan beban kerja dekat dengan penggunanya akan menghasilkan latensi terendah sambil mengurangi pergerakan data di seluruh jaringan dan mengurangi dampak lingkungan.

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

Panduan implementasi

Infrastruktur AWS Cloud dibangun di seputar opsi lokasi seperti Wilayah, Zona Ketersediaan, grup penempatan, dan lokasi edge seperti AWS Outposts dan Zona Lokal AWS. Opsi lokasi ini bertanggung jawab untuk memelihara konektivitas antara komponen aplikasi, layanan cloud, jaringan edge, dan pusat data on-premise.

Analisis pola akses jaringan di beban kerja Anda untuk mengidentifikasi cara menggunakan opsi lokasi cloud ini dan mengurangi jarak yang harus ditempuh lalu lintas jaringan.

Langkah implementasi

- Analisis pola akses jaringan di beban kerja Anda untuk mengidentifikasi cara pengguna menggunakan aplikasi Anda.
  - Gunakan alat pemantauan, seperti Amazon CloudWatch dan AWS CloudTrail, untuk mengumpulkan data tentang aktivitas jaringan.
  - Analisis data untuk mengidentifikasi pola akses jaringan.
- Pilih Wilayah untuk deployment beban kerja Anda berdasarkan elemen utama berikut:
  - Tujuan Pelestarian Lingkungan Anda: seperti dijelaskan dalam Pemilihan wilayah.
  - Lokasi data Anda: Untuk aplikasi dengan banyak data (seperti big data dan machine learning), kode aplikasi harus dijalankan sedekat mungkin dengan data.
  - Lokasi pengguna Anda: Untuk aplikasi yang ditampilkan kepada pengguna, pilih Wilayah yang dekat dengan pengguna beban kerja Anda.
  - Kendala lainnya: Pertimbangkan kendala seperti biaya dan kepatuhan sebagaimana dijelaskan dalam Hal-Hal yang Perlu Dipertimbangkan saat Memilih Wilayah untuk Beban Kerja.
- Gunakan caching lokal atau Solusi Caching AWS untuk aset yang sering digunakan guna meningkatkan performa, mengurangi pergerakan data, dan mengurangi dampak lingkungan.

Layanan	Kapan harus digunakan
Amazon CloudFront	Gunakan untuk meng-cache konten statis seperti gambar, skrip, dan video, serta konten dinamis seperti respons API atau aplikasi web.
Amazon ElastiCache	Gunakan untuk meng-cache konten bagi aplikasi web.
DynamoDB Accelerator	Gunakan untuk menambahkan percepatan dalam memori ke tabel DynamoDB Anda.

- Gunakan layanan yang dapat membantu Anda menjalankan kode lebih dekat dengan pengguna beban kerja Anda:

Layanan	Kapan harus digunakan
Lambda@Edge	Gunakan untuk operasi dengan banyak komputasi yang dimulai saat objek tidak ada dalam cache.
Fungsi Amazon CloudFront	Gunakan untuk kasus penggunaan sederhana seperti permintaan HTTP atau manipulasi respons yang dapat dimulai oleh fungsi dengan masa pakai singkat.
AWS IoT Greengrass	Gunakan untuk menjalankan komputasi lokal, olahpesan, dan caching data untuk perangkat yang terhubung.

- Gunakan pooling koneksi untuk mengizinkan penggunaan ulang koneksi dan mengurangi sumber daya yang diperlukan.
- Gunakan penyimpanan data terdistribusi yang tidak mengandalkan koneksi persisten dan pembaruan sinkron untuk mendapatkan konsistensi guna melayani populasi wilayah.
- Ganti kapasitas jaringan statis yang disediakan di awal dengan kapasitas dinamis bersama, dan bagikan dampak pelestarian lingkungan kapasitas jaringan kepada pelanggan lain.

Sumber daya

Dokumen terkait:

- Mengoptimalkan Infrastruktur AWS untuk Pelestarian Lingkungan, Bagian III: Jaringan
- Dokumentasi Amazon ElastiCache
- Apa itu Amazon CloudFront?
- Fitur Utama Amazon CloudFront

Video terkait:

- Menjelaskan transfer data di AWS
- Menskalakan kinerja jaringan pada instans Amazon EC2 generasi berikutnya

Contoh terkait:

- Lokakarya Jaringan AWS
- Arsitektur untuk keberlanjutan - Meminimalkan pergerakan data lintas jaringan

SUS02-BP05 Mengoptimalkan sumber daya anggota tim untuk aktivitas yang dijalankan

Optimalkan sumber daya yang disediakan bagi anggota tim untuk meminimalkan dampak pelestarian lingkungan sambil mendukung kebutuhan mereka.

Antipola umum:

- Anda mengabaikan dampak dari perangkat yang digunakan oleh anggota tim Anda pada efisiensi aplikasi cloud secara keseluruhan.
- Anda secara manual mengelola dan memperbarui sumber daya yang digunakan oleh anggota tim.

Manfaat menjalankan praktik terbaik ini: Mengoptimalkan sumber daya anggota tim meningkatkan efisiensi keseluruhan aplikasi yang diaktifkan oleh cloud.

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

Panduan implementasi

Pahami sumber daya yang digunakan anggota tim Anda untuk mengonsumsi layanan Anda, ekspektasi siklus hidup mereka, dan dampak finansial serta dampak pada pelestarian lingkungan. Implementasikan strategi untuk mengoptimalkan berbagai sumber daya ini. Sebagai contoh, lakukan operasi yang kompleks, seperti rendering dan kompilasi, pada infrastruktur yang dapat diskalakan dan sangat banyak digunakan, bukan di sistem pengguna tunggal berdaya tinggi namun jarang digunakan.

Langkah implementasi

- Sediakan workstation dan perangkat lainnya untuk menyesuaikan dengan cara penggunaannya.
- Gunakan streaming aplikasi dan desktop virtual untuk membatasi persyaratan perangkat dan pemutakhiran.
- Alihkan tugas yang sarat dengan memori atau prosesor ke cloud untuk menggunakan elastisitasnya.

- Evaluasi dampak proses dan sistem atas siklus hidup perangkat, dan pilih solusi yang meminimalkan persyaratan untuk penggantian perangkat sekaligus memenuhi persyaratan bisnis.
- Implementasikan manajemen jarak jauh untuk perangkat guna mengurangi perjalanan bisnis yang diperlukan.
  - AWS Systems ManagerFleet Manager adalah pengalaman antarmuka (UI) pengguna terpadu yang membantu Anda mengelola simpul yang beroperasi di AWS atau on-premise dari jarak jauh.

Sumber daya

Dokumen terkait:

- Apa itu Amazon WorkSpaces?
- Pengoptimal Biaya untuk Amazon WorkSpaces
- Dokumentasi Amazon AppStream 2.0
- NICE DCV

Video terkait:

- Mengelola biaya untuk Amazon WorkSpaces di AWS

SUS02-BP06 Mengimplementasikan buffering atau throttling untuk meratakan kurva permintaan

Buffering dan throttling meratakan kurva permintaan dan mengurangi kapasitas tersedia yang diperlukan untuk beban kerja Anda.

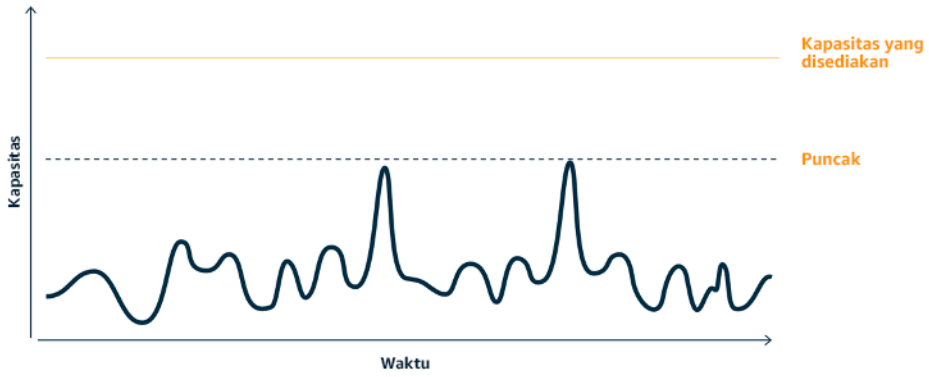
Antipola umum:

- Anda memproses permintaan klien dengan segera walaupun tidak diperlukan.
- Anda tidak menganalisis persyaratan untuk permintaan klien.

Manfaat menjalankan praktik terbaik ini: Meratakan kurva permintaan mengurangi kapasitas tersedia yang diperlukan untuk beban kerja. Mengurangi kapasitas tersedia artinya konsumsi energi berkurang dan dampak pada lingkungan juga berkurang.

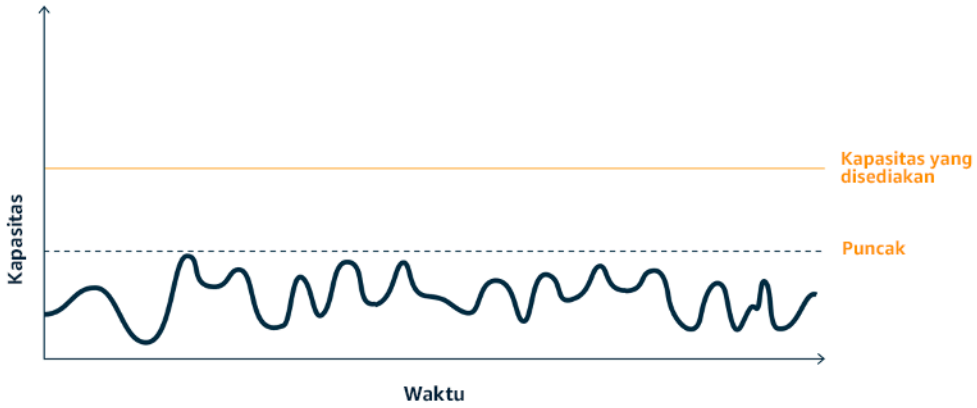
Tingkat risiko yang terjadi jika praktik terbaik ini tidak dijalankan: Rendah

Meratakan kurva permintaan beban kerja dapat membantu Anda mengurangi kapasitas tersedia untuk beban kerja dan mengurangi dampaknya pada lingkungan. Asumsikan beban kerja dengan kurva permintaan yang ditunjukkan pada gambar di bawah ini. Beban kerja ini memiliki dua puncak. Untuk menangani puncak-puncak ini, kapasitas sumber daya sebagaimana ditunjukkan oleh garis oranye disediakan. Sumber daya dan energi yang digunakan untuk beban kerja ini tidak diindikasikan oleh area di bawah kurva permintaan, tetapi oleh area di bawah garis kapasitas tersedia, karena kapasitas tersedia diperlukan untuk menangani kedua puncak ini.



Kurva permintaan dengan dua puncak terpisah yang memerlukan penyediaan kapasitas tinggi.

Anda dapat menggunakan buffering atau throttling untuk memodifikasi kurva permintaan dan meratakan puncak, yang berarti konsumsi lebih sedikit energi dan penyediaan kapasitas lebih rendah. Implementasikan throttling ketika klien Anda dapat mencoba ulang. Implementasikan buffering untuk menyimpan permintaan dan menunda pemrosesan ke lain waktu.



Efek throttling pada kurva permintaan dan kapasitas tersedia.

Langkah implementasi

- Analisis permintaan klien untuk menentukan cara merespons permintaan. Pertanyaan yang harus dipertimbangkan antara lain:
  - Dapatkah permintaan ini diproses secara asinkron?
  - Apakah klien memiliki kemampuan untuk mencoba ulang?
- Jika klien memiliki kemampuan untuk coba ulang, maka Anda dapat mengimplementasikan throttling, yang memberi tahu sumber bahwa jika sumber tidak dapat melayani permintaan pada saat ini maka sumber harus mencoba lagi nanti.
  - Anda dapat menggunakan Amazon API Gateway untuk mengimplementasikan throttling.
- Untuk klien yang tidak dapat mencoba ulang, buffer harus diimplementasikan untuk meratakan kurva permintaan. Buffer menunda pemrosesan permintaan, sehingga aplikasi yang dijalankan pada tingkat yang berlainan dapat berkomunikasi secara efektif. Pendekatan berbasis buffer menggunakan antrean atau aliran untuk menerima pesan dari produsen. Pesan dibaca oleh konsumen dan diproses, sehingga pesan dapat dijalankan dengan tingkat yang memenuhi persyaratan bisnis konsumen.
  - Amazon Simple Queue Service (Amazon SQS) merupakan layanan terkelola yang memberikan antrean yang memungkinkan satu konsumen membaca pesan secara individu.
  - Amazon Kinesis memberikan aliran yang memungkinkan banyak konsumen membaca pesan yang sama.
- Analisis permintaan secara keseluruhan, tingkat perubahan, dan waktu respons yang diperlukan untuk ukuran throttle atau buffer yang tepat.

Sumber daya

Dokumen terkait:

- Mulai menggunakan Amazon SQS
- Integrasi Aplikasi Menggunakan Antrean dan Pesan

Video terkait:

- Memilih Layanan Olah Pesan yang Tepat untuk Aplikasi Terdistribusi Anda

Perangkat lunak dan arsitektur

Implementasikan pola untuk melancarkan beban dan mempertahankan penggunaan yang tinggi dan konsisten atas sumber daya yang di-deploy guna meminimalkan sumber daya yang dipakai. Komponen dapat menjadi tidak aktif akibat kurangnya pemakaian, karena adanya perubahan perilaku pengguna dari waktu ke waktu. Revisi pola dan arsitektur untuk menggabungkan komponen dengan pemanfaatan rendah guna meningkatkan pemanfaatan secara keseluruhan. Pensiunkan komponen yang tidak lagi diperlukan. Pahami kinerja komponen beban kerja Anda, dan optimalkan komponen yang memakai sumber daya terbanyak. Ketahui perangkat yang digunakan pelanggan untuk mengakses layanan Anda, dan implementasikan pola untuk meminimalkan kebutuhan pemutakhiran perangkat.

Praktik terbaik

- SUS03-BP01 Optimalkan perangkat lunak dan arsitektur untuk tugas yang asinkron dan terjadwal
- SUS03-BP02 Menyingkirkan atau memfaktor ulang komponen beban kerja yang jarang atau tidak pernah digunakan
- SUS03-BP03 Mengoptimalkan area kode yang memakai waktu atau sumber daya paling banyak
- SUS03-BP04 Mengoptimalkan dampak pada perangkat dan perlengkapan
- SUS03-BP05 Menggunakan pola perangkat lunak dan arsitektur yang paling mendukung pola akses dan penyimpanan data

SUS03-BP01 Optimalkan perangkat lunak dan arsitektur untuk tugas yang asinkron dan terjadwal

Gunakan pola arsitektur dan perangkat lunak yang efisien seperti berbasis antrean untuk mempertahankan pemanfaatan sumber daya ter-deploy yang terus-menerus tinggi.

Antipola umum:

- Anda melakukan pengadaan sumber daya secara berlebihan di beban kerja cloud Anda untuk memenuhi lonjakan permintaan tidak terduga.
- Arsitektur Anda tidak memisahkan pengirim dan penerima pesan asinkron dengan komponen pesan.

Manfaat menjalankan praktik terbaik ini:

- Pola arsitektur dan perangkat lunak yang efisien meminimalkan sumber daya tidak terpakai di dalam beban kerja Anda dan meningkatkan keseluruhan efisiensi.
- Anda dapat menskalakan pemrosesan tanpa terikat penerimaan pesan asinkron.
- Melalui komponen pesan, Anda memiliki persyaratan ketersediaan yang longgar yang dapat Anda penuhi dengan sumber daya yang lebih sedikit.

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

Panduan implementasi

Gunakan pola arsitektur yang efisien seperti event-driven architecture (arsitektur berbasis peristiwa) yang menghasilkan bahkan pemanfaatan komponen dan meminimalkan pengadaan yang berlebihan dalam beban kerja Anda. Menggunakan pola arsitektur yang efisien meminimalkan sumber daya tidak aktif akibat kurangnya pemakaian yang disebabkan oleh perubahan permintaan seiring berjalannya waktu.

Pahami persyaratan komponen beban kerja Anda dan adopsi pola arsitektur yang meningkatkan keseluruhan pemanfaatan sumber daya. Pensiunkan komponen yang sudah tidak diperlukan.

Langkah implementasi

- Analisis permintaan untuk beban kerja Anda guna menentukan cara meresponsnya.
- Untuk permintaan atau tugas yang tidak memerlukan respons sinkron, gunakan arsitektur berbasis antrean dan pekerja penskalaan otomatis untuk memaksimalkan pemanfaatan. Berikut ini adalah beberapa contoh kapan Anda mungkin perlu mempertimbangkan arsitektur berbasis antrean:

Queuing mechanism	Description
Antrean tugas AWS Batch	Tugas AWS Batch dikirimkan ke antrean tugas dan menunggu untuk dijadwalkan berjalan di lingkungan komputasi.
Amazon Simple Queue Service dan Instans Spot Amazon EC2	Memasangkan Amazon SQS dan Instans Spot untuk membangun arsitektur yang efisien dan tahan terhadap kesalahan.

- Untuk permintaan atau tugas yang dapat diproses kapan saja, gunakan mekanisme penjadwalan untuk memproses tugas dalam batch untuk mendapatkan efisiensi yang lebih tinggi. Berikut beberapa contoh mekanisme penjadwalan di AWS:

Scheduling mechanism	Description
Penjadwal Amazon EventBridge	Sebuah kemampuan dari Amazon EventBridge yang memungkinkan Anda untuk membuat, menjalankan, dan mengelola tugas terjadwal pada skala besar.
Jadwal berbasis waktu AWS Glue	Tentukan jadwal berbasis waktu untuk perayap dan tugas Anda di AWS Glue.
Tugas terjadwal Amazon Elastic Container Service (Amazon ECS)	Amazon ECS mendukung pembuatan tugas terjadwal. Tugas terjadwal menggunakan aturan Amazon EventBridge untuk menjalankan tugas baik secara terjadwal atau dalam rangka merespons peristiwa EventBridge.
Penjadwal Instans	Konfigurasikan jadwal mulai dan berhenti untuk Instans Amazon EC2 dan Amazon Relational Database Service Anda.

- Jika Anda menggunakan mekanisme polling dan webhook dalam arsitektur Anda, gantilah dengan peristiwa. Gunakan arsitektur berbasis peristiwa untuk membangun beban kerja yang sangat efisien.
- Manfaatkan nirserver di AWS untuk menyingkirkan infrastruktur yang disediakan secara berlebihan.
- Sesuaikan ukuran setiap komponen arsitektur Anda untuk menghindari sumber daya yang tidak aktif karena menunggu input.

Sumber daya

Dokumen terkait:

- Apa itu Amazon Simple Queue Service?
- Apa itu Amazon MQ?

- Menskalakan berdasarkan Amazon SQS
- Apa itu AWS Step Functions?
- Apa itu AWS Lambda?
- Menggunakan AWS Lambda dengan Amazon SQS
- Apa itu Amazon EventBridge?

Video terkait:

- Beralih ke arsitektur berbasis peristiwa

SUS03-BP02 Menyingkirkan atau memfaktor ulang komponen beban kerja yang jarang atau tidak pernah digunakan

Singkirkan komponen yang tidak digunakan dan sudah tidak diperlukan, dan faktorkan ulang komponen dengan sedikit pemanfaatan, untuk meminimalkan limbah di beban kerja Anda.

Antipola umum:

- Anda tidak secara teratur memeriksa tingkat penggunaan masing-masing komponen beban kerja Anda.
- Anda tidak memeriksa dan menganalisis rekomendasi dari alat penyesuaian ukuran AWS seperti AWS Compute Optimizer.

Manfaat menjalankan praktik terbaik ini: Menyingkirkan komponen yang tidak digunakan akan meminimalkan limbah dan meningkatkan efisiensi beban kerja cloud secara keseluruhan.

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

Panduan implementasi

Tinjau beban kerja Anda untuk mengidentifikasi komponen yang pasif atau tidak digunakan. Tindakan ini adalah proses peningkatan yang berulang, yang dapat dipicu oleh perubahan dalam permintaan atau rilis layanan cloud baru. Contohnya, penurunan signifikan dalam waktu pelaksanaan fungsi AWS Lambda dapat menjadi indikator dibutuhkannya pengurangan ukuran memori. Selain itu, ketika AWS merilis layanan dan fitur baru, arsitektur dan layanan optimal untuk beban kerja Anda mungkin berubah.

Terus pantau aktivitas beban kerja Anda dan cari peluang untuk meningkatkan tingkat pemanfaatan masing-masing komponen. Dengan menyingkirkan komponen yang pasif dan melakukan aktivitas penyesuaian ukuran, Anda memenuhi persyaratan bisnis dengan sesedikit mungkin sumber daya cloud.

Langkah implementasi

- Pantau dan tangkap metrik pemanfaatan untuk komponen penting beban kerja Anda (seperti pemanfaatan CPU, pemanfaatan memori, atau throughput jaringan di metrik Amazon CloudWatch).
- Untuk beban kerja yang stabil, periksa AWS alat penyesuaian ukuran seperti AWS Compute Optimizer secara teratur untuk mengidentifikasi komponen yang pasif, tidak digunakan, atau kurang dimanfaatkan.
- Untuk beban kerja sementara, evaluasi metrik pemanfaatan untuk mengidentifikasi komponen yang pasif, tidak digunakan, atau kurang dimanfaatkan.
- Pensiunkan komponen dan aset terkait (seperti gambar Amazon ECR) yang tidak diperlukan lagi.
- Faktor ulang atau gabungkan komponen yang kurang dimanfaatkan dengan sumber daya lain untuk meningkatkan efisiensi pemanfaatan. Contohnya, Anda dapat menyediakan beberapa basis data kecil sebagai satu instans basis data Amazon RDS dan bukannya menjalankan basis data di instans individu yang kurang dimanfaatkan.
- Pahami sumber daya yang disediakan oleh beban kerja Anda untuk menyelesaikan unit kerja.

Sumber daya

Dokumen terkait:

- AWS Trusted Advisor
- Apa itu Amazon CloudWatch?
- Pembersihan Otomatis Gambar yang Tidak Digunakan di Amazon ECR

Contoh terkait:

- Well-Architected Lab - Penyesuaian Ukuran dengan AWS Compute Optimizer
- Well-Architected Lab - Optimisasi Pola Perangkat Keras dan Pengamatan KPI Pelestarian Lingkungan

SUS03-BP03 Mengoptimalkan area kode yang memakai waktu atau sumber daya paling banyak

Optimalkan kode Anda yang dijalankan di dalam berbagai macam komponen arsitektur Anda untuk meminimalkan penggunaan sumber daya sambil memaksimalkan performa.

Antipola umum:

- Anda mengabaikan optimisasi kode Anda untuk penggunaan sumber daya.
- Anda biasanya merespons masalah performa dengan meningkatkan sumber daya.
- Proses pengembangan dan peninjauan kode Anda tidak melacak perubahan performa.

Manfaat menjalankan praktik terbaik ini: Menggunakan kode yang efisien akan meminimalkan penggunaan sumber daya dan meningkatkan performa.

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

Panduan implementasi

Setiap area fungsional harus diperiksa, termasuk kode untuk aplikasi dengan arsitektur cloud, untuk mengoptimalkan penggunaan sumber dayanya dan performanya. Terus pantau performa beban kerja Anda di lingkungan pembangunan dan produksi dan identifikasi peluang untuk meningkatkan snippet kode yang memiliki penggunaan sumber daya sangat tinggi. Adopsi proses peninjauan secara teratur untuk mengidentifikasi bug atau antipola di dalam kode Anda yang menggunakan sumber daya secara tidak efisien. Manfaatkan algoritme sederhana dan efisien yang memberikan hasil yang sama untuk kasus penggunaan Anda.

Langkah implementasi

- Saat mengembangkan beban kerja Anda, adopsi proses peninjauan kode otomatis untuk meningkatkan kualitas dan mengidentifikasi bug dan antipola.
  - Otomatiskan peninjauan kode dengan Amazon CodeGuru Reviewer
  - Mendeteksi bug konkurensi dengan Amazon CodeGuru
  - Meningkatkan kualitas kode untuk aplikasi Python menggunakan Amazon CodeGuru
- Saat Anda menjalankan beban kerja Anda, pantau sumber daya untuk mengidentifikasi komponen dengan persyaratan sumber daya tinggi per unit kerja sebagai target untuk peninjauan kode.

- Untuk peninjauan kode, gunakan profiler kode untuk mengidentifikasi area kode yang menggunakan waktu atau sumber daya paling banyak sebagai target untuk dioptimalkan.
  - Mengurangi jejak karbon organisasi Anda dengan Amazon CodeGuru Profiler
  - Memahami penggunaan memori di aplikasi Java Anda dengan Amazon CodeGuru Profiler
  - Meningkatkan pengalaman pelanggan dan mengurangi biaya dengan Amazon CodeGuru Profiler
- Gunakan sistem operasi dan bahasa pemrograman paling efisien untuk beban kerja. Untuk informasi mendetail tentang bahasa pemrograman hemat energi (termasuk Rust), lihat Pelestarian lingkungan dengan Rust.
- Ganti algoritme yang banyak memerlukan komputasi dengan versi yang lebih sederhana dan lebih efisien, yang akan memberikan hasil yang sama.
- Singkirkan kode yang tidak perlu seperti penyortiran dan pemformatan.

Sumber daya

Dokumen terkait:

- Apa itu Amazon CodeGuru Profiler?
- Instans FPGA
- SDK AWS di Alat-Alat untuk Membangun di AWS

Video terkait:

- Tingkatkan Efisiensi Kode Menggunakan Amazon CodeGuru Profiler
- Otomatiskan Peninjauan Kode dan Rekomendasi Performa Aplikasi dengan Amazon CodeGuru

SUS03-BP04 Mengoptimalkan dampak pada perangkat dan perlengkapan

Pahami perangkat dan perlengkapan yang digunakan dalam arsitektur Anda dan gunakan strategi untuk mengurangi penggunaannya. Tindakan ini dapat meminimalkan dampak beban kerja cloud Anda pada lingkungan secara keseluruhan.

Antipola umum:

- Anda mengabaikan dampak dari perangkat yang digunakan oleh pelanggan Anda pada lingkungan.

- Anda secara manual mengelola dan memperbarui sumber daya yang digunakan oleh pelanggan.

Manfaat menjalankan praktik terbaik ini: Mengimplementasikan fitur dan pola perangkat lunak yang dioptimalkan untuk perangkat pelanggan dapat mengurangi dampak beban kerja cloud pada lingkungan secara keseluruhan.

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

Panduan implementasi

Mengimplementasikan fitur dan pola perangkat lunak yang dioptimalkan untuk perangkat pelanggan dapat mengurangi dampak pada lingkungan dengan beberapa cara:

- Mengimplementasikan fitur baru yang kompatibel dengan versi lama dapat mengurangi jumlah penggantian perangkat keras.
- Mengoptimalkan aplikasi untuk beroperasi secara efisien di perangkat dapat membantu mengurangi pemakaian energinya dan memperpanjang masa pakai baterainya (jika bertenaga baterai).
- Mengoptimalkan aplikasi untuk perangkat dapat juga mengurangi transfer data lewat jaringan.

Pahami perangkat dan perlengkapan yang digunakan dalam arsitektur Anda, ekspektasi siklus hidupnya, dan dampak dari penggantian komponen-komponen tersebut. Implementasikan fitur dan pola perangkat lunak yang dapat membantu meminimalkan pemakaian energi perangkat, keharusan pelanggan untuk mengganti perangkat dan juga melakukan pemutakhiran perangkat secara manual.

Langkah implementasi

- Inventarisasikan perangkat yang digunakan dalam arsitektur Anda. Perangkat dapat berupa perangkat seluler, tablet, perangkat IOT, lampu pintar, atau bahkan perangkat pintar dalam pabrik.
- Optimalkan aplikasi yang beroperasi pada perangkat:
  - Gunakan strategi seperti menjalankan tugas di latar belakang untuk mengurangi pemakaian energinya.
  - Perhitungkan bandwidth jaringan dan latensi saat membangun payload, dan implementasikan kemampuan yang membantu aplikasi bekerja dengan baik pada tautan yang memiliki bandwidth rendah dan latensi tinggi.
  - Ubah format payload dan file ke format optimal yang diperlukan oleh perangkat. Contohnya, Anda dapat menggunakan Amazon Elastic Transcoder atau AWS Elemental MediaConvert untuk

mengubah format file media digital besar dan berkualitas tinggi ke format yang dapat diputar pengguna di perangkat seluler, tablet, browser web, dan televisi yang terhubung.

- Lakukan aktivitas yang membutuhkan banyak komputasi di sisi server (seperti render gambar), atau gunakan streaming aplikasi untuk meningkatkan pengalaman pengguna pada perangkat yang lebih lama.
- Segmentasikan dan beri nomor halaman pada output, terutama untuk sesi interaktif, guna mengelola payload dan membatasi persyaratan penyimpanan lokal.

- Gunakan mekanisme otomatis lewat udara (OTA) untuk melakukan deployment pembaruan ke satu atau lebih perangkat.
  - Anda dapat menggunakan pipeline CI/CD untuk memperbarui aplikasi seluler.
  - Anda dapat menggunakan AWS IoT Device Management untuk mengelola perangkat terhubung dalam skala besar dari jarak jauh.
- Untuk menguji fitur baru dan pembaruan, gunakan device farm terkelola dengan set perangkat keras representatif dan ulang pengembangan untuk memaksimalkan perangkat yang didukung. Untuk detail selengkapnya, lihat SUS06-BP04 Menggunakan device farm terkelola untuk pengujian.

Sumber daya

Dokumen terkait:

- Apa itu AWS Device Farm?
- Dokumentasi Amazon AppStream 2.0
- NICE DCV
- Tutorial OTA untuk memperbarui firmware di perangkat yang menjalankan FreeRTOS

Video terkait:

- Pengantar AWS Device Farm

SUS03-BP05 Menggunakan pola perangkat lunak dan arsitektur yang paling mendukung pola akses dan penyimpanan data

Pahami bagaimana data digunakan di dalam beban kerja Anda, dipakai oleh pengguna Anda, ditransfer, dan disimpan. Gunakan pola perangkat lunak dan arsitektur yang paling mendukung akses dan penyimpanan data untuk meminimalkan sumber daya komputasi, jaringan, dan penyimpanan yang diperlukan untuk mendukung beban kerja.

Antipola umum:

- Anda berasumsi bahwa semua beban kerja memiliki pola penyimpanan dan akses data yang serupa.
- Anda hanya menggunakan satu tingkat penyimpanan, dengan anggapan semua beban kerja masuk dalam tingkat tersebut.
- Anda berasumsi bahwa pola akses data tidak akan berubah.
- Arsitektur Anda mendukung potensi lonjakan akses data yang tinggi, yang mengakibatkan sumber daya tetap tidak aktif dalam sebagian besar waktu.

Manfaat menjalankan praktik terbaik ini: Memilih dan mengoptimalkan arsitektur Anda berdasarkan akses data dan pola penyimpanan akan membantu mengurangi kompleksitas pengembangan dan meningkatkan pemanfaatan secara keseluruhan. Memahami kapan harus menggunakan tabel global, partisi data, dan caching akan membantu Anda mengurangi biaya operasional dan menskalakan sesuai kebutuhan beban kerja Anda.

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

Panduan implementasi

Gunakan pola arsitektur dan perangkat lunak yang paling sesuai dengan karakteristik data dan pola akses Anda. Contohnya, gunakan arsitektur data modern di AWS yang memungkinkan Anda menggunakan layanan yang dibuat khusus dan dioptimalkan untuk kasus penggunaan analitik unik Anda. Pola arsitektur ini memungkinkan pemrosesan data yang efisien dan mengurangi penggunaan sumber daya.

Langkah implementasi

- Analisis karakteristik data dan pola akses Anda untuk mengidentifikasi konfigurasi yang benar untuk sumber daya cloud Anda. Karakteristik utama yang perlu dipertimbangkan antara lain:

  - Jenis data: terstruktur, semi-terstruktur, tidak terstruktur
  - Pertumbuhan data: dibatasi, tidak dibatasi
  - Ketahanan data: persisten, sementara, transien
  - Pola akses: baca atau tulis, frekuensi pembaruan, berfluktuasi, atau konsisten
- Gunakan pola arsitektur yang paling mendukung pola akses dan penyimpanan data.
  - Mari Merancang! Arsitektur data modern
  - Basis data di AWS: Alat yang Tepat untuk Tugas yang Tepat
- Gunakan teknologi yang berfungsi secara native dengan data terkompresi.
- Gunakan layanan analitik yang dibuat khusus untuk pemrosesan data di arsitektur Anda.
- Gunakan mesin basis data yang paling mendukung pola kueri dominan Anda. Kelola indeks basis data Anda untuk memastikan pelaksanaan kueri yang efisien. Untuk detail selengkapnya, lihat Basis Data AWS.
- Pilih protokol jaringan yang mengurangi jumlah kapasitas jaringan yang dipakai di arsitektur Anda.

Sumber daya

Dokumen terkait:

- Format file Dukungan Kompresi Athena
- MENYALIN dari format data kolom dengan Amazon Redshift
- Mengubah Format Catatan Input Anda di Firehose
- Opsi Format untuk Input dan Output ETL di AWS Glue
- Meningkatkan performa kueri di Amazon Athena dengan Mengubah ke Format Kolom
- Memuat file data terkompresi dari Amazon S3 dengan Amazon Redshift
- Memantau beban DB dengan Wawasan Performa di Amazon Aurora
- Memantau beban DB dengan Wawasan Performa di Amazon RDS
- Kelas penyimpanan Intelligent-Tiering Amazon S3

Video terkait:

- Membangun arsitektur data modern di AWS

Manajemen data

Implementasikan praktik manajemen data untuk mengurangi penyimpanan yang diberikan dan diperlukan untuk mendukung beban kerja Anda, serta sumber daya yang diperlukan untuk menggunakannya. Pahami data Anda, dan gunakan konfigurasi dan teknologi penyimpanan penggunaan yang paling tepat untuk mendukung nilai bisnis data dan cara data digunakan. Buat siklus hidup data di penyimpanan yang lebih efisien dan memiliki kinerja lebih rendah ketika persyaratan berkurang, dan hapus data yang tidak lagi diperlukan.

Praktik terbaik

- SUS04-BP01 Mengimplementasikan kebijakan klasifikasi data
- SUS04-BP02 Menggunakan teknologi yang mendukung pola akses dan penyimpanan data
- SUS04-BP03 Menggunakan kebijakan untuk mengelola siklus hidup set data Anda
- SUS04-BP04 Menggunakan elastisitas dan otomatisasi untuk memperluas sistem file atau penyimpanan blok
- SUS04-BP05 Menyingkirkan data yang tidak diperlukan atau redundan
- SUS04-BP06 Menggunakan sistem file atau penyimpanan bersama untuk mengakses data umum
- SUS04-BP07 Meminimalkan perpindahan data di jaringan
- SUS04-BP08 Hanya mencadangkan data saat sulit untuk dibuat ulang

SUS04-BP01 Mengimplementasikan kebijakan klasifikasi data

Kelompokkan data untuk memahami tingkat kekritisannya terhadap hasil bisnis dan pilih tingkat penyimpanan hemat energi yang tepat untuk menyimpan data.

Antipola umum:

- Anda tidak mengidentifikasi aset data dengan karakteristik serupa (seperti sensitivitas, kekritisan bisnis, atau persyaratan peraturan) yang diproses atau disimpan.
- Anda belum mengimplementasikan katalog data untuk menginventarisasi aset data Anda.

Manfaat menjalankan praktik terbaik ini: Dengan mengimplementasikan kebijakan klasifikasi data, Anda dapat menentukan tingkat penyimpanan paling hemat energi untuk data.

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

Panduan implementasi

Klasifikasi data melibatkan identifikasi jenis-jenis data yang sedang diproses dan disimpan di sistem informasi yang dimiliki atau dioperasikan oleh organisasi. Klasifikasi data juga melibatkan penentuan tingkat kekritisan data dan kemungkinan dampak penyusupan, kehilangan, atau penyalahgunaan data.

Implementasikan kebijakan klasifikasi data dengan bekerja mundur dari penggunaan data kontekstual dan membuat skema kategorisasi yang mempertimbangkan tingkat kekritisan set data tertentu terhadap operasi organisasi.

Langkah implementasi

- Lakukan inventaris berbagai jenis data yang ada untuk beban kerja Anda.
  - Untuk detail selengkapnya tentang kategori data, lihat Laporan Resmi Klasifikasi Data.
- Tentukan kekritisan, kerahasiaan, integritas, dan ketersediaan data berdasarkan risiko terhadap organisasi. Gunakan persyaratan ini untuk mengelompokkan data ke dalam satu tingkat klasifikasi data yang Anda adopsi.
  - Sebagai contoh, lihat Empat langkah sederhana untuk mengklasifikasikan data Anda dan mengamankan startup Anda.
- Audit lingkungan Anda secara berkala untuk data yang tidak ditandai dan tidak diklasifikasikan, serta klasifikasikan dan tandai data dengan tepat.
  - Sebagai contoh, lihat Katalog Data dan perayap di AWS Glue.
- Bangun katalog data yang menyediakan kemampuan audit dan tata kelola.
- Tentukan dan dokumentasikan prosedur penanganan untuk setiap kelas data.
- Gunakan otomatisasi untuk mengaudit lingkungan Anda secara kontinu guna mengidentifikasi data yang tidak ditandai dan tidak diklasifikasikan, serta klasifikasikan dan tandai data dengan tepat.

Sumber daya

Dokumen terkait:

- Memanfaatkan AWS Cloud untuk Mendukung Klasifikasi Data
- Kebijakan tag dari AWS Organizations

Video terkait:

- Mewujudkan ketangkasan dengan tata kelola data di AWS

SUS04-BP02 Menggunakan teknologi yang mendukung pola akses dan penyimpanan data

Gunakan teknologi penyimpanan yang paling mendukung cara data Anda diakses dan disimpan untuk meminimalkan sumber daya yang disediakan sambil mendukung beban kerja Anda.

Antipola umum:

- Anda berasumsi bahwa semua beban kerja memiliki pola penyimpanan dan akses data yang serupa.
- Anda hanya menggunakan satu tingkat penyimpanan, dengan anggapan semua beban kerja masuk dalam tingkat tersebut.
- Anda berasumsi bahwa pola akses data tidak akan berubah.

Manfaat menjalankan praktik terbaik ini: Memilih dan mengoptimalkan teknologi penyimpanan Anda berdasarkan pola akses dan penyimpanan data akan membantu Anda mengurangi sumber daya cloud yang diperlukan untuk memenuhi kebutuhan bisnis dan meningkatkan keseluruhan efisiensi beban kerja cloud.

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

Panduan implementasi

Pilih solusi penyimpanan yang paling sesuai dengan pola akses Anda, atau pertimbangkan untuk mengubah pola akses untuk menyesuaikan dengan solusi penyimpanan guna memaksimalkan efisiensi kinerja.

- Evaluasi karakteristik dan pola akses data Anda untuk mengumpulkan karakteristik utama kebutuhan penyimpanan Anda. Karakteristik utama yang perlu dipertimbangkan antara lain:
  - Tipe data: terstruktur, semi-terstruktur, tidak terstruktur
  - Pertumbuhan data: dibatasi, tidak dibatasi
  - Daya tahan data: persisten, sementara, transien
  - Pola akses: baca atau tulis, frekuensi, berfluktuasi, atau konsisten

- Migrasikan data ke teknologi penyimpanan yang tepat yang mendukung karakteristik dan pola akses data Anda. Berikut adalah beberapa contoh teknologi penyimpanan AWS serta karakteristik utamanya:

Tipe	Teknologi	Karakteristik utama
Penyimpanan objek	Amazon S3	Layanan penyimpanan objek dengan skalabilitas tak terbatas, ketersediaan tinggi, dan berbagai opsi aksesibilitas. Mentransfer dan mengakses objek masuk dan keluar dari Amazon S3 dapat menggunakan layanan, seperti Transfer Acceleration atau Access Points, untuk mendukung lokasi, kebutuhan keamanan, dan pola akses Anda.
Penyimpanan pengarsipan	Amazon S3 Glacier	Kelas penyimpanan Amazon S3 yang dibuat untuk pengarsipan data.
Sistem file bersama	Amazon Elastic File System (Amazon EFS)	Sistem file mountable yang dapat diakses oleh berbagai jenis solusi komputasi . Amazon EFS secara otomatis memperbesar dan memperkecil penyimpanan serta dioptimalkan performan ya agar memberikan latensi rendah yang konsisten.

Tipe	Teknologi	Karakteristik utama
Sistem file bersama	Amazon FSx	Dibangun berdasarkan solusi komputasi AWS terbaru untuk mendukung empat sistem file yang umum digunakan: NetApp ONTAP, OpenZFS, Windows File Server, dan Lustre. Amazon FSx memiliki latensi, throughput, dan IOPS yang bervariasi per sistem file dan hal ini harus dipertimbangkan saat memilih sistem file yang tepat untuk kebutuhan beban kerja Anda.
Penyimpanan blok	Amazon Elastic Block Store (Amazon EBS)	Layanan penyimpanan blok kinerja tinggi yang dapat diskalakan yang dirancang untuk Amazon Elastic Compute Cloud (Amazon EC2). Amazon EBS mencakup penyimpanan yang didukung SSD untuk beban kerja transaksional intensif IOPS dan penyimpanan yang didukung HDD untuk beban kerja intensif throughput.

Tipe	Teknologi	Karakteristik utama
Basis data relasional	Amazon Aurora, Amazon RDS, Amazon Redshift	Didesain untuk mendukung transaksi ACID (atomisit as, konsistensi, isolasi, durabilitas), dan mempertah ankan integritas referensi al serta konsistensi data yang tinggi. Banyak aplikasi tradisional, perencanaan sumber daya perusahaan (ERP), manajemen hubungan pelanggan (CRM), dan sistem perdagangan elektronik menggunakan basis data relasional untuk menyimpan data mereka.
Basis data nilai-kunci	Amazon DynamoDB	Dioptimalkan untuk pola akses umum, biasanya untuk menyimpan dan mengambil data dalam volume besar. Aplikasi web dengan lalu lintas tinggi, sistem perdagangan elektroni k, dan aplikasi gaming merupakan kasus penggunaa n umum untuk basis data nilai kunci.

- Untuk sistem penyimpanan dengan ukuran tetap, seperti Amazon EBS atau Amazon FSx, pantau ruang penyimpanan yang tersedia dan otomatiskan alokasi penyimpanan saat ambang batas tercapai. Anda dapat memanfaatkan Amazon CloudWatch untuk mengumpulkan dan menganalisis metrik yang berbeda-beda untuk Amazon EBS dan Amazon FSx.
- Kelas Penyimpanan Amazon S3 dapat dikonfigurasi pada level objek dan satu bucket dapat berisi objek yang disimpan di semua kelas penyimpanan.

- Anda juga dapat menggunakan kebijakan Siklus Hidup Amazon S3 untuk mengalihkan objek secara otomatis antar kelas-kelas penyimpanan atau menghapus data tanpa perubahan aplikasi apa pun. Secara umum, Anda harus memilih mana yang penting antara efisiensi sumber daya, latensi akses, dan keandalan saat mempertimbangkan semua mekanisme penyimpanan ini.

Sumber daya

Dokumen terkait:

- Jenis volume Amazon EBS
- Penyimpanan instans Amazon EC2
- Amazon S3 Intelligent-Tiering
- Karakteristik I/O Amazon EBS
- Menggunakan kelas penyimpanan Amazon S3
- Apa itu Amazon S3 Glacier?

Video terkait:

- Pola Arsitektur untuk Danau Data di AWS
- Pendalaman tentang Amazon EBS (STG303-R1)
- Optimalkan kinerja penyimpanan Anda dengan Amazon S3 (STG343)
- Membangun arsitektur data modern di AWS

Contoh terkait:

- Driver CSI Amazon EFS
- Driver CSI Amazon EBS
- Utilitas Amazon EFS
- Penskalaan Otomatis Amazon EBS
- Contoh Amazon S3

SUS04-BP03 Menggunakan kebijakan untuk mengelola siklus hidup set data Anda

Kelola siklus hidup semua data Anda dan terapkan penghapusan secara otomatis untuk meminimalkan total penyimpanan yang diperlukan untuk beban kerja Anda.

Antipola umum:

- Anda menghapus data secara manual.
- Anda tidak menghapus data beban kerja Anda sama sekali.
- Anda tidak mengalihkan data ke tingkat penyimpanan yang lebih hemat energi berdasarkan persyaratan retensi dan aksesnya.

Manfaat menjalankan praktik terbaik ini: Penggunaan kebijakan siklus hidup data memastikan akses dan retensi data yang efisien dalam suatu beban kerja.

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

Panduan implementasi

Set data biasanya memiliki persyaratan retensi dan akses data yang berbeda-beda selama masa hidupnya. Misalnya, aplikasi Anda mungkin memerlukan akses yang sering ke sejumlah set data dalam jangka waktu terbatas. Setelah itu, set-set data tersebut jarang diakses.

Untuk mengelola set data Anda secara efisien di sepanjang siklus hidupnya, konfigurasikan kebijakan siklus hidup, yakni aturan yang menetapkan cara menangani set data.

Dengan Aturan konfigurasi siklus hidup, Anda dapat meminta layanan penyimpanan tertentu untuk mengalihkan suatu set data ke tingkat penyimpanan yang lebih hemat energi, mengarsipkannya, atau menghapusnya.

Langkah implementasi

- Klasifikasikan set data dalam beban kerja Anda.
- Tetapkan prosedur penanganan untuk setiap kelas data.
- Atur kebijakan siklus hidup otomatis untuk menegakkan aturan siklus hidup. Berikut ini adalah beberapa cara menyiapkan kebijakan siklus hidup otomatis untuk berbagai layanan penyimpanan AWS:

Storage service	How to set automated lifecycle policies
Amazon S3	Anda dapat menggunakan Siklus Hidup Amazon S3 untuk mengelola objek-objek Anda di sepanjang siklus hidupnya. Jika pola akses Anda tidak diketahui, berubah-u bah, atau tidak dapat diprediksi, Anda dapat menggunakan Amazon S3 Intelligent-Tierin g, yang memantau pola akses dan secara otomatis memindahkan objek yang belum diakses ke tingkat akses dengan biaya lebih rendah. Anda dapat memanfaatkan metrik Lensa Penyimpanan Amazon S3 untuk mengidentifikasi peluang dan celah pengoptim alan dalam manajemen siklus hidup.
Amazon Elastic Block Store	Anda dapat menggunakan Amazon Data Lifecycle Manager untuk mengotomatiskan pembuatan, retensi, dan penghapusan snapshot Amazon EBS dan AMI yang dicadangkan Amazon EBS.
Amazon Elastic File System	Manajemen siklus hidup Amazon EFS secara otomatis mengelola penyimpanan file untuk sistem file Anda.
Amazon Elastic Container Registry	Kebijakan siklus hidup Amazon ECR mengotomatiskan pembersihan image kontainer Anda dengan mengakhiri masa berlaku image berdasarkan usia atau jumlah.
AWS Elemental MediaStore	Anda dapat menggunakan kebijakan siklus hidup objek yang mengatur seberapa lama objek harus disimpan di kontainer MediaStore.

- Hapus volume, snapshot, dan data yang tidak digunakan yang sudah melebihi masa retensinya. Manfaatkan fitur layanan native seperti Amazon DynamoDB Time To Live atau retensi log Amazon CloudWatch untuk penghapusan.
- Agregasikan dan kompresi data jika memungkinkan berdasarkan aturan siklus hidup.

Sumber daya

Dokumen terkait:

- Optimalkan aturan Siklus Hidup Amazon S3 Anda dengan Analisis Kelas Penyimpanan Amazon S3
- Mengevaluasi Sumber Daya dengan Aturan AWS Config

Video terkait:

- Sederhanakan Siklus Hidup Data Anda dan Optimalkan Biaya Penyimpanan dengan Siklus Hidup Amazon S3
- Kurangi Biaya Penyimpanan Anda Menggunakan Lensa Penyimpanan Amazon S3

SUS04-BP04 Menggunakan elastisitas dan otomatisasi untuk memperluas sistem file atau penyimpanan blok

Gunakan elastisitas dan otomatisasi untuk memperluas sistem file atau penyimpanan blok seiring pertumbuhan data untuk meminimalkan total penyimpanan yang disediakan.

Antipola umum:

- Anda membeli sistem file atau penyimpanan blok besar untuk keperluan di waktu mendatang.
- Anda memberikan persediaan berlebih untuk operasi input dan output per detik (IOPS) sistem file Anda.
- Anda tidak memantau pemanfaatan volume data Anda.

Manfaat menjalankan praktik terbaik ini: Meminimalkan penyediaan yang berlebihan untuk sistem penyimpanan mengurangi sumber daya yang tidak aktif dan meningkatkan efisiensi beban kerja Anda secara keseluruhan.

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

Panduan implementasi

Buat sistem file dan penyimpanan blok dengan alokasi ukuran, throughput, dan latensi yang sesuai untuk beban kerja Anda. Gunakan elastisitas dan otomatisasi untuk memperluas sistem file atau penyimpanan blok seiring pertumbuhan data untuk meminimalkan total penyimpanan yang disediakan.

Langkah implementasi

- Untuk penyimpanan berukuran tetap seperti Amazon EBS, pastikan Anda memantau kapasitas penyimpanan yang digunakan dibandingkan dengan keseluruhan ukuran penyimpanan, dan buat otomatisasi jika memungkinkan untuk meningkatkan ukuran penyimpanan saat mencapai ambang batas.
- Gunakan volume elastis dan layanan data blok terkelola untuk mengotomatisasi alokasi penyimpanan tambahan seiring tumbuhnya data persisten Anda. Contohnya, Anda dapat menggunakan Elastic Volumes Amazon EBS untuk mengubah ukuran volume, jenis volume, atau menyesuaikan performa volume Amazon EBS Anda.
- Pilih kelas penyimpanan, mode performa, dan mode throughput yang tepat untuk sistem file Anda guna memenuhi kebutuhan bisnis, tidak melebihinya.
  - Performa Amazon EFS
  - Performa volume Amazon EBS di instans Linux
- Atur target tingkat pemanfaatan untuk volume data Anda, dan ubah ukuran volume di luar rentang yang diperkirakan.
- Sesuaikan ukuran volume hanya-baca agar sesuai dengan data.
- Migrasikan data ke penyimpanan objek untuk menghindari penyediaan kapasitas yang berlebihan dari ukuran volume tetap di penyimpanan blok.
- Secara rutin tinjau volume elastis dan sistem file untuk menghentikan volume yang tidak aktif dan memperkecil sumber daya dengan penyediaan berlebihan agar sesuai dengan ukuran data saat ini.

Sumber daya

Dokumen terkait:

- Dokumentasi Amazon FSx
- Apa itu Amazon Elastic File System?

Video terkait:

- Memahami Elastic Volumes Amazon EBS
- Strategi Optimisasi Cuplikan dan Amazon EBS untuk Peningkatan Performa dan Penghematan Biaya
- Mengoptimalkan Amazon EFS untuk biaya dan performa, menggunakan praktik terbaik

SUS04-BP05 Menyingkirkan data yang tidak diperlukan atau redundan

Hapus data yang tidak diperlukan atau redundan untuk meminimalkan sumber daya penyimpanan yang diperlukan untuk menyimpan set data Anda.

Antipola umum:

- Anda menduplikasi data yang dapat diperoleh atau dibuat ulang dengan mudah
- Anda mencadangkan semua data tanpa mempertimbangkan tingkat kekritisannya.
- Anda menghapus data tidak rutin, hanya pada peristiwa operasional, atau tidak menghapusnya sama sekali.
- Anda menyimpan data secara redundan dengan mengabaikan durabilitas layanan penyimpanan.
- Anda mengaktifkan versioning Amazon S3 tanpa alasan bisnis apa pun.

Manfaat menjalankan praktik terbaik ini: Penghapusan data yang tidak diperlukan dapat mengurangi ukuran penyimpanan yang diperlukan untuk beban kerja Anda serta dampak beban kerja terhadap lingkungan.

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

Panduan implementasi

Jangan menyimpan data yang tidak Anda perlukan. Otomatiskan penghapusan data yang tidak diperlukan. Gunakan teknologi yang menghilangkan data ganda pada tingkat file dan blok. Manfaatkan fitur replikasi dan redundansi data native dari layanan.

Langkah implementasi

- Evaluasi apakah Anda dapat menghindari menyimpan data menggunakan set data yang saat ini tersedia untuk publik di AWS Data Exchange dan Data Terbuka di AWS.

- Gunakan mekanisme yang dapat membatalkan duplikasi data pada tingkat blok dan objek. Berikut ini adalah beberapa contoh cara membatalkan duplikasi data di AWS:

Storage service	Deduplication mechanism
Amazon S3	Gunakan AWS Lake Formation FindMatches untuk menemukan catatan pencocokan di sebuah set data (termasuk tanpa pengidentifikasi) menggunakan FindMatches ML Transform baru.
Amazon FSx	Aktifkan pembatalan duplikasi data di Amazon FSx untuk Windows.
Snapshot Amazon Elastic Block Store	Snapshot adalah cadangan bertahap, yang berarti penyimpanan hanya dilakukan untuk blok di perangkat yang telah berubah setelah snapshot terbaru Anda.

- Analisis akses data untuk mengidentifikasi data yang tidak diperlukan. Otomatiskan kebijakan siklus hidup. Manfaatkan fitur layanan native seperti Amazon DynamoDB Time To Live, Siklus Hidup Amazon S3, atau retensi log Amazon CloudWatch untuk penghapusan.
- Gunakan kemampuan virtualisasi data di AWS untuk mempertahankan data di sumbernya dan menghindari duplikasi data.
  - Virtualisasi Data Cloud Native di AWS
  - Lab: Mengoptimalkan Pola Data Menggunakan Fitur Berbagi Data Amazon Redshift
- Gunakan teknologi pencadangan yang dapat membuat cadangan bertahap.
- Manfaatkan durabilitas Amazon S3 dan replikasi Amazon EBS untuk memenuhi tujuan durabilitas Anda, bukan teknologi yang dikelola mandiri (seperti rangkaian disk independen yang redundan (RAID)).
- Pusatkan log dan lacak data, batalkan duplikasi entri log yang identik, dan buat mekanisme untuk menyesuaikan verbositas saat diperlukan.
- Pra-isi cache hanya saat ada alasan yang dibenarkan.
- Lakukan pemantauan dan otomatisasi cache untuk menyesuaikan ukuran cache dengan tepat.
- Singkirkan deployment dan aset usang dari penyimpanan objek dan cache edge saat mendorong versi baru untuk beban kerja Anda.

Sumber daya

Dokumen terkait:

- Ubah retensi data log di CloudWatch Logs
- Pembatalan duplikasi data di Amazon FSx untuk Windows File Server
- Fitur Amazon FSx untuk ONTAP termasuk pembatalan duplikasi data
- Membatalkan File di Amazon CloudFront
- Menggunakan AWS Backup untuk mencadangkan dan memulihkan sistem file Amazon EFS
- Apa yang dimaksud dengan Amazon CloudWatch Logs?
- Bekerja dengan cadangan di Amazon RDS

Video terkait:

- Pencocokan Fuzzy dan Pembatalan Duplikasi Data dengan ML Transforms untuk AWS Lake Formation

Contoh terkait:

- Bagaimana cara menganalisis log akses server Amazon S3 menggunakan Amazon Athena?

SUS04-BP06 Menggunakan sistem file atau penyimpanan bersama untuk mengakses data umum

Adopsi sistem file atau penyimpanan bersama untuk menghindari duplikasi data dan memungkinkan infrastruktur yang lebih efisien untuk beban kerja Anda.

Antipola umum:

- Anda menyediakan penyimpanan untuk setiap klien secara individu.
- Anda tidak melepaskan volume data dari klien yang tidak aktif.
- Anda tidak memberikan akses ke penyimpanan di semua platform dan sistem.

Manfaat menjalankan praktik terbaik ini: Menggunakan sistem file atau penyimpanan bersama memungkinkan pemberian data ke satu atau lebih pemakai tanpa harus menyalin data tersebut. Hal ini membantu mengurangi sumber daya penyimpanan yang diperlukan untuk beban kerja.

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

Panduan implementasi

Jika Anda memiliki beberapa pengguna atau aplikasi yang mengakses set data yang sama, penggunaan teknologi penyimpanan bersama sangatlah penting agar infrastruktur untuk beban kerja Anda efisien. Teknologi penyimpanan bersama memberikan lokasi sentral untuk menyimpan dan mengelola set data dan menghindari duplikasi data. Teknologi ini juga memastikan konsistensi data di berbagai sistem yang berlainan. Lebih lanjut, teknologi penyimpanan bersama memungkinkan penggunaan daya komputasi yang lebih efisien, karena beberapa sumber daya komputasi dapat mengakses dan memproses data pada saat yang sama secara paralel.

Hanya ambil data dari layanan penyimpanan bersama ini sesuai kebutuhan dan lepaskan volume yang tidak digunakan untuk membebaskan sumber daya.

Langkah implementasi

- Migrasikan data ke penyimpanan bersama ketika data memiliki beberapa pemakai. Berikut beberapa contoh teknologi penyimpanan bersama di AWS:

Storage option	When to use
Multi-Attach Amazon EBS	Amazon EBS Multi-Attach memungkinkan Anda melampirkan satu volume SSD IOPS yang Tersedia (io1 atau io2) ke beberapa instans yang berada di Zona Ketersediaan yang sama.
Amazon EFS	Lihat Kapan Harus Memilih Amazon EFS.
Amazon FSx	Lihat Memilih Sistem File Amazon FSx.
Amazon S3	Aplikasi yang tidak memerlukan struktur sistem file dan didesain untuk berfungsi dengan penyimpanan objek dapat menggunakan Amazon S3 sebagai solusi penyimpanan

Storage option	When to use
	objek yang dapat diskalakan besar-besaran, tahan lama, dan murah.

- Salin data ke sistem file atau ambil data dari sistem file bersama hanya jika dibutuhkan. Contohnya, Anda dapat membuat sistem file Amazon FSx for Lustreyang didukung oleh Amazon S3 dan hanya memuat subset data yang diperlukan untuk tugas pemrosesan ke Amazon FSx.
- Hapus data sesuai pola penggunaan Anda sebagaimana dijelaskan di SUS04-BP03 Menggunakan kebijakan untuk mengelola siklus hidup set data Anda.
- Lepaskan volume dari klien yang tidak menggunakannya secara aktif.

Sumber daya

Dokumen terkait:

- Menghubungkan sistem file Anda ke bucket Amazon S3
- Menggunakan Amazon EFS untuk AWS Lambda di aplikasi nirserver Anda
- Intelligent-Tiering Amazon EFS Mengoptimalkan Biaya untuk Beban Kerja dengan Mengubah Pola Akses
- Menggunakan Amazon FSx dengan repositori data on-premise Anda

Video terkait:

- Optimisasi biaya penyimpanan dengan Amazon EFS

SUS04-BP07 Meminimalkan perpindahan data di jaringan

Gunakan sistem file atau penyimpanan objek bersama untuk mengakses data umum dan meminimalkan total sumber daya jaringan yang diperlukan untuk mendukung perpindahan data beban kerja Anda.

Antipola umum:

- Anda menyimpan semua data di Wilayah AWS yang sama terlepas di mana pengguna data berada.
- Anda tidak mengoptimalkan format dan ukuran data sebelum memindahkannya melalui jaringan.

Manfaat menjalankan praktik terbaik ini: Mengoptimalkan perpindahan data di seluruh jaringan mengurangi total sumber daya jaringan yang diperlukan untuk beban kerja dan memperkecil dampaknya pada lingkungan.

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

Panduan implementasi

Memindahkan data ke berbagai bagian dalam organisasi memerlukan sumber daya komputasi, jaringan, dan penyimpanan. Gunakan teknik untuk meminimalkan perpindahan data dan meningkatkan efisiensi beban kerja Anda secara keseluruhan.

Langkah implementasi

- Pertimbangkan kedekatan jarak ke data atau pengguna sebagai faktor penentu ketika memilih Wilayah untuk beban kerja Anda.
- Partisi layanan yang digunakan secara Regional sehingga data khusus Wilayahnya disimpan di Wilayah tempat penggunaan data.
- Gunakan format file yang efisien (seperti Parquet atau ORC) dan kompresi data sebelum memindahkannya melalui jaringan.
- Jangan pindahkan data yang tidak digunakan. Beberapa contoh tindakan yang dapat membantu Anda menghindari pemindahan data yang tidak digunakan:
  - Kurangi respons API untuk data yang relevan saja.
  - Kumpulkan data apabila terperinci (informasi tingkat catatan tidak diperlukan).
  - Lihat Lab Well-Architected - Mengoptimalkan Pola Data Menggunakan Fitur Berbagi Data Amazon Redshift.
  - Pertimbangkan Berbagi data lintas akun di AWS Lake Formation.
- Gunakan layanan yang dapat membantu Anda menjalankan kode lebih dekat dengan pengguna beban kerja Anda.

Layanan	Kapan harus digunakan
Lambda@Edge	Gunakan untuk operasi dengan banyak komputasi yang dijalankan saat objek tidak ada dalam cache.

Layanan	Kapan harus digunakan
Fungsi CloudFront	Gunakan untuk kasus penggunaan sederhana seperti permintaan HTTP atau manipulasi respons yang dapat dimulai oleh fungsi dengan masa pakai singkat.
AWS IoT Greengrass	Jalankan komputasi lokal, olahpesan, dan caching data untuk perangkat yang terhubung.

Sumber daya

Dokumen terkait:

- Mengoptimalkan Infrastruktur AWS untuk Pelestarian Lingkungan, Bagian III: Jaringan
- Infrastruktur Global AWS
- Fitur Utama Amazon CloudFront meliputi Jaringan Edge Global CloudFront
- Mengompresi permintaan HTTP di Amazon OpenSearch Service
- Kompresi data menengah dengan Amazon EMR
- Memuat file kompresi data dari Amazon S3 ke Amazon Redshift
- Menyajikan file kompresi dengan Amazon CloudFront

Video terkait:

- Menjelaskan transfer data di AWS

Contoh terkait:

- Arsitektur untuk keberlanjutan - Meminimalkan pergerakan data lintas jaringan

SUS04-BP08 Hanya mencadangkan data saat sulit untuk dibuat ulang

Hindari mencadangkan data yang tidak memiliki nilai bisnis untuk meminimalkan persyaratan sumber daya penyimpanan untuk beban kerja Anda.

Antipola umum:

- Anda tidak memiliki strategi cadangan untuk data Anda.
- Anda mencadangkan data yang dapat dibuat ulang dengan mudah.

Manfaat menjalankan praktik terbaik ini: Menghindari pencadangan data yang tidak penting mengurangi sumber daya penyimpanan yang diperlukan untuk beban kerja dan memperkecil dampaknya pada lingkungan.

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

Panduan implementasi

Menghindari pencadangan data yang tidak perlu dapat membantu menurunkan biaya dan mengurangi sumber daya penyimpanan yang digunakan oleh beban kerja. Hanya cadangkan data yang memiliki nilai bisnis atau yang diperlukan untuk memenuhi persyaratan kepatuhan. Periksa kebijakan pencadangan dan jangan sertakan penyimpanan sementara yang tidak memberikan nilai dalam skenario pemulihan.

Langkah implementasi

- Implementasikan kebijakan klasifikasi data sebagaimana dijelaskan di SUS04-BP01 Mengimplementasikan kebijakan klasifikasi data.
- Gunakan kritikalitas klasifikasi data Anda dan desain strategi pencadangan berdasarkan sasaran waktu pemulihan (RTO) dan sasaran titik pemulihan (RPO). Hindari mencadangkan data yang tidak penting.
  - Jangan sertakan data yang dapat dibuat ulang dengan mudah.
  - Jangan sertakan data sementara dari cadangan Anda.
  - Jangan sertakan salinan lokal data, kecuali apabila waktu yang diperlukan untuk memulihkan data tersebut dari lokasi umum melebihi perjanjian tingkat layanan (SLA) Anda.
- Gunakan solusi otomatis atau layanan terkelola untuk mencadangkan data yang penting bagi bisnis.
  - AWS Backup adalah layanan terkelola penuh yang mempermudah pemusatan dan pengotomatisan perlindungan data di seluruh layanan AWS, di cloud, dan on-premise. Untuk panduan praktik langsung tentang cara membuat cadangan otomatis menggunakan AWS Backup, lihat Well-Architected Labs - Pengujian Pencadangan dan Pemulihan Data.
  - Otomatiskan cadangan dan optimalkan biaya cadangan untuk Amazon EFS menggunakan AWS Backup.

Sumber daya

Praktik terbaik terkait:

- REL09-BP01 Mengidentifikasi dan mencadangkan semua data yang perlu dicadangkan, atau memproduksi ulang data dari sumber
- REL09-BP03 Melakukan pencadangan data secara otomatis
- REL13-BP02 Menggunakan strategi pemulihan untuk memenuhi sasaran pemulihan

Dokumen terkait:

- Menggunakan AWS Backup untuk mencadangkan dan memulihkan sistem file Amazon EFS
- Snapshot Amazon EBS
- Bekerja dengan cadangan di Amazon Relational Database Service
- Partner APN: partner yang dapat membantu pencadangan
- AWS Marketplace: produk yang dapat digunakan untuk pencadangan
- Mencadangkan Amazon EFS
- Mencadangkan Amazon FSx untuk Windows File Server
- Pencadangan dan Pemulihan untuk Amazon ElastiCache for Redis

Video terkait:

- AWS re:Invent 2021 - Pencadangan, pemulihan bencana, dan perlindungan ransomware dengan AWS
- Demo AWS Backup: Pencadangan Lintas Akun dan Lintas Wilayah
- AWS re:Invent 2019: Memahami AWS Backup, dengan Rackspace (STG341)

Contoh terkait:

- Well-Architected Lab - Pengujian Pencadangan dan Pemulihan Data
- Lab Well-Architected - Pencadangan dan Pemulihan dengan Failback untuk Beban Kerja Analitik
- Lab Well-Architected - Pemulihan Bencana - Pencadangan dan Pemulihan

Perangkat keras dan layanan

Cari peluang untuk mengurangi dampak beban kerja terhadap pelestarian lingkungan dengan membuat perubahan pada praktik manajemen perangkat keras Anda. Minimalkan jumlah perangkat keras yang perlu disediakan dan di-deploy, serta pilih perangkat keras dan layanan yang paling efisien untuk setiap beban kerja Anda.

Praktik terbaik

- SUS05-BP01 Menggunakan perangkat keras dalam jumlah minim untuk memenuhi kebutuhan Anda
- SUS05-BP02 Menggunakan jenis instans dengan dampak paling sedikit
- SUS05-BP03 Menggunakan layanan terkelola
- SUS05-BP04 Mengoptimalkan penggunaan akselerator komputasi berbasis perangkat keras

SUS05-BP01 Menggunakan perangkat keras dalam jumlah minim untuk memenuhi kebutuhan Anda

Gunakan perangkat keras dalam jumlah minim untuk beban kerja Anda guna memenuhi kebutuhan bisnis secara efisien.

Antipola umum:

- Anda tidak memantau pemanfaatan sumber daya.
- Anda memiliki sumber daya dengan tingkat pemanfaatan rendah di arsitektur Anda.
- Anda tidak meninjau pemanfaatan perangkat keras statis untuk menentukan apakah harus diubah ukurannya.
- Anda tidak menetapkan target pemanfaatan perangkat keras untuk infrastruktur komputasi Anda berdasarkan KPI bisnis.

Manfaat menjalankan praktik terbaik ini: Menyesuaikan ukuran sumber daya cloud Anda membantu mengurangi dampak beban kerja pada lingkungan, menghemat uang, dan mempertahankan tolok ukur performa.

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

Panduan implementasi

Pilih secara optimal jumlah total perangkat keras yang diperlukan untuk beban kerja Anda guna meningkatkan efisiensinya secara keseluruhan. AWS Cloud memberikan fleksibilitas untuk memperluas atau mengurangi jumlah sumber daya secara dinamis melalui beragam mekanisme, seperti AWS Auto Scaling, dan memenuhi perubahan sesuai permintaan. Layanan ini juga menyediakan API dan SDK yang memungkinkan sumber daya dapat dimodifikasi dengan upaya minimal. Gunakan kemampuan ini untuk membuat perubahan dengan sering pada implementasi beban kerja Anda. Selain itu, gunakan panduan penyesuaian ukuran dari alat AWS untuk secara efisien mengoperasikan sumber daya cloud Anda dan memenuhi kebutuhan bisnis.

Langkah implementasi

- Pilih jenis instans yang paling sesuai dengan kebutuhan Anda.
  - Bagaimana cara memilih jenis instans Amazon EC2 yang tepat untuk beban kerja saya?
  - Pemilihan jenis Instans berdasarkan atribut untuk Armada Amazon EC2.
  - Buat grup Auto Scaling menggunakan pemilihan jenis instans berdasarkan atribut.
- Skalakan menggunakan peningkatan kecil untuk beban kerja variabel.
- Gunakan beberapa opsi pembelian komputasi untuk menyeimbangkan fleksibilitas instans, skalabilitas, dan penghematan biaya.
  - Instans Sesuai Permintaan paling sesuai untuk beban kerja baru, berfluktuasi, dan stateful yang tidak dapat berupa jenis instans, lokasi, atau fleksibel waktunya.
  - Instans Spot merupakan cara yang bagus untuk menambahkan opsi lain untuk aplikasi yang fleksibel dan toleransi terhadap kesalahan.
  - Manfaatkan Compute Savings Plans untuk beban kerja steady state yang memungkinkan fleksibilitas jika kebutuhan Anda (seperti AZ, Wilayah, kelompok instans, atau jenis instans) berubah.
- Gunakan keragaman zona ketersediaan dan instans untuk memaksimalkan ketersediaan aplikasi dan memanfaatkan kapasitas yang berlebih apabila mungkin.
- Gunakan rekomendasi penyesuaian ukuran dari alat AWS untuk melakukan penyesuaian pada beban kerja Anda.
  - AWS Compute Optimizer
  - AWS Trusted Advisor
- Negosiasikan perjanjian tingkat layanan (SLA) agar kapasitas dapat dikurangi sementara di saat otomatisasi melakukan deployment sumber daya pengganti.

Sumber daya

Dokumen terkait:

- Mengoptimalkan Infrastruktur AWS untuk Pelestarian Lingkungan, Bagian I: Komputasi
- Pemilihan Jenis Instans Berdasarkan Atribut untuk Auto Scaling untuk Armada Amazon EC2
- Dokumentasi AWS Compute Optimizer
- Mengoperasikan Lambda: Optimisasi performa
- Dokumentasi Penskalaan Otomatis

Video terkait:

- Bangun lingkungan komputasi yang hemat biaya, energi, dan sumber daya

Contoh terkait:

- Well-Architected Lab: Menyesuaikan Ukuran dengan Mengaktifkan AWS Compute Optimizer dan Pemanfaatan Memori (Level 200)

SUS05-BP02 Menggunakan jenis instans dengan dampak paling sedikit

Terus pantau dan gunakan jenis instans baru untuk memanfaatkan peningkatan penghematan energi.

Antipola umum:

- Anda hanya menggunakan satu kelompok instans.
- Anda hanya menggunakan instans x86.
- Anda menentukan satu jenis instans dalam konfigurasi Amazon EC2 Auto Scaling Anda.
- Anda menggunakan instans AWS dengan cara yang tidak dirancang untuk instans tersebut (misalnya, Anda menggunakan instans komputasi yang dioptimalkan untuk beban kerja intensif memori).
- Anda tidak mengevaluasi jenis instans baru secara teratur.
- Anda tidak melihat rekomendasi dari alat rightsizing AWS seperti AWS Compute Optimizer.

Manfaat menjalankan praktik terbaik ini: Dengan memanfaatkan instans hemat energi dan berukuran tepat, Anda dapat jauh mengurangi dampak lingkungan dan biaya beban kerja Anda.

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

Panduan implementasi

Menggunakan instans yang efisien di beban kerja cloud sangat penting untuk menurunkan penggunaan sumber daya dan menghemat biaya. Terus pantau rilis instans jenis baru dan manfaatkan peningkatan penghematan energi, termasuk jenis instans yang dirancang untuk mendukung beban kerja spesifik seperti pelatihan dan inferensi machine learning, serta transkode video.

Langkah implementasi

- Pelajari dan jelajahi jenis instans yang dapat menurunkan dampak lingkungan beban kerja Anda.
  - Berlangganan ke Apa yang Baru dengan AWS untuk mendapatkan informasi terbaru terkait teknologi dan instans AWS terbaru.
  - Pelajari tentang jenis instans AWS yang berbeda-beda.
  - Pelajari tentang instans berbasis AWS Graviton yang menawarkan performa terbaik per watt untuk penggunaan energi di Amazon EC2 dengan menonton re:Invent 2020 - Pendalaman tentang instans Amazon EC2 yang didukung prosesor AWS Graviton2 dan Pendalaman tentang AWS Graviton3 dan instans C7g Amazon EC2.
- Rencanakan dan transisikan beban kerja Anda ke jenis instans dengan dampak paling kecil.
  - Tentukan proses untuk mengevaluasi fitur atau instans baru untuk beban kerja Anda. Manfaatkan ketangkasan di cloud untuk menguji dengan cepat bagaimana jenis instans baru dapat meningkatkan pelestarian lingkungan beban kerja Anda. Gunakan metrik proksi untuk mengukur berapa banyak sumber daya yang Anda perlukan untuk menyelesaikan satu unit pekerjaan.
  - Jika memungkinkan, ubah beban kerja menjadi menggunakan jumlah vCPU yang berbeda dan jumlah memori yang berbeda guna memaksimalkan pilihan jenis instans.
  - Pertimbangkan untuk mengalihkan beban kerja Anda ke instans berbasis Graviton guna meningkatkan efisiensi performa beban kerja Anda.
    - AWS Graviton Fast Start
    - Pertimbangan saat mentransisikan beban kerja ke instans Amazon Elastic Compute Cloud berbasis AWS Graviton

    - AWS Graviton2 untuk ISV
  - Pertimbangkan untuk memilih opsi AWS Graviton dalam penggunaan layanan terkelola AWS Anda.
  - Migrasikan beban kerja ke Wilayah yang menawarkan instans dengan dampak paling sedikit terhadap pelestarian lingkungan dan masih memenuhi kebutuhan bisnis Anda.
  - Untuk beban kerja machine learning, manfaatkan perangkat keras yang dibuat khusus untuk beban kerja Anda seperti AWS Trainium, AWS Inferentia, dan Amazon EC2 DL1. Instans AWS Inferentia seperti Inf2 menawarkan kinerja per watt hingga 50% lebih baik daripada instans berbasis Amazon EC2 yang setara.
  - Gunakan Amazon SageMaker Inference Recommender untuk menyesuaikan titik akhir inferensi ML secara tepat.
  - Untuk beban kerja yang berfluktuasi (beban kerja yang jarang memerlukan kapasitas tambahan), gunakan instans performa burstable.
  - Untuk beban kerja stateless dan toleran terhadap kesalahan, gunakan Instans Spot Amazon EC2 guna meningkatkan pemanfaatan cloud secara keseluruhan, serta mengurangi dampak terhadap pelestarian lingkungan dari sumber daya yang tidak digunakan.
- Operasikan dan optimalkan instans beban kerja Anda.
  - Untuk beban kerja sementara, evaluasi metrik Amazon CloudWatch instans seperti `CPUUtilization` untuk mengidentifikasi apakah instans tidak aktif atau kurang dimanfaatkan.
  - Untuk beban kerja stabil, lihat alat rightsizing AWS seperti AWS Compute Optimizer secara berkala untuk mengidentifikasi peluang guna mengoptimalkan dan menyesuaikan ukuran instans dengan tepat.
    - Lab Well-Architected: Rekomendasi Penyesuaian Ukuran
    - Lab Well-Architected - Penyesuaian Ukuran dengan Compute Optimizer
    - Lab Well-Architected - Optimisasi Pola Perangkat Keras dan Pengamatan KPI Pelestarian Lingkungan

Sumber daya

Dokumen terkait:

- Mengoptimalkan Infrastruktur AWS untuk Pelestarian Lingkungan, Bagian I: Komputasi
- AWS Graviton
- Amazon EC2 DL1

- Armada Reservasi Kapasitas Amazon EC2
- Armada Spot Amazon EC2
- Fungsi: Konfigurasi Fungsi Lambda
- Pemilihan jenis instans berdasarkan atribut untuk Armada Amazon EC2
- Membangun Aplikasi yang Berkelanjutan, Efisien, dan Dioptimalkan untuk Biaya di AWS
- Bagaimana Dasbor Pelestarian Lingkungan Contino Membantu Pelanggan Mengoptimalkan Jejak Karbon Mereka

Video terkait:

- Pendalaman tentang instans Amazon EC2 yang didukung prosesor AWS Graviton2
- Pendalaman tentang AWS Graviton3 dan instans C7g Amazon EC2
- Bangun lingkungan komputasi yang hemat biaya, energi, dan sumber daya

Contoh terkait:

- Solusi: Panduan untuk Mengoptimalkan Beban Kerja Deep Learning untuk Pelestarian Lingkungan di AWS
- Lab Well-Architected: Rekomendasi Penyesuaian Ukuran
- Lab Well-Architected - Penyesuaian Ukuran dengan Compute Optimizer
- Lab Well-Architected - Optimisasi Pola Perangkat Keras dan Pengamatan KPI Pelestarian Lingkungan
- Lab Well-Architected - Migrasi Layanan ke Graviton

SUS05-BP03 Menggunakan layanan terkelola

Gunakan layanan terkelola untuk beroperasi dengan lebih efisien di cloud.

Antipola umum:

- Anda menggunakan instans Amazon EC2 dengan pemanfaatan rendah untuk menjalankan aplikasi Anda.
- Tim internal Anda hanya mengelola beban kerja, tanpa ada waktu untuk berfokus pada inovasi atau simplifikasi.

- Anda melakukan deployment dan memelihara teknologi untuk tugas-tugas yang dapat dijalankan dengan lebih efisien di layanan terkelola.

Manfaat menjalankan praktik terbaik ini:

- Menggunakan layanan terkelola mengalihkan tanggung jawab ke AWS, yang memiliki wawasan atas jutaan pelanggan yang dapat membantu mendorong efisiensi dan inovasi baru.
- Layanan terkelola mendistribusikan dampak lingkungan dari layanan ke banyak pengguna karena bidang kendali multi-prinsip.

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

Panduan implementasi

Layanan terkelola mengalihkan tanggung jawab ke AWS untuk mempertahankan pemanfaatan tinggi dan optimisasi pelestarian lingkungan dari deployment perangkat keras. Layanan terkelola juga menghilangkan beban administratif dan operasional pemeliharaan layanan, sehingga tim Anda dapat memiliki lebih banyak waktu dan berfokus pada inovasi.

Tinjau beban kerja Anda untuk mengidentifikasi komponen yang dapat digantikan oleh layanan terkelola AWS. Contoh, Amazon RDS, Amazon Redshift, dan Amazon ElastiCache memberikan layanan basis data terkelola. Amazon Athena, Amazon EMR, dan Amazon OpenSearch Service memberikan layanan analitik terkelola.

Langkah implementasi

1. Inventarisasikan beban kerja Anda untuk layanan dan komponen.
2. Nilai dan identifikasi komponen yang dapat digantikan oleh layanan terkelola. Berikut ini adalah beberapa contoh kapan Anda mungkin perlu mempertimbangkan penggunaan layanan terkelola:

Task	What to use on AWS
Hosting basis data	Gunakan instans Amazon Relational Database Service (Amazon RDS) terkelola dan bukannya memelihara instans Amazon RDS Anda sendiri di Amazon Elastic Compute Cloud (Amazon EC2).

Task	What to use on AWS
Hosting beban kerja kontainer	Gunakan AWS Fargate, dan bukannya implementasi infrastruktur kontainer Anda sendiri.
Hosting aplikasi web	Gunakan AWS Amplify Hosting sebagai layanan hosting dan CI/CD terkelola penuh untuk situs web statis dan render aplikasi web sisi server.

3. Identifikasi dependensi dan buat rencana migrasi. Perbarui runbook dan playbook sesuai dengannya.
    - AWS Application Discovery Service secara otomatis mengumpulkan dan menyajikan informasi terperinci tentang dependensi dan pemanfaatan aplikasi untuk membantu Anda membuat keputusan yang lebih tepat saat merencanakan migrasi Anda.
4. Uji layanan sebelum migrasi ke layanan terkelola.
5. Gunakan rencana migrasi untuk mengganti layanan yang di-host mandiri dengan layanan terkelola.
6. Terus pantau layanan setelah migrasi selesai untuk membuat penyesuaian sebagaimana diperlukan dan optimalkan layanan.

Sumber daya

Dokumen terkait:

- Produk AWS Cloud
- Kalkulator Total Biaya Kepemilikan (TCO) AWS
- Amazon DocumentDB
- Amazon Elastic Kubernetes Service (EKS)
- Amazon Managed Streaming for Apache Kafka (Amazon MSK)

Video terkait:

- Operasi cloud dalam skala besar dengan AWS Managed Services

SUS05-BP04 Mengoptimalkan penggunaan akselerator komputasi berbasis perangkat keras

Optimalkan penggunaan instans komputasi terakselerasi Anda untuk mengurangi permintaan infrastruktur fisik beban kerja Anda.

Antipola umum:

- Anda tidak memantau penggunaan GPU.
- Anda menggunakan instans tujuan umum untuk beban kerja, padahal instans yang dibuat khusus dapat menghadirkan kinerja lebih tinggi, biaya lebih rendah, dan kinerja per watt yang lebih baik.
- Anda menggunakan akselerator komputasi berbasis perangkat keras untuk tugas yang akan lebih efisien jika menggunakan alternatif berbasis CPU.

Manfaat menjalankan praktik terbaik ini: Dengan mengoptimalkan penggunaan akselerator berbasis perangkat keras, Anda dapat mengurangi permintaan infrastruktur fisik untuk beban kerja Anda.

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

Panduan implementasi

Jika Anda memerlukan kemampuan pemrosesan tinggi, Anda dapat memanfaatkan instans komputasi terakselerasi, yang menyediakan akses ke akselerator komputasi berbasis perangkat keras seperti unit pemrosesan grafis (GPU) dan field programmable gate array (FPGA). Akselerator perangkat keras ini menjalankan fungsi-fungsi tertentu seperti pemrosesan grafis atau pencocokan pola data secara lebih efisien daripada alternatif berbasis CPU. Banyak beban kerja yang terakselerasi, seperti perenderan, transkode, dan machine learning, memiliki variabel tinggi sehubungan dengan penggunaan sumber daya. Jalankan perangkat keras ini hanya ketika diperlukan, dan nonaktifkan instans GPU secara otomatis saat tidak diperlukan, guna meminimalkan sumber daya yang digunakan.

Langkah implementasi

- Identifikasi instans komputasi terakselerasi mana yang dapat memenuhi kebutuhan Anda.
- Untuk beban kerja machine learning, manfaatkan perangkat keras yang dibuat khusus untuk beban kerja Anda, seperti AWS Trainium, AWS Inferentia, dan Amazon EC2 DL1. Instans AWS Inferentia seperti instans Inf2 menawarkan hingga 50% peningkatan kinerja per watt dibandingkan instans Amazon EC2 yang setara.

- Kumpulkan metrik penggunaan untuk instans komputasi terakselerasi Anda. Misalnya, Anda dapat menggunakan agen CloudWatch untuk mengumpulkan metrik seperti `utilization_gpu` dan `utilization_memory` untuk GPU Anda seperti yang ditunjukkan di Kumpulkan metrik GPU NVIDIA dengan Amazon CloudWatch.
- Optimalkan kode, operasi jaringan, dan pengaturan akselerator perangkat keras untuk memastikan perangkat keras yang mendasarinya dimanfaatkan sepenuhnya.
  - Optimalkan pengaturan GPU
  - Pemantauan dan Pengoptimalan GPU dalam AMI Deep Learning
  - Mengoptimalkan I/O untuk penyetelan kinerja GPU pelatihan deep learning di Amazon SageMaker
- Gunakan driver GPU dan pustaka berkinerja tinggi terbaru.
- Gunakan otomatisasi untuk melepaskan instans GPU ketika tidak digunakan.

Sumber daya

Dokumen terkait:

- Komputasi Dipercepat
- Mari Merancang! Merancang dengan chip dan akselerator kustom
- Bagaimana cara memilih jenis instans Amazon EC2 yang tepat untuk beban kerja saya?
- Instans VT1 Amazon EC2
- Pilih akselerator AI dan kompilasi model terbaik untuk inferensi penglihatan komputer dengan Amazon SageMaker

Video terkait:

- Cara memilih instans GPU Amazon EC2 untuk deep learning
- Melakukan Deployment Inferensi Deep Learning yang Hemat Biaya

Proses dan budaya

Cari peluang untuk mengurangi dampak operasi Anda terhadap pelestarian lingkungan dengan membuat perubahan pada praktik deployment, pengujian, dan pengembangan.

Praktik terbaik

SUS06-BP01 Mengadopsi metode yang dapat menghadirkan peningkatan pelestarian lingkungan dengan cepat

Adopsi metode dan proses untuk memvalidasi potensi peningkatan, meminimalkan biaya pengujian, dan memberikan peningkatan kecil.

Antipola umum:

- Meninjau aplikasi Anda untuk pelestarian lingkungan adalah tugas yang dilakukan hanya satu kali pada awal proyek.
- Beban kerja Anda telah menjadi kedaluwarsa, karena proses rilis terlalu merepotkan guna melakukan perubahan kecil untuk efisiensi sumber daya.
- Anda tidak memiliki mekanisme untuk meningkatkan beban kerja untuk pelestarian lingkungan.

Manfaat menjalankan praktik terbaik ini: Dengan menetapkan proses untuk menghadirkan dan melacak peningkatan pelestarian lingkungan, Anda akan dapat terus mengadopsi fitur dan kemampuan baru, menghilangkan masalah, dan meningkatkan efisiensi beban kerja.

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

Panduan implementasi

Uji dan validasi potensi peningkatan pelestarian lingkungan sebelum melakukan deployment peningkatan ini ke produksi. Pertimbangkan biaya pengujian saat menghitung potensi manfaat sebuah peningkatan untuk masa depan. Kembangkan metode pengujian berbiaya rendah untuk memberikan peningkatan kecil.

Langkah implementasi

- Tambahkan persyaratan pelestarian lingkungan ke backlog pengembangan Anda.

- Gunakan proses peningkatan berulang untuk mengidentifikasi, mengevaluasi, memprioritaskan, menguji, dan melakukan deployment peningkatan ini.
- Terus tingkatkan dan sederhanakan proses pengembangan Anda. Sebagai contoh, Otomatiskan proses pengiriman perangkat lunak Anda menggunakan pipeline integrasi dan pengiriman berkelanjutan (CI/CD) untuk menguji dan melakukan deployment potensi peningkatan untuk mengurangi tingkat upaya dan membatasi kesalahan yang disebabkan oleh proses manual.
- Kembangkan dan uji potensi peningkatan menggunakan komponen representatif yang dapat digunakan pada tingkat minimum untuk mengurangi biaya pengujian.
- Terus nilai dampak peningkatan dan buat penyesuaian jika diperlukan.

Sumber daya

Dokumen terkait:

- AWS memungkinkan solusi pelestarian lingkungan
- Praktik pengembangan tangkas yang dapat diskalakan berdasarkan AWS CodeCommit

Video terkait:

- Menghadirkan arsitektur pelestarian lingkungan dengan performa tinggi

Contoh terkait:

- Well-Architected Lab - Mengubah laporan biaya & penggunaan menjadi laporan efisiensi

SUS06-BP02 Selalu pastikan beban kerja Anda mutakhir

Selalu pastikan beban kerja Anda mutakhir untuk mengadopsi fitur yang efisien, menghilangkan masalah, dan meningkatkan efisiensi beban kerja Anda secara keseluruhan.

Antipola umum:

- Anda berasumsi bahwa arsitektur Anda saat ini statis dan tidak akan diperbarui seiring waktu.
- Anda tidak memiliki sistem atau koordinasi rutin untuk mengevaluasi apakah perangkat lunak dan paket yang diperbarui kompatibel dengan beban kerja Anda.

Manfaat menjalankan praktik terbaik ini: Dengan menetapkan proses untuk memastikan beban kerja Anda mutakhir, Anda dapat menerapkan fitur dan kemampuan baru, menyelesaikan masalah, dan meningkatkan efisiensi beban kerja.

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

Panduan implementasi

Sistem operasi, runtime, perangkat lunak perantara (middleware), pustaka, dan aplikasi yang mutakhir dapat meningkatkan efisiensi beban kerja serta memudahkan pengadopsian teknologi yang lebih efisien. Perangkat lunak yang mutakhir juga dapat menyertakan fitur-fitur untuk mengukur dampak beban kerja terhadap pelestarian lingkungan secara lebih akurat, mengingat vendor juga menghadirkan fitur-fitur untuk memenuhi tujuan pelestarian lingkungan mereka sendiri. Secara teratur jaga agar beban kerja Anda mutakhir dengan fitur dan rilis terbaru.

Langkah implementasi

- Tentukan proses dan jadwal untuk mengevaluasi fitur atau instans baru untuk beban kerja Anda. Manfaatkan ketangkasan di cloud untuk menguji dengan cepat bagaimana fitur baru dapat meningkatkan beban kerja Anda untuk:
  - Mengurangi dampak pelestarian lingkungan.
  - Memperoleh efisiensi performa.
  - Menghilangkan penghalang untuk peningkatan terencana.
  - Meningkatkan kemampuan Anda dalam mengukur dan mengelola dampak terhadap pelestarian lingkungan.
- Buat inventaris perangkat lunak dan arsitektur beban kerja Anda dan identifikasi komponen yang perlu diperbarui.
  - Anda dapat menggunakan AWS Systems Manager Inventory untuk mengumpulkan metadata sistem operasi (OS), aplikasi, dan instans dari instans Amazon EC2 dan secara cepat memahami instans mana yang menjalankan perangkat lunak dan konfigurasi yang diperlukan oleh kebijakan perangkat lunak Anda dan instans mana yang perlu diperbarui.
- Pahami cara memperbarui komponen beban kerja Anda.

Workload component	How to update
Gambar mesin	Gunakan EC2 Image Builder untuk mengelola pembaruan Amazon Machine Image (AMI) untuk gambar server Linux atau Windows.
Gambar kontainer	Gunakan Amazon Elastic Container Registry (Amazon ECR) dengan pipeline Anda yang ada untuk mengelola Amazon Elastic Container Service (Amazon ECS) gambar.
AWS Lambda	AWS Lambda mencakup fitur manajemen versi.

- Gunakan otomatisasi untuk proses pembaruan guna mengurangi tingkat upaya dalam melakukan deployment fitur baru dan membatasi kesalahan yang disebabkan oleh proses manual.
  - Anda dapat menggunakan CI/CD untuk secara otomatis memperbarui AMI, gambar kontainer, dan artefak lain yang terkait dengan aplikasi cloud Anda.
  - Anda dapat menggunakan alat seperti AWS Systems Manager Patch Manager untuk mengotomatiskan proses pembaruan sistem, dan menjadwalkan aktivitas menggunakan AWS Systems Manager Maintenance Windows.

Sumber daya

Dokumen terkait:

- Pusat Arsitektur AWS
- Yang Baru dengan AWS
- Alat Developer AWS

Contoh terkait:

- Well-Architected Labs - Manajemen Inventaris dan Patch
- Lab: AWS Systems Manager

SUS06-BP03 Meningkatkan pemanfaatan lingkungan build

Tingkatkan pemanfaatan sumber daya untuk mengembangkan, menguji, dan membangun beban kerja Anda.

Antipola umum:

- Anda secara manual menyediakan atau menghentikan lingkungan build Anda.
- Anda mempertahankan lingkungan build terus berjalan terlepas dari aktivitas pengujian, build, atau rilis (misalnya, menjalankan lingkungan di luar jam kerja anggota tim pengembangan Anda).
- Anda menyediakan terlalu banyak sumber daya untuk lingkungan build Anda.

Manfaat menjalankan praktik terbaik ini: Dengan meningkatkan pemanfaatan lingkungan build, Anda dapat meningkatkan efisiensi beban kerja cloud Anda secara keseluruhan sekaligus mengalokasikan sumber daya kepada para builder untuk mengembangkan, menguji, dan membangun secara efisien.

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

Panduan implementasi

Gunakan otomatisasi dan infrastruktur sebagai kode untuk mengaktifkan lingkungan build saat diperlukan dan menonaktifkannya saat tidak digunakan. Hal yang umum dilakukan adalah menjadwalkan periode ketersediaan yang bertepatan dengan jam kerja anggota tim pengembangan. Lingkungan uji Anda harus sangat mirip dengan konfigurasi produksi. Tetapi, cari peluang untuk menggunakan jenis instans dengan kapasitas lonjakan, Instans Spot Amazon EC2, layanan basis data penskalaan otomatis, kontainer, dan teknologi nirserver untuk menyesuaikan pengembangan dan menguji kapasitas dengan penggunaan. Batasi volume data untuk tepat memenuhi persyaratan pengujian. Jika menggunakan data produksi dalam pengujian, jelajahi kemungkinan berbagi data dari produksi dan tidak memindahkan data ke mana-mana.

Langkah implementasi

- Gunakan infrastruktur sebagai kode untuk menyediakan lingkungan build Anda.
- Gunakan otomatisasi untuk mengelola siklus hidup pengembangan dan menguji lingkungan serta memaksimalkan efisiensi sumber daya build Anda.
- Gunakan strategi untuk memaksimalkan pemanfaatan lingkungan pengembangan dan pengujian.
  - Gunakan lingkungan representatif yang dapat digunakan pada tingkat minimum untuk mengembangkan dan menguji potensi peningkatan.

- Gunakan teknologi nirserver jika mungkin.
- Gunakan Instans Sesuai Permintaan untuk membantu perangkat developer Anda.
- Gunakan jenis instans dengan kapasitas lonjakan, Instans Spot, dan teknologi lainnya untuk menyesuaikan kapasitas build dengan penggunaan.
- Adopsi layanan cloud native untuk akses shell instans yang aman daripada melakukan deployment armada host bastion.
- Skalakan secara otomatis sumber daya build Anda menurut tugas build.

Sumber daya

Dokumen terkait:

- Manajer Sesi Systems Manager AWS
- Instans performa burstable Amazon EC2
- Apa itu AWS CloudFormation?
- Apa itu AWS CodeBuild?
- Penjadwal Instans di AWS

Video terkait:

- Praktik Terbaik Integrasi Berkelanjutan

SUS06-BP04 Menggunakan device farm terkelola untuk pengujian

Gunakan device farm terkelola untuk secara efisien menguji fitur baru pada serangkaian perangkat keras representatif.

Antipola umum:

- Anda menguji dan melakukan deployment aplikasi di masing-masing perangkat fisik secara manual.
- Anda tidak menggunakan layanan pengujian aplikasi untuk menguji dan berinteraksi dengan aplikasi Anda (contohnya, Android, iOS, dan aplikasi web) pada perangkat fisik nyata.

Manfaat menjalankan praktik terbaik ini: Menggunakan device farm terkelola untuk menguji aplikasi yang didukung cloud memberikan sejumlah manfaat:

- Device farm disertai fitur yang lebih efisien untuk menguji aplikasi di berbagai macam perangkat.
- Device farm terkelola menghilangkan kebutuhan akan infrastruktur internal untuk pengujian.
- Device farm menawarkan berbagai macam jenis perangkat, termasuk perangkat keras yang lebih lama dan kurang populer, yang menghilangkan kebutuhan akan pemutakhiran perangkat yang tidak perlu.

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

Panduan implementasi

Menggunakan device farm terkelola dapat membantu Anda menyederhanakan proses pengujian untuk fitur baru di serangkaian perangkat keras representatif. Device farm terkelola menawarkan berbagai jenis perangkat, termasuk perangkat keras yang lebih lama dan kurang populer, serta menghindari dampak pelestarian lingkungan pelanggan akibat pemutakhiran perangkat yang tidak perlu.

Langkah implementasi

- Tetapkan persyaratan pengujian Anda dan rencanakan (seperti jenis pengujian, sistem operasi, dan jadwal pengujian).
  - Anda dapat menggunakan Amazon CloudWatch RUM untuk mengumpulkan dan menganalisis data di sisi klien dan membentuk rencana pengujian Anda.
- Pilih device farm terkelola yang dapat mendukung persyaratan pengujian Anda. Contohnya, Anda dapat menggunakan AWS Device Farm untuk menguji dan memahami dampak perubahan Anda pada serangkaian perangkat keras representatif.
- Gunakan integrasi berkelanjutan/deployment berkelanjutan (CI/CD) untuk menjadwalkan dan menjalankan pengujian Anda.
  - Mengintegrasikan AWS Device Farm dengan pipeline CI/CD Anda untuk menjalankan uji Selenium di semua browser
  - Membangun dan menguji aplikasi iOS dan iPadOS dengan DevOps AWS dan layanan mobile
- Terus tinjau hasil pengujian Anda dan buat peningkatan yang perlu.

Sumber daya

Dokumen terkait:

- Daftar perangkat AWS Device Farm
- Melihat dasbor RUM CloudWatch

Contoh terkait:

- Contoh Aplikasi AWS Device Farm untuk Android
- Contoh Aplikasi AWS Device Farm untuk iOS
- Uji Web Appium untuk AWS Device Farm

Video terkait:

- Mengoptimalkan aplikasi melalui wawasan pengguna akhir dengan Amazon CloudWatch RUM

Kesimpulan

Jumlah organisasi yang menetapkan target pelestarian lingkungan kian bertambah. Hal ini disebabkan perubahan peraturan pemerintah, keuntungan kompetitif, dan permintaan investor, pegawai, serta pelanggan. CTO, arsitek, developer. dan anggota tim operasi tengah mencari cara agar mereka dapat terlibat secara langsung dalam mencapai tujuan pelestarian lingkungan organisasi. Dengan menggunakan prinsip desain dan praktik terbaik yang didukung AWS ini, Anda dapat membuat keputusan yang matang terkait penyeimbangan keamanan, biaya, kinerja, keandalan, dan keunggulan operasional dengan hasil pelestarian lingkungan untuk beban kerja AWS Cloud Anda. Setiap tindakan yang Anda ambil untuk meminimalkan penggunaan sumber daya dan meningkatkan efisiensi di seluruh beban kerja turut berkontribusi pada pengurangan dampak lingkungan dan berkontribusi pada tujuan pelestarian lingkungan yang lebih luas milik organisasi Anda.

Kontributor

Kontributor dokumen ini antara lain:

- Sam Mokhtari, Senior Efficiency Lead Solutions Architect, Amazon Web Services
- Brendan Sisson, Principal Sustainability Solutions Architect, Amazon Web Services
- Margaret O'Toole, Sustainability Tech Leader, Amazon Web Services
- Steffen Grunwald, Principal Sustainability Solutions Architect, Amazon Web Services
- Ryan Eccles, Principal Engineer, Sustainability, Amazon
- Rodney Lester, Principal Architect, Amazon Web Services
- Adrian Cockcroft, VP Sustainability Architecture, Amazon Web Services
- Ian Meyers, Director of Technology, Solutions Architecture, Amazon Web Services

Bacaan lebih lanjut

Untuk informasi tambahan, baca:

- AWS Well-Architected
- Pusat Arsitektur AWS
- Keberlanjutan di Cloud
- AWS menghadirkan solusi pelestarian lingkungan
- The Climate Pledge (Sumpah Iklim)
- Tujuan Pembangunan Berkelanjutan PBB
- Greenhouse Gas Protocol (Protokol Gas Rumah Kaca)

Revisi Dokumen

Berlangganan umpan RSS untuk memperoleh pemberitahuan tentang pembaruan laporan resmi ini.

Perubahan	Deskripsi	Tanggal
Tingkat risiko yang diperbarui	Pembaruan kecil untuk tingkat risiko praktik terbaik.	October 3, 2023
Panduan praktik terbaik yang diperbarui	Praktik terbaik diperbarui dengan panduan baru di area-area berikut: Penyelarasan dengan permintaan, Perangkat lunak dan arsitektur, Data, dan Perangkat keras dan layanan.	July 13, 2023
Diperbarui 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
Publikasi awal	Pilar Pelestarian Lingkunga n - AWS Well-Architected Framework diterbitkan.	December 2, 2021

Pemberitahuan

Pelanggan bertanggung jawab untuk membuat penilaian independen mereka sendiri atas informasi dalam dokumen ini. Dokumen ini: (a) hanya disediakan sebagai informasi, (b) berisi praktik dan penawaran produk AWS saat ini, yang dapat berubah tanpa pemberitahuan, dan (c) tindak menjadi komitmen atau jaminan apa pun dari AWS dan afiliasi, pemasok, atau pemberi lisensinya. Layanan atau produk AWS diberikan “apa adanya” tanpa jaminan, pernyataan, atau syarat apa pun, baik secara tersurat maupun tersirat. Tanggung jawab dan kewajiban AWS kepada pelanggannya dikendalikan oleh perjanjian AWS, dan dokumen ini bukan bagian dari, juga tidak mengubah, perjanjian apa pun antara AWS dan pelanggannya.

AWS Glossary

For the latest AWS terminology, see the AWS glossary in the Glosarium AWS Reference.