Panduan penilaian portofolio aplikasi untuk AWS Cloud migrasi

AWS Panduan Preskriptif

AWS Panduan Preskriptif: Panduan penilaian portofolio aplikasi untuk AWS Cloud migrasi

Table of Contents

Panduan penilaian portofolio aplikasi untuk AWS Cloud migrasi

Goncalves Jerman dan Mark Berner, Amazon Web Services

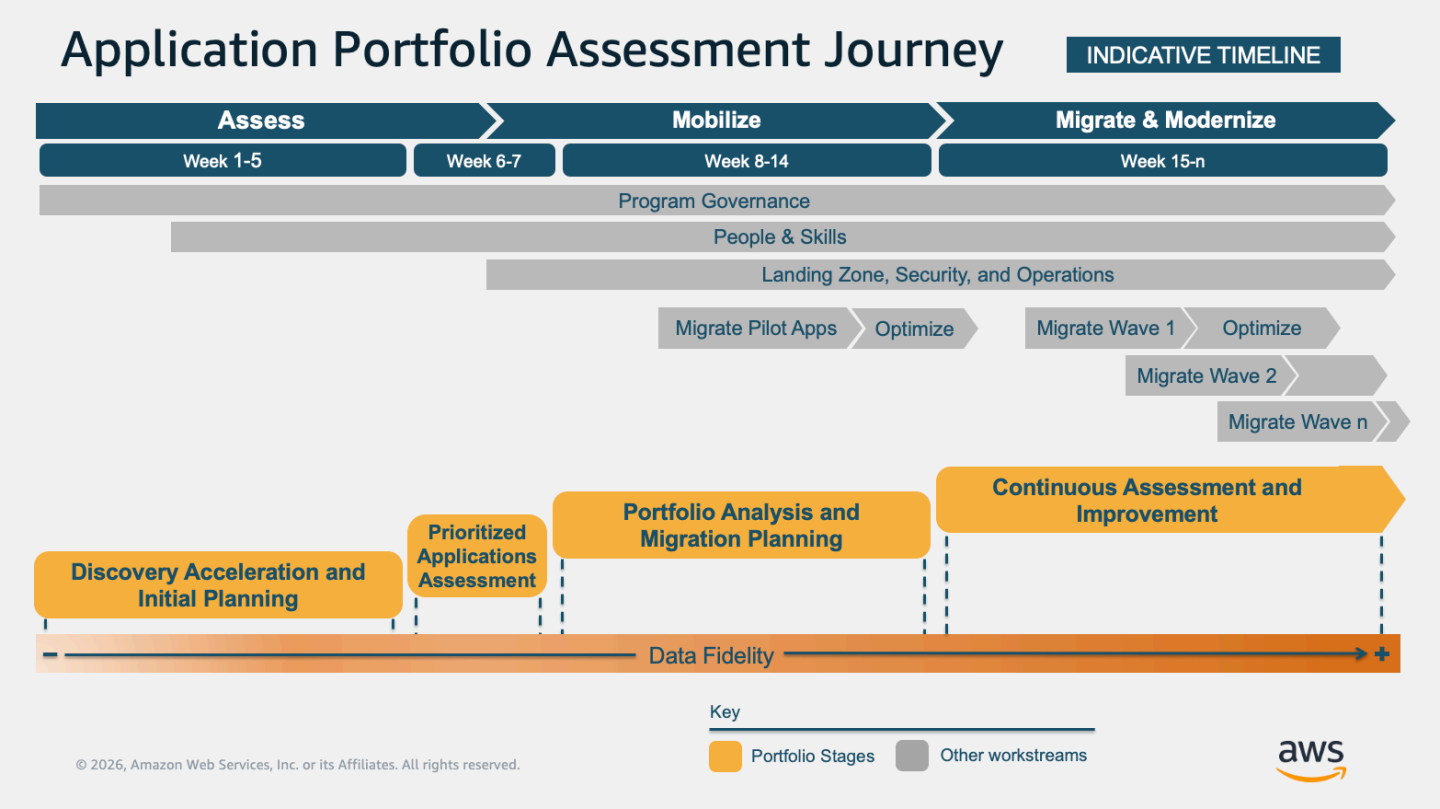
Juni 2026 (sejarah dokumen)

Dokumen Panduan Preskriptif Amazon Web Services (AWS) ini menyelami penerapan strategi penilaian portofolio aplikasi. Anda dapat menggunakan panduan ini untuk membantu Anda memulai dan maju melalui penilaian portofolio aplikasi dan infrastruktur terkait Anda. Penilaian meliputi penemuan, analisis, dan perencanaan. Infrastruktur meliputi komputasi, penyimpanan, dan jaringan.

Ikhtisar

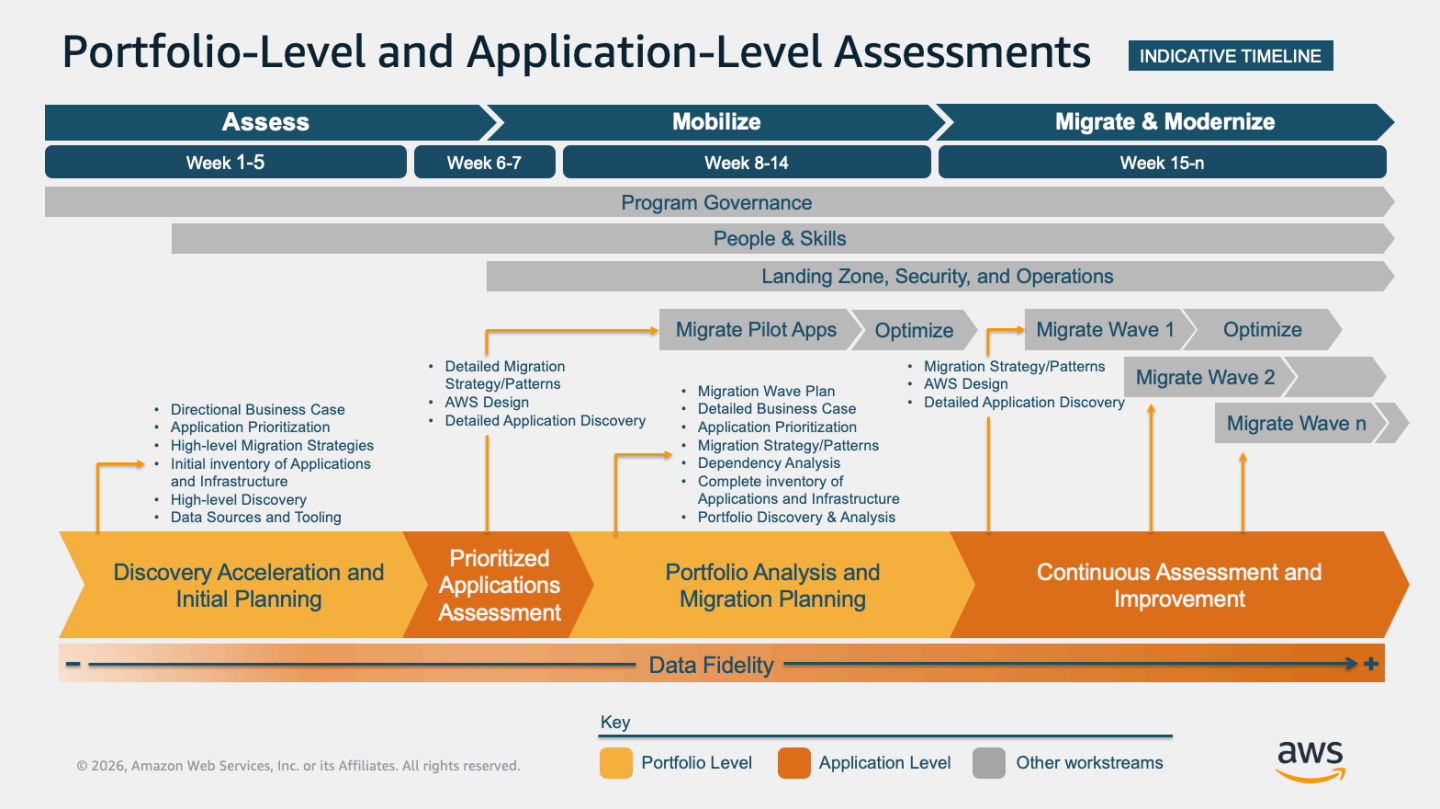
Long-running Program migrasi cloud memerlukan koordinasi beberapa workstream seperti tata kelola program, landing zone (lingkungan target operatif dengan kontrol keamanan), migrasi, dan portofolio aplikasi. Nama-nama alur kerja ini dapat bervariasi tergantung pada cara Anda memilih untuk mengatur program migrasi Anda. Sebagai alur kerja, penilaian portofolio aplikasi merupakan aktivitas dasar di seluruh siklus hidup program ini. Pemahaman tentang portofolio yang diperoleh melalui penilaian memberikan masukan kunci ke alur kerja lain yang bergantung pada data dan analisis yang dihasilkan dari penilaian portofolio aplikasi berkelanjutan.

Diagram berikut menunjukkan bagaimana tahapan penilaian portofolio sesuai dengan AWS fase migrasi dan alur kerja lainnya. Penemuan portofolio dan tahap perencanaan awal dimulai pada fase penilaian, biasanya selama lima minggu pertama. Penilaian aplikasi yang diprioritaskan, pada minggu keenam dan ketujuh, mencakup fase penilaian dan mobilisasi. Analisis portofolio dan tahap perencanaan migrasi terjadi pada minggu ke 8-14, dalam fase mobilisasi. Tahap penilaian dan peningkatan berkelanjutan terjadi pada fase migrasi dan modernisasi, dari minggu ke 15 hingga akhir program migrasi. Garis waktu ini bersifat indikatif. Durasi sebenarnya dari tahapan akan tergantung pada organisasi program secara keseluruhan. Tahap penilaian portofolio juga berlaku di luar kerangka kerja ini, dan mereka dapat dimasukkan ke dalam struktur program migrasi apa pun.



- Percepatan penemuan dan perencanaan awal berfokus pada pemahaman portofolio saat ini. Ini termasuk membuat kasus bisnis terarah, menetapkan model rasionalisasi dasar untuk migrasi, dan mengidentifikasi kandidat migrasi awal.
- Penilaian aplikasi yang diprioritaskan memberikan waktu-ke-nilai yang lebih cepat melalui penilaian terperinci, desain awal arsitektur status target, dan identifikasi aplikasi yang dapat dipindahkan dalam jangka pendek. Memindahkan aplikasi dengan cepat memberi tim pengalaman migrasi dan membangun fondasi cloud, seperti landing zone awal dan komponen infrastruktur lainnya.
- Analisis portofolio dan perencanaan migrasi berfokus pada membangun tampilan portofolio aplikasi yang lengkap dan terkini. Pandangan ini dibangun dengan memperkaya dataset portofolio secara berulang, menutup kesenjangan data, mengembangkan kasus bisnis, dan membuat rencana gelombang migrasi dengan kepercayaan tinggi.
- Penilaian dan peningkatan berkelanjutan mendukung migrasi dalam skala besar dengan menghasilkan aplikasi terperinci dan penilaian teknologi untuk setiap gelombang migrasi sebagai aktivitas berkelanjutan. Tahap ini mencakup iterasi rencana gelombang migrasi dan melakukan analisis lebih lanjut dari beban kerja yang dimigrasi untuk optimasi dan modernisasi.

Diagram berikut menunjukkan kegiatan utama untuk setiap tahap penilaian dan bagaimana mereka berputar antara penilaian tingkat portofolio dan penilaian tingkat aplikasi. Portfolio-level penilaian berfokus pada penemuan tingkat tinggi dan analisis keseluruhan portofolio. Misalnya, sumber data portofolio, inventaris aplikasi dan infrastruktur, prioritas, strategi migrasi tingkat tinggi, dan kasus bisnis terarah. Application-levelpenilaian berfokus pada penemuan terperinci dari satu atau lebih aplikasi. Misalnya, penemuan aplikasi terperinci, AWS desain target, dan strategi migrasi terperinci di tingkat arsitektur dan teknologi aplikasi. Portfolio-level dan penilaian tingkat aplikasi mewakili luas dan kedalaman informasi yang diperlukan.



Akselerasi penemuan dan perencanaan awal

Tahap pertama penilaian portofolio ini berfokus pada langkah-langkah awal memperoleh dan menganalisis data di tingkat portofolio. Tujuan utamanya adalah untuk mengidentifikasi driver bisnis dan mengumpulkan data umum dari aplikasi dan infrastruktur untuk mendapatkan pandangan awal portofolio. Data ini mencakup atribut teknis dan bisnis tingkat tinggi seperti nama aplikasi, lingkungan, versi produk, kekritisan, nilai kinerja, dan lainnya, seperti yang dijelaskan di bagian persyaratan data. Menyelesaikan tahap ini adalah kunci untuk memahami ruang lingkup proyek, mengidentifikasi kandidat migrasi awal, dan menginformasikan kasus bisnis.

Hasil utama dari tahap ini

- Penggerak bisnis, hasil, tujuan, dan prinsip panduan teknis yang terdokumentasi.
- Inventarisasi awal aplikasi dan infrastruktur, dan kesenjangan data yang diidentifikasi. Ini adalah pandangan awal dari portofolio yang akan diulang dan disempurnakan dalam tahap selanjutnya.
- Strategi migrasi indikatif per aplikasi. Misalnya, Rehost, Replatform yang akan membantu Anda menentukan upaya indikatif dan timeline.
- Kasus bisnis terarah dan perkiraan biaya untuk bermigrasi.
- Daftar kandidat migrasi awal (misalnya, tiga-lima aplikasi).
- Ditentukan langkah selanjutnya.

Memahami persyaratan data penilaian awal

Pengumpulan data dapat memakan banyak waktu dan dengan mudah menjadi pemblokir ketika tidak ada kejelasan tentang data apa yang dibutuhkan dan kapan dibutuhkan. Kuncinya adalah memahami keseimbangan antara apa yang terlalu sedikit dan apa yang terlalu banyak data untuk hasil tahap ini. Untuk fokus pada data dan tingkat kesetiaan yang diperlukan untuk tahap awal penilaian portofolio ini, mengadopsi pendekatan berulang untuk pengumpulan data.

Sumber data dan persyaratan data

Langkah pertama adalah mengidentifikasi sumber data Anda. Mulailah dengan mengidentifikasi pemangku kepentingan utama dalam organisasi Anda yang dapat memenuhi persyaratan data. Ini biasanya anggota manajemen layanan, operasi, perencanaan kapasitas, pemantauan, dan tim dukungan, dan pemilik aplikasi. Buat sesi kerja dengan anggota kelompok-kelompok ini.

Komunikasikan persyaratan data dan dapatkan daftar alat dan dokumentasi yang ada yang dapat menyediakan data.

Untuk memandu percakapan ini, gunakan serangkaian pertanyaan berikut:

- Seberapa akurat dan mutakhir infrastruktur dan inventaris aplikasi saat ini? Misalnya, untuk database manajemen konfigurasi perusahaan (CMDB), apakah kita sudah tahu di mana celahnya?
- Apakah kami memiliki alat dan proses aktif yang membuat CMDB (atau yang setara) diperbarui? Jika demikian, seberapa sering diperbarui? Apa tanggal penyegaran terbaru?
- Apakah dokumentasi saat ini, seperti CMDB atau yang setara, berisi pemetaan aplikasi-ke-infrastruktur? Apakah setiap aset infrastruktur terkait dengan aplikasi? Apakah setiap aplikasi dipetakan ke infrastruktur?
- Apakah dokumentasi berisi katalog lisensi dan perjanjian lisensi untuk setiap produk?
- Apakah dokumentasi berisi data ketergantungan? Perhatikan keberadaan data komunikasi seperti server ke server, aplikasi ke aplikasi, aplikasi atau server ke database.
- Alat apa lagi yang dapat menyediakan informasi aplikasi dan infrastruktur yang tersedia di lingkungan? Perhatikan keberadaan kinerja, pemantauan, repositori, basis pengetahuan, dan alat manajemen yang dapat digunakan sebagai sumber data.
- Apa saja lokasi yang berbeda, seperti pusat data, hosting aplikasi dan infrastruktur kita?

Setelah pertanyaan-pertanyaan ini dijawab, daftarkan sumber data Anda yang teridentifikasi. Kemudian tetapkan tingkat kesetiaan, atau tingkat kepercayaan, untuk masing-masing dari mereka. Data yang divalidasi baru-baru ini (dalam 30 hari) dari sumber program aktif, seperti alat, memiliki tingkat kesetiaan tertinggi. Data statis dianggap memiliki kesetiaan yang lebih rendah dan kurang dipercaya. Contoh data statis adalah dokumen, buku kerja, CMDB yang diperbarui secara manual, atau kumpulan data lain yang tidak dipelihara secara terprogram, atau yang tanggal penyegaran terakhirnya lebih dari 60 hari.

Tingkat kesetiaan data dalam tabel berikut disediakan sebagai contoh. Kami menyarankan Anda menilai persyaratan organisasi Anda dalam hal toleransi maksimum terhadap asumsi dan risiko terkait untuk menentukan tingkat kesetiaan yang sesuai. Dalam tabel, pengetahuan kelembagaan mengacu pada informasi apa pun tentang aplikasi dan infrastruktur yang tidak didokumentasikan.

Sumber data	Penilaian tingkat kesetiaan	Cakupan portofolio	Komentar

Pengetahuan kelembagaan	Rendah - Hingga 25% dari data akurat, 75% nilai atau data yang diasumsikan lebih tua dari 150 hari.	Rendah	Langka, fokus pada aplikasi kritis
Basis pengetahuan	Medium-low - 35-40% dari data akurat, 65-60% nilai atau data yang diasumsikan berusia 120-150 hari.	Sedang	Tingkat detail yang dipelihara secara manual dan tidak konsisten
CMDB	Sedang - ~ 50% dari data akurat, ~ 50% nilai atau data yang diasumsikan berusia 90-120 hari.	Sedang	Berisi data dari sumber campuran, beberapa kesenjang an data
Ekspor vCenter VMware	Medium-high - 75-80% data akurat, 25-20% nilai asumsi atau data berusia 60-90 hari.	Tinggi	Mencakup 90% dari perkebunan virtual
Pemantauan kinerja aplikasi	Tinggi - Sebagian besar data akurat, ~ 5% nilai atau data yang diasumsikan berusia 0-60 hari.	Rendah	Terbatas untuk sistem produksi kritis (mencakup 15% dari portofolio aplikasi)

Tabel berikut menentukan atribut data yang diperlukan dan opsional untuk setiap kelas aset (aplikasi, infrastruktur, jaringan, dan migrasi), aktivitas spesifik (inventaris atau kasus bisnis), dan kesetiaan data yang direkomendasikan untuk tahap penilaian ini. Tabel menggunakan singkatan berikut:

- R, untuk yang dibutuhkan
- (D), untuk kasus bisnis terarah, diperlukan untuk perbandingan biaya kepemilikan total (TCO) dan kasus bisnis terarah

- (F), untuk kasus bisnis terarah penuh, diperlukan untuk perbandingan TCO dan kasus bisnis terarah yang mencakup biaya migrasi dan modernisasi
- O, untuk opsional
- N/A, karena tidak berlaku

Aplikasi

Nama atribut	Deskripsi	Inventarisasi dan prioritas	Kasus bisnis	Tingkat kesetiaan yang disarankan (minimum)
Pengidentifikasi unik	Misalnya, ID aplikasi. Biasanya tersedia pada CMDB yang ada atau inventaris internal dan sistem kontrol lainnya. Pertimbangkan untuk membuat ID unik setiap kali ini tidak ditentukan dalam organisasi Anda.	R	R (D)	Tinggi
Nama aplikasi	Nama yang dengannya aplikasi ini diketahui organisasi Anda. Sertakan vendor komersial off-the-shelf	R	R (D)	Medium-high

	(COTS) dan nama produk bila berlaku.			
Apakah COTS?	Ya atau Tidak. Apakah ini aplikasi komersial atau pengembangan internal	R	R (D)	Medium-high
Produk dan versi COTS	Nama dan versi produk perangkat lunak komersial	R	R (D)	Sedang
Deskripsi	Fungsi dan konteks aplikasi utama	R	O	Sedang
Kekritisan	Misalnya, aplikasi strategis atau menghasilkan pendapatan, atau mendukung fungsi kritis	R	O	Medium-high
Tipe	Misalnya, database, manajemen hubungan pelanggan (CRM), aplikasi web, multimedia, layanan bersama TI	R	O	Sedang

Lingkungan	Misalnya, produksi, pra-produksi, pengembangan, pengujian, kotak pasir	R	R (D)	Medium-high
Kepatuhan dan peraturan	Kerangka kerja yang berlaku untuk beban kerja (misalnya , HIPAA, SOX,, ISO, SOC PCI-DSS, FedRAMP) dan persyaratan peraturan	R	R (D)	Medium-high
Dependensi	Dependensi hulu dan hilir ke aplikasi atau layanan internal dan eksternal. Non-technical dependensi seperti elemen operasional (misalnya, siklus pemeliharaan)	O	O	Medium-low
Pemetaan infrastruktur	Pemetaan ke aset and/or virtual fisik yang membentuk aplikasi	R	O	Sedang

Lisensi	Jenis lisensi perangkat lunak komoditas (misalnya, Microsoft SQL Server Enterprise)	O	R	Medium-high
Biaya	Biaya untuk lisensi perangkat lunak, operasi perangkat lunak, dan pemeliharaan	O	O	Sedang

Infrastruktur

Nama Atribut	Deskripsi	Inventarisasi dan prioritas	Kasus bisnis	Tingkat kesetiaan yang disarankan (minimum)
Pengidentifikasi unik	Misalnya, ID server. Biasanya tersedia pada CMDB yang ada atau inventaris internal dan sistem kontrol lainnya. Pertimbangkan untuk membuat ID unik setiap kali ini tidak	R	R	Tinggi

	ditentukan dalam organisasi Anda.			
Nama jaringan	Nama aset dalam jaringan (misalnya, nama host)	R	O	Medium-high
Nama DNS (nama domain yang sepenuhnya memenuhi syarat, atau FQDN)	Nama DNS	O	O	Sedang
Alamat IP dan netmask	Alamat IP and/or publik internal	O	O	Medium-high
Jenis aset	Server fisik atau virtual, hypervisor, wadah, perangkat, instance database, dll.	R	R	Medium-high
Nama produk	Vendor komersial dan nama produk (misalnya, VMware ESXi, IBM Power Systems, Exadata)	R	R	Sedang
Sistem operasi	Misalnya, REHL 8, Windows Server 2019, AIX 6.1	R	R	Medium-high

Konfigurasi	CPU yang dialokasikan, jumlah inti, utas per inti, total memori, penyimpanan, kartu jaringan	R	R	Medium-high
Pemanfaatan	CPU, memori, dan puncak penyimpanan dan rata-rata . Throughput instance database.	O	O	Medium-high
Lisensi	Jenis lisensi komoditas (misalnya, Standar RHEL)	O	R	Sedang
Pemetaan aplikasi	Aplikasi atau komponen aplikasi yang berjalan di infrastruktur ini	R	O	Sedang

Biaya	Biaya terisi penuh untuk server bare-metal, termasuk perangkat keras, pemeliharaan, operasi, penyimpanan (SAN, NAS, objek), lisensi sistem operasi, berbagi rackspace, dan overhead pusat data	O	O	Medium-high

Jaringan

Nama atribut	Deskripsi	Inventarisasi dan prioritas	Kasus bisnis	Tingkat kesetiaan yang disarankan (minimum)
Ukuran pipa (Mb/s), redundansi () Y/N	Spesifikasi tautan WAN saat ini (mis., 1000 Mb/s redundan)	O	R	Sedang
Lokasi yang terhubung	Lokasi bernama yang dihubungkan oleh tautan ini	O	O	Sedang
Pemanfaatan tautan	Puncak dan pemanfaatan rata-rata, transfer	O	R	Sedang

	data keluar () GB/month			
Latensi (ms)	Latensi saat ini antara lokasi yang terhubung	O	O	Sedang
Biaya	Biaya saat ini per bulan	N/A	O	Sedang

Migrasi

Nama Atribut	Deskripsi	Inventarisasi dan prioritas	Kasus bisnis	Tingkat kesetiaan yang disarankan (minimum)
Biaya rehost	Upaya pelanggan dan mitra untuk setiap beban kerja (orang-hari), tarif biaya pelanggan dan Mitra per hari, biaya alat, jumlah beban kerja	N/A	R (F)	Medium-high
Biaya replatform	Upaya pelanggan dan mitra untuk setiap beban kerja (orang-hari), tarif biaya pelanggan dan mitra per hari,	N/A	R (F)	Medium-high

	jumlah beban kerja			
Biaya refactor	Upaya pelanggan dan mitra untuk setiap beban kerja (orang-hari), tarif biaya pelanggan dan mitra per hari, jumlah beban kerja	N/A	R (F)	Medium-high
Biaya pensiun	Jumlah server, biaya dekomisi rata-rata	N/A	O	Medium-high
Zona pendaratan	Re-use existing (Y/N), daftar Wilayah AWS yang dibutuhkan, biaya	N/A	R (F)	Medium-high
Orang dan perubahan	Jumlah staf yang akan dilatih dalam operasi dan pengembangan cloud, biaya pelatihan per orang, biaya waktu pelatihan per orang	N/A	R (F)	Medium-high
Durasi	Durasi migrasi beban kerja dalam lingkup (bulan)	O	R (F)	Medium-high

Biaya paralel	Kerangka waktu dan tingkat biaya apa adanya dapat dihapus selama migrasi	N/A	O	Medium-high
Biaya paralel	Kerangka waktu dan tingkat di mana produk dan layanan AWS, serta biaya infrastru ktur lainnya, diperkenalkan selama migrasi	N/A	O	Medium-high

Mengevaluasi kebutuhan alat penemuan

Apakah organisasi Anda membutuhkan alat penemuan? Penilaian portofolio membutuhkan kepercayaan tinggi, data terkini tentang aplikasi dan infrastruktur. Tahap awal penilaian portofolio dapat menggunakan asumsi untuk mengisi kesenjangan data. Namun, seiring kemajuan yang dicapai, data dengan ketelitian tinggi membantu Anda membuat rencana migrasi yang sukses dan memperkirakan infrastruktur target dengan benar untuk mengurangi biaya dan memaksimalkan manfaat. Ini juga mengurangi risiko dengan mengaktifkan implementasi yang mempertimbangkan dependensi dan menghindari jebakan migrasi. Kasus penggunaan utama untuk alat penemuan dalam program migrasi cloud adalah untuk mengurangi risiko dan meningkatkan tingkat kepercayaan pada data melalui hal-hal berikut:

- Pengumpulan data otomatis atau terprogram, menghasilkan data yang divalidasi dan sangat tepercaya
- Akselerasi tingkat di mana data diperoleh, meningkatkan kecepatan proyek dan mengurangi biaya
- Peningkatan tingkat kelengkapan data, termasuk data komunikasi dan dependensi yang biasanya tidak tersedia di CMDB
- Memperoleh wawasan seperti identifikasi aplikasi otomatis, analisis TCO, laju operasional yang diproyeksikan, dan rekomendasi pengoptimalan

- High-confidence perencanaan gelombang migrasi

Ketika ada ketidakpastian tentang apakah sistem ada di lokasi tertentu, sebagian besar alat penemuan dapat memindai subnet jaringan dan menemukan sistem yang merespons permintaan ping atau Simple Network Management Protocol (SNMP). Perhatikan bahwa tidak semua konfigurasi jaringan atau sistem akan memungkinkan lalu lintas ping atau SNMP. Diskusikan opsi ini dengan jaringan dan tim teknis Anda.

Tahap lebih lanjut dari penilaian portofolio aplikasi dan migrasi sangat bergantung pada informasi pemetaan ketergantungan yang akurat. Pemetaan dependensi memberikan pemahaman tentang infrastruktur dan konfigurasi yang akan diperlukan di AWS (seperti grup keamanan, jenis instans, penempatan akun, dan perutean jaringan). Ini juga membantu pengelompokan aplikasi yang harus bergerak pada saat yang sama (seperti aplikasi yang harus berkomunikasi melalui jaringan latensi rendah).

Saat memutuskan alat penemuan, penting untuk mempertimbangkan semua tahapan proses penilaian dan untuk mengantisipasi persyaratan data. Kesenjangan data berpotensi menjadi pemblokir, jadi penting untuk mengantisipasinya dengan menganalisis persyaratan data dan sumber data masa depan. Pengalaman di lapangan menentukan bahwa sebagian besar proyek migrasi yang macet memiliki kumpulan data terbatas di mana aplikasi dalam ruang lingkup, infrastruktur terkait, dan dependensinya tidak diidentifikasi dengan jelas. Kurangnya identifikasi ini dapat menyebabkan metrik, keputusan, dan penundaan yang salah. Memperoleh data terbaru adalah langkah pertama menuju proyek migrasi yang sukses.

Bagaimana cara memilih alat penemuan?

Beberapa alat penemuan di pasar menyediakan fitur dan kemampuan yang berbeda. Pertimbangkan kebutuhan Anda. Dan tentukan opsi yang paling tepat untuk organisasi Anda. Faktor paling umum saat memutuskan alat penemuan untuk migrasi adalah sebagai berikut:

Keamanan

- Data apa yang dikumpulkan oleh alat ini?
- Apa metode otentikasi untuk mengakses repositori data alat atau mesin analitik?
- Siapa yang dapat mengakses data, dan apa kontrol keamanan untuk mengakses alat?
- Bagaimana alat mengumpulkan data? Apakah perlu kredensyal khusus?
- Kredensi dan tingkat akses apa yang dibutuhkan alat untuk mengakses sistem saya dan mendapatkan data?

- Bagaimana data ditransfer antar komponen alat?
- Apakah alat ini mendukung enkripsi data saat istirahat dan dalam perjalanan?
- Apakah data terpusat dalam satu komponen di dalam atau di luar lingkungan saya?
- Apa persyaratan jaringan dan firewall?

Pastikan bahwa tim keamanan terlibat dalam percakapan awal tentang alat penemuan.

Kedaulatan data

- Dimana data disimpan dan diproses?
- Apakah alat tersebut menggunakan model perangkat lunak sebagai layanan (SaaS)?
- Apakah itu memiliki kemungkinan untuk menyimpan semua data dalam batas-batas lingkungan saya?
- Dapatkah data disaring sebelum meninggalkan batas-batas organisasi saya?

Pertimbangkan kebutuhan organisasi Anda dalam hal persyaratan residensi data.

Arsitektur

- Infrastruktur apa yang diperlukan untuk menyebarkan alat, dan apa saja komponen yang berbeda?
- Apakah lebih dari satu arsitektur atau model penerapan tersedia?
- Apakah alat ini mendukung pemasangan komponen di zona keamanan yang terkunci di udara?

Performa

- Apa dampak pengumpulan data pada sistem saya?

Kompatibilitas dan ruang lingkup

- Apakah alat ini mendukung semua atau sebagian besar produk dan versi saya? Tinjau dokumentasi alat untuk memverifikasi platform yang didukung terhadap informasi terkini tentang ruang lingkup Anda.
- Apakah sebagian besar sistem operasi saya didukung untuk pengumpulan data? Jika Anda tidak tahu versi sistem operasi Anda, cobalah untuk mempersempit daftar alat penemuan untuk mereka yang memiliki jangkauan yang lebih luas dari sistem yang didukung.

Metode pengumpulan

- Apakah alat ini perlu menginstal agen pada setiap sistem yang ditargetkan?
- Apakah ini mendukung pengumpulan data tanpa agen?
- Apakah agen dan tanpa agen menyediakan fitur yang sama?
- Apa proses pengumpulannya?

Fitur

- Apa saja fitur yang tersedia?
- Dapatkah menghitung total biaya kepemilikan (TCO) dan estimasi AWS Cloud run rate?
- Apakah itu mendukung perencanaan migrasi?
- Apakah itu mengukur kinerja?
- Bisakah itu merekomendasikan AWS infrastruktur target?
- Apakah itu melakukan pemetaan ketergantungan?
- Tingkat pemetaan ketergantungan apa yang disediakannya?
- Apakah itu menyediakan akses API? (misalnya, dapatkah diakses secara terprogram untuk mendapatkan data?)

Pertimbangkan alat dengan fungsi pemetaan ketergantungan aplikasi dan infrastruktur yang kuat dan yang dapat menyimpulkan aplikasi dari pola komunikasi.

Biaya

- Apa model perizinannya?
- Berapa biaya lisensi?
- Apakah harga untuk setiap server? Apakah itu harga berjenjang?
- Apakah ada opsi dengan fitur terbatas yang dapat dilisensikan sesuai permintaan?

Alat penemuan biasanya digunakan di seluruh siklus hidup proyek migrasi. Jika anggaran Anda terbatas, pertimbangkan setidaknya 6 bulan. Namun, tidak adanya alat penemuan biasanya mengarah pada upaya manual dan biaya internal yang lebih tinggi.

Model Support

- Tingkat dukungan apa yang disediakan secara default?
- Apakah ada rencana dukungan yang tersedia?
- Berapa waktu respons insiden?

Layanan profesional

- Apakah vendor menawarkan layanan profesional untuk menganalisis hasil penemuan?
- Bisakah mereka mencakup elemen-elemen panduan ini?
- Apakah ada diskon atau bundel untuk layanan perkakas +?

> **Tip**
>
> Untuk menemukan dan mengevaluasi alat penemuan, gunakan situs Discovery, Planning, dan Rekomendasi.

Fitur yang direkomendasikan untuk alat penemuan

Untuk menghindari penyediaan dan penggabungan data dari beberapa alat dari waktu ke waktu, alat penemuan harus mencakup fitur minimum berikut:

- Perangkat lunak — Alat penemuan harus dapat mengidentifikasi proses yang berjalan dan runtime, paket, dan kerangka kerja yang diinstal.
- Pemetaan ketergantungan — Ini harus dapat mengumpulkan informasi koneksi jaringan dan membangun peta ketergantungan masuk dan keluar dari server dan aplikasi yang sedang berjalan. Selain itu, alat penemuan harus dapat menyimpulkan aplikasi dari kelompok infrastruktur berdasarkan pola komunikasi.
- Penemuan profil dan konfigurasi - Ini harus dapat melaporkan profil infrastruktur seperti keluarga CPU (misalnya, x86, PowerPC), jumlah inti CPU, ukuran memori, jumlah disk dan ukuran, dan antarmuka jaringan.
- Penemuan penyimpanan jaringan — Ini harus dapat mendeteksi dan memprofilkan berbagi jaringan dari penyimpanan terlampir jaringan (NAS).
- Penemuan basis data — Ini harus dapat menemukan database, contoh, dan skema, termasuk jenis dan versi mesin.

- Kinerja — Ini harus dapat melaporkan puncak dan rata-rata pemanfaatan CPU, memori, disk, dan jaringan.
- Analisis kesenjangan — Ini harus dapat memberikan wawasan tentang kuantitas dan kesetiaan data.
- Pemindaian jaringan — Ini harus dapat memindai subnet jaringan dan menemukan aset infrastruktur yang tidak diketahui.
- Pelaporan — Ini harus dapat memberikan status pengumpulan dan analisis.
- Akses API — Ini harus dapat menyediakan sarana terprogram untuk mengakses data yang dikumpulkan.

Fitur tambahan untuk dipertimbangkan

- Analisis TCO untuk memberikan perbandingan biaya antara biaya lokal saat ini dan biaya yang diproyeksikan AWS .
- Analisis lisensi dan rekomendasi pengoptimalan untuk Microsoft SQL Server dan sistem Oracle dalam skenario rehost dan replatform.
- Rekomendasi strategi migrasi (Dapatkah alat penemuan membuat rekomendasi tipe R migrasi default berdasarkan teknologi saat ini?)
- Ekspor inventaris (ke CSV atau format serupa)
- Right-sizing rekomendasi (misalnya, dapatkah memetakan AWS infrastruktur target yang direkomendasikan?)
- Visualisasi ketergantungan (misalnya, dapatkah pemetaan ketergantungan divisualisasikan dalam mode grafis?)
- Tampilan arsitektur (misalnya, dapatkah diagram arsitektur diproduksi secara otomatis?)
- Prioritas aplikasi (Dapatkah itu menetapkan bobot atau relevansi dengan atribut aplikasi dan infrastruktur untuk membuat kriteria prioritas untuk migrasi?)
- Perencanaan gelombang (misalnya, kelompok aplikasi yang direkomendasikan dan membantu membuat rencana gelombang migrasi)
- Estimasi biaya migrasi (estimasi upaya migrasi)

Pertimbangan deployment

Setelah Anda memilih dan mendapatkan alat penemuan, pertimbangkan pertanyaan berikut untuk mendorong percakapan dengan tim yang bertanggung jawab untuk menerapkan alat di organisasi Anda:

- Apakah server atau aplikasi dioperasikan oleh pihak ketiga? Ini bisa mendikte tim untuk melibatkan dan proses yang harus diikuti.
- Apa proses tingkat tinggi untuk mendapatkan persetujuan untuk menyebarkan alat penemuan?
- Apa proses otentikasi utama untuk mengakses sistem seperti server, kontainer, penyimpanan, dan database? Apakah kredensi server lokal atau terpusat? Bagaimana proses untuk mendapatkan kredensyal? Kredensil akan diperlukan untuk mengumpulkan data dari sistem Anda (misalnya, kontainer, server virtual atau fisik, hypervisor, dan database). Memperoleh kredensi untuk alat penemuan untuk terhubung ke setiap aset dapat menjadi tantangan, terutama ketika aset ini tidak terpusat.
- Apa garis besar zona keamanan jaringan? Apakah diagram jaringan tersedia?
- Bagaimana proses untuk meminta aturan firewall di pusat data?
- Apa perjanjian tingkat layanan dukungan (SLA) saat ini dalam kaitannya dengan operasi pusat data (instalasi alat penemuan, permintaan firewall)?

Penggerak bisnis dan prinsip panduan teknis

Pengemudi bisnis

Apakah organisasi Anda telah memutuskan untuk pindah ke cloud atau mendekati keputusan itu, mendefinisikan dan mendokumentasikan driver bisnis untuk migrasi cloud akan mengklarifikasi alasan migrasi. Setelah alasan didokumentasikan, Anda dapat menentukan apa yang akan dimigrasikan dan bagaimana hal itu akan dimigrasikan. Kegiatan ini penting. Kami menyarankan agar dilakukan sedini mungkin dalam proses untuk menginformasikan dan memandu langkah selanjutnya.

Identifikasi pemangku kepentingan yang harus menjadi bagian dari diskusi untuk mendokumentasikan driver. Biasanya CxOs, manajer senior, dan pemimpin teknologi kunci dalam organisasi, dan pelanggan Anda sendiri. Meskipun pelanggan Anda tidak mungkin menjadi bagian dari diskusi ini, kami menyarankan agar satu atau lebih orang di organisasi Anda ditunjuk untuk mewakili pandangan dan tujuan pelanggan Anda.

Penggerak bisnis harus dikaitkan dengan metrik yang dapat diukur sepanjang perjalanan migrasi untuk memvalidasi apakah hasil telah tercapai. Sasaran strategis dan laporan tahunan perusahaan dapat bertindak sebagai titik awal.

Fokuskan percakapan di mana perusahaan ingin berada, berdasarkan metrik yang ada dan yang diproyeksikan, sebagai hasil dari pindah ke cloud. Pertimbangkan tujuan dan hasil bisnis. Juga, pertimbangkan seperti apa kesuksesan saat adopsi cloud meningkat.

Selanjutnya, tetapkan tingkat kepentingan untuk setiap pengemudi. Apa prioritasnya? Apa manfaat yang diharapkan? Bagaimana manfaat mendukung tujuan dan hasil bisnis? Dalam konteks penilaian portofolio aplikasi, jawabannya akan membantu memprioritaskan beban kerja untuk migrasi dan menetapkan prinsip panduan teknis. Namun, driver bisnis akan menentukan dan berdampak pada program migrasi secara keseluruhan.

Prinsip panduan teknis

Prinsip panduan teknis menginformasikan pemilihan strategi migrasi pada tahap selanjutnya dari penilaian portofolio. Pada tahap saat ini, fokusnya adalah mengidentifikasi mereka.

Prinsip-prinsip panduan dapat ditetapkan sebagai keputusan terkait teknologi umum dan terkait pendekatan yang berasal dari tujuan dan hasil bisnis.

Misalnya, perusahaan memiliki tujuan utama untuk mengurangi biaya, dan hasil yang diinginkan adalah menutup pusat data lokal pada tanggal tertentu dalam 12 bulan. Prinsip panduan yang dihasilkan adalah mengangkat dan memindahkan semua aplikasi ke cloud dengan menggunakan strategi migrasi rehost bila memungkinkan. Dalam hal ini, pendekatan lift-and-shift mempercepat hasil migrasi jangka pendek. Setelah aplikasi dipindahkan dari pusat data lokal, perusahaan dapat fokus pada driver bisnis lain untuk mengoptimalkan atau memodernisasi beban kerja yang dimigrasi.

Untuk menetapkan prinsip-prinsip panduan teknis, mulailah dengan menganalisis driver bisnis. Identifikasi daftar teknologi dan teknik yang akan mencapai tujuan dan hasil bisnis. Selanjutnya, perbaiki daftar dan tetapkan urutan relevansi berdasarkan kesesuaian atau preferensi untuk mencapai hasil yang diinginkan.

Mendokumentasikan dan mengkomunikasikan prinsip-prinsip panduan dengan orang-orang yang terlibat dalam perencanaan dan pelaksanaan migrasi. Sorot kekhawatiran dan potensi konflik antara prinsip dan implementasi aktual.

Tabel berikut memberikan contoh driver bisnis dan prinsip-prinsip panduan teknis.

Pengemudi bisnis	Hasil	Metrik-metrik	Prinsip panduan teknis
Mempercepat inovasi.	Peningkatan daya saing, peningkatan kelincahan bisnis	Jumlah penyebaran per hari atau bulan, fitur baru yang dirilis per kuartal, skor kepuasan pelanggan, jumlah eksperimen	Memfaktorkan ulang aplikasi yang membedakan dengan menggunakan layanan mikro dan model DevOps operasi untuk meningkatkan kelincahan dan kecepatan ke pasar fitur baru.
Mengurangi biaya operasional dan infrastruktur.	Penawaran dan permintaan sesuai, basis biaya elastis (bayar untuk apa yang Anda gunakan)	Variasi pengeluaran dari waktu ke waktu	1. Rehost aplikasi dengan ukuran tepat infrastruktur. 2. Pensiun aplikasi yang memiliki pemanfaatan rendah atau tidak ada.
Meningkatkan ketahanan operasional.	Uptime yang ditingkatkan, mengurangi waktu rata-rata untuk pemulihan	SLA, jumlah insiden	1. Replatform aplikasi ke versi sistem operasi terbaru dan didukung terbaik. 2. Menerapkan arsitektur ketersediaan tinggi untuk aplikasi penting.
Keluar dari pusat data.	Data-center penutupan pada tanggal dalam 6-12 bulan	Kecepatan migrasi server	Rehost aplikasi dengan menggunakan AWS Transform MGN.

Tetap di tempat, tetapi tingkatkan kelincahan dan ketahanan.	Peningkatan daya saing dan uptime sambil tetap berada di tempat	Jumlah penyebaran per hari atau bulan, rilis fitur baru per kuartal, SLA, jumlah insiden	1. Modernisasi sistem dengan memperluas fungsionalitasnya ke cloud. 2. Nilai untuk rehosting atau replatforming ke. AWS Outposts

Penting juga untuk mendefinisikan prinsip-prinsip arsitektur yang menginformasikan strategi migrasi. Misalnya, Anda mungkin memperkenalkan prinsip arsitektur untuk AWS Cloud bahwa semua aplikasi yang dimigrasi harus mengenkripsi semua komunikasi. Jika prinsip semacam ini saat ini tidak diterapkan di tempat, ini memaksa tim migrasi untuk menerapkan perubahan sebagai bagian dari migrasi, yang memengaruhi upaya dan perencanaan. Tentukan semua prinsip arsitektur dan bagikan dengan tim yang relevan.

Memulai pengumpulan data

Pengumpulan data adalah proses pengumpulan metadata dari aplikasi dan infrastruktur. Prosesnya berulang di semua tahap penilaian. Di setiap tahap, kuantitas dan kesetiaan data akan meningkat. Pada tahap ini, fokusnya adalah mengumpulkan data umum yang dapat membantu membangun inventaris awal. Inventaris akan digunakan untuk membuat kasus dan rencana bisnis terarah, dan akan digunakan untuk mengidentifikasi kandidat migrasi awal.

Setelah sumber data saat ini diidentifikasi, kami sarankan untuk mengumpulkan informasi dari sebanyak mungkin sistem. Untuk informasi selengkapnya, lihat persyaratan data untuk tahap ini.

Pendekatan ini memiliki manfaat membantu memperbarui tampilan portofolio saat ini dan pengetahuan organisasi tentang aplikasi dan layanan mereka. Ini juga membantu menentukan apa yang ditargetkan untuk dipindahkan. Pendekatan yang disarankan adalah meninjau data yang ada, seperti output database manajemen konfigurasi (CMDB) dan sistem manajemen layanan teknologi informasi (ITSM). Kemudian buat daftar aset yang ditargetkan untuk pengumpulan data tambahan. Jika organisasi Anda memiliki kejelasan lengkap tentang apa yang ada di ruang lingkup dan di luar cakupan migrasi, Anda dapat membatasi pengumpulan data ke sistem yang berada dalam cakupan.

Saat membangun portofolio Anda, pertimbangkan aplikasi dan lingkungannya atau siklus hidup rilis perangkat lunak. Misalnya, alih-alih mengidentifikasi aplikasi manajemen hubungan pelanggan (CRM) dan menentukan bahwa ia memiliki lingkungan pengujian, pengembang, dan prod, daftarkan tiga aplikasi (misalnya,, CRM-Test, CRM-Dev). CRM-Prod Atau, gunakan nama CRM tetapi tetapkan ID unik untuk setiap lingkungan dan sajikan sebagai catatan terpisah di repositori data Anda. Ini akan membantu merencanakan dan melacak migrasi lingkungan ini secara individual. Misalnya, Anda mungkin ingin memigrasikan lingkungan non-produksi terlebih dahulu. Dengan mencantumkan contoh aplikasi Anda sesuai dengan lingkungan, Anda dapat dengan jelas mengelola dan mengatur transisi mereka.

Selama pengumpulan data, mungkin ada ketidakpastian tentang aplikasi atau server mana yang berada di pusat data atau lokasi sumber tertentu. Dalam kasus ini, mendapatkan daftar bare-metal dan hypervisor dari alat manajemen yang ada sangat membantu. Misalnya, Anda dapat terhubung ke hypervisor untuk mendapatkan daftar mesin virtual yang akan ditargetkan untuk pengumpulan data.

Perhatikan bahwa output awal, saat menggabungkan sumber data yang ada, bisa jadi tidak lengkap. Kuncinya adalah melakukan analisis kesenjangan dalam hal persyaratan data untuk tahap ini dan apa yang dapat diperoleh dari sumber yang ada. Penting untuk membedakan persentase kelengkapan dengan tingkat kesetiaan data. Tingkat kelengkapan yang lebih tinggi dari sumber kesetiaan rendah akan berisi beberapa asumsi yang dapat mengarah pada analisis yang salah. Meskipun tahap penilaian ini tidak memerlukan kesetiaan data maksimum, kami merekomendasikan bahwa sumber data setidaknya memiliki kesetiaan sedang hingga menengah-tinggi. Bandingkan angka-angka ini dengan toleransi organisasi Anda terhadap risiko, termasuk penggunaan asumsi untuk mengisi kesenjangan data.

Analisis kesenjangan membantu Anda memahami kuantitas dan kualitas data yang Anda kerjakan. Analisis ini juga membantu Anda menetapkan tingkat asumsi yang harus dibuat untuk membuat kasus bisnis terarah dan memprioritaskan aplikasi untuk migrasi. Alat penemuan dapat membantu mengisi celah dan mengumpulkan data dengan kesetiaan tinggi. Untuk meningkatkan tingkat kepercayaan pada data dan mempercepat hasil migrasi, sebaiknya gunakan alat penemuan sedini mungkin. Tindakan awal juga penting karena proses pengadaan, keamanan, dan implementasi internal untuk alat baru dapat memerlukan beberapa minggu untuk menyelesaikannya.

Kami merekomendasikan untuk membuat rencana komunikasi atau irama dan mekanisme kontrol perubahan ruang lingkup pada tahap ini. Ini membantu Anda untuk terus memberi tahu para pemangku kepentingan sehingga mereka dapat merencanakan ke depan dan mengurangi risiko. Elemen kunci untuk komunikasi yang jelas adalah mendefinisikan satu sumber kebenaran untuk portofolio aplikasi dan infrastruktur terkait. Hindari menyimpan beberapa sistem catatan dan daftar

aplikasi dan infrastruktur. Simpan data di satu tempat (misalnya, database, alat, atau spreadsheet) yang mendukung pembuatan versi dan kolaborasi online, dan tetapkan pemiliknya.

Strategi prioritas dan migrasi

Elemen kunci dari perencanaan migrasi adalah menetapkan kriteria prioritas. Inti dari latihan ini adalah untuk memahami urutan aplikasi yang akan dimigrasikan. Strateginya adalah mengambil pendekatan berulang dan progresif untuk mengembangkan model prioritas.

Memprioritaskan aplikasi

Tahap penilaian ini berfokus pada penetapan kriteria awal untuk memprioritaskan beban kerja berisiko rendah dan kompleksitas rendah. Beban kerja ini adalah kandidat yang baik untuk aplikasi percontohan. Menggunakan beban kerja berisiko rendah dan kompleksitas rendah dalam migrasi awal mengurangi risiko dan memberi tim kesempatan untuk mendapatkan pengalaman. Kriteria ini akan dikembangkan dalam tahap penilaian lebih lanjut untuk menyelaraskan prioritas dengan penggerak bisnis saat membuat rencana gelombang migrasi.

Kriteria awal harus memprioritaskan aplikasi dengan sejumlah kecil dependensi, berjalan di infrastruktur yang didukung cloud, dan dari lingkungan non-produksi. Contohnya adalah aplikasi dengan 0—3 dependensi yang siap dihosting ulang apa adanya di lingkungan pengembangan atau pengujian. Kriteria ini berlaku untuk mendefinisikan aplikasi percontohan dan berpotensi gelombang migrasi pertama dan kedua, tergantung pada tingkat kematangan adopsi cloud dan tingkat kepercayaan.

Memutuskan kriteria awal apa yang akan digunakan

Pilih 2—10 titik data yang akan digunakan untuk memprioritaskan beban kerja pertama Anda. Titik data ini berasal dari pengumpulan data awal Anda (lihat bagian pengumpulan data).

Selanjutnya, tentukan skor, atau bobot, untuk setiap nilai yang mungkin dari setiap titik data. Misalnya, jika atribut lingkungan dipilih, dan nilai yang mungkin adalah produksi, pengembangan, dan pengujian, setiap nilai diberi skor, angka yang lebih besar mewakili prioritas yang lebih tinggi. Meskipun opsional, kami merekomendasikan untuk menetapkan faktor pengganda untuk kepentingan atau relevansi dengan setiap titik data. Langkah opsional ini memberikan pembeda tingkat yang lebih tinggi untuk menekankan apa yang lebih penting, yang membantu menjaga kriteria tetap selaras saat Anda mengulangi penetapan skor ke nilai.

Berdasarkan strategi untuk memprioritaskan aplikasi sederhana dan berisiko rendah untuk beberapa gelombang migrasi pertama, tabel berikut menunjukkan contoh pemilihan atribut dan penetapan nilainya.

Atribut (titik data)	Kemungkinan nilai	Skor (0-99)	Pentingnya atau relevansi faktor pengalikan
Lingkungan	Uji	80	Tinggi (1x)
	Pengembangan	50	
	Produksi	20	
Kekritisan bisnis	Rendah	60	Tinggi (1x)
	Sedang	40	
	Tinggi	20	
Kerangka peraturan atau kepatuhan	Tidak ada	60	Tinggi (1x)
	FedRAMP	10	
Dukungan sistem operasi	Cloud siap	60	Medium-high (0,8x)
	Tidak didukung di cloud	10	
Jumlah instans komputasi	1-3	60	Medium-high (0,8x)
	4-10	40	
	11 atau lebih	20	
Jumlah dependensi	0-3	70	Tinggi (1x)
	4-10	30	
	11 atau lebih	10	
Strategi migrasi	Rehost	70	Sedang (0.6x)

	Platform Ulang	30	
	Refactor, atau arsitek ulang	10	
Kematangan atau kesiapan cloud tim operasi	Tinggi	80	Tinggi (1x)
	Sedang	50	
	Rendah	10	

Pastikan Anda memilih atribut yang dapat bertindak sebagai pembeda utama antar aplikasi. Jika tidak, kriteria akan menghasilkan banyak beban kerja yang berbagi prioritas yang sama. Setelah Anda menerapkan model, kami sarankan untuk melihat bagian atas dan bawah peringkat yang dihasilkan untuk melihat apakah Anda setuju. Jika Anda umumnya tidak setuju, Anda dapat meninjau kembali kriteria yang Anda gunakan untuk menilai beban kerja.

Setelah Anda mendapatkan peringkat, lihat distribusi skor di seluruh portofolio. Skor itu sendiri tidak masalah. Ini adalah perbedaan antara skor yang penting. Misalnya, Anda mungkin menemukan bahwa skor total teratas adalah 8.000 dan skor terbawah adalah 800. Pertimbangkan untuk memplot skor yang dihasilkan sebagai histogram, sehingga Anda dapat memverifikasi bahwa Anda memiliki distribusi yang baik. Distribusi yang ideal terlihat seperti kurva lonceng standar, dengan beberapa beban kerja prioritas sangat tinggi dan beberapa beban kerja dengan prioritas sangat rendah. Sebagian besar aplikasi akan berada di suatu tempat di tengah.

Aspek kunci lain dari prioritas awal adalah memasukkan tim internal atau unit bisnis yang menunjukkan minat untuk menjadi pengadopsi awal cloud. Ini bisa menjadi tuas yang cukup besar dalam memperoleh dukungan bisnis untuk memigrasikan aplikasi tertentu, terutama di masa-masa awal. Jika ini terjadi di organisasi Anda, sertakan atribut unit bisnis di tabel sebelumnya. Tetapkan skor tinggi untuk unit-unit bisnis yang bersedia untuk maju dengan aplikasi mereka. Menggunakan atribut unit bisnis akan membantu membawa aplikasi tersebut ke bagian atas daftar.

Setelah Anda setuju dengan peringkat yang dihasilkan, pilih 5-10 aplikasi teratas. Ini akan menjadi kandidat migrasi aplikasi awal Anda. Perbaiki daftar sehingga Anda mengonfirmasi 3-5 aplikasi. Ini membantu Anda mengambil pendekatan yang ditargetkan saat melakukan penilaian aplikasi terperinci. Untuk informasi selengkapnya, lihat Penilaian aplikasi yang diprioritaskan.

Menentukan tipe R untuk migrasi

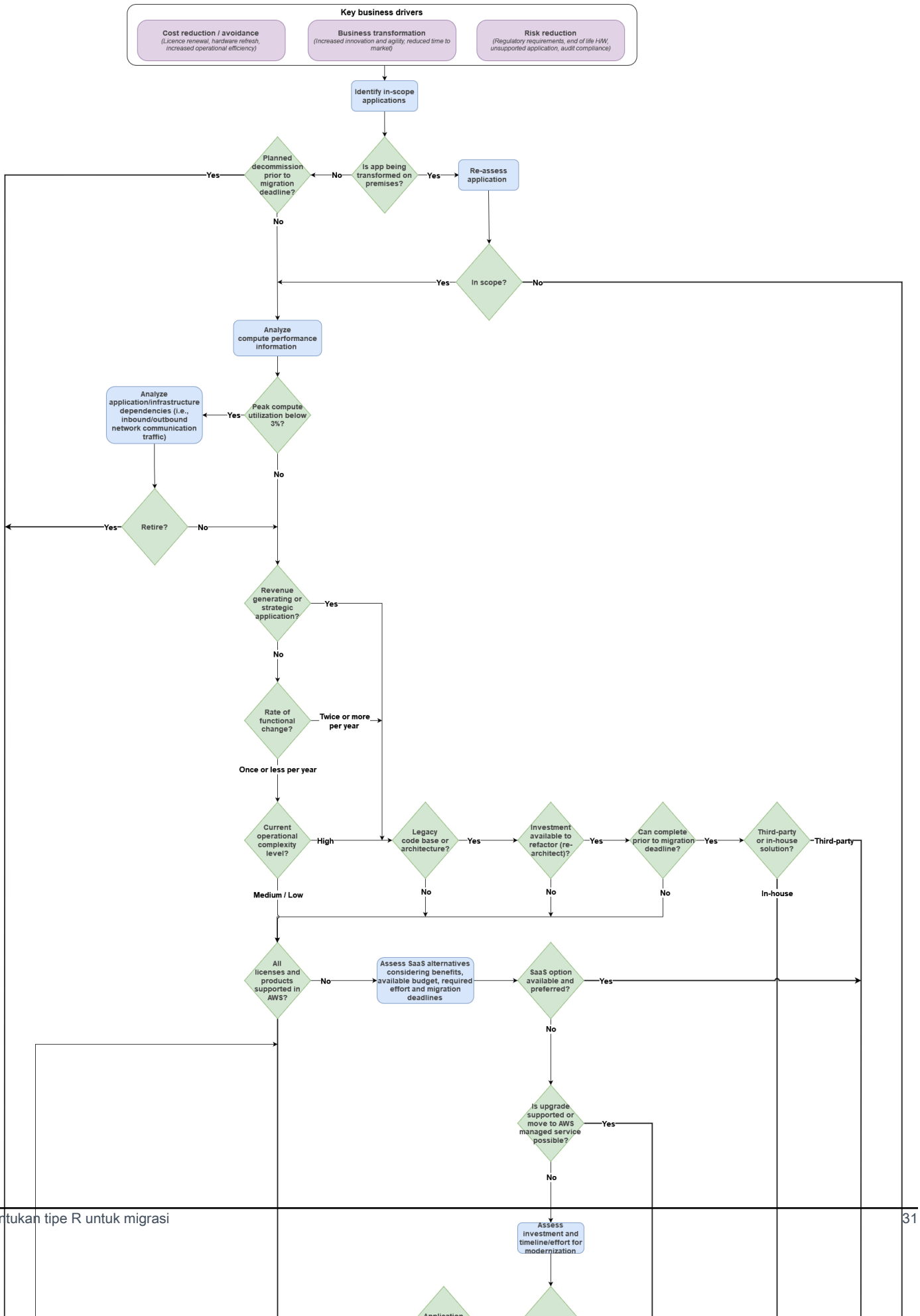
Memutuskan strategi migrasi untuk setiap aplikasi dan infrastruktur terkait akan berimplikasi pada kecepatan migrasi, biaya, dan tingkat manfaat. Ini adalah kunci untuk menentukan strategi berdasarkan kombinasi faktor yang seimbang, termasuk pendorong bisnis, prinsip panduan teknis, kriteria prioritas, dan strategi bisnis.

Terkadang faktor-faktor ini menciptakan pandangan yang saling bertentangan. Misalnya, pendorong utama migrasi mungkin inovasi dan kelincahan. Pada saat yang sama, Anda mungkin perlu mengurangi biaya dengan cepat. Memodernisasi semua aplikasi dalam lingkup akan mengurangi biaya dalam jangka panjang, tetapi akan membutuhkan investasi yang lebih besar di muka. Dalam hal ini, salah satu pendekatannya adalah memigrasikan aplikasi dengan menggunakan strategi yang membutuhkan lebih sedikit usaha, seperti rehost atau replatform. Itu dapat memberikan efisiensi cepat dan pengurangan biaya dalam jangka pendek. Kemudian investasikan kembali tabungan untuk memodernisasi aplikasi pada tahap selanjutnya, dan mencapai pengurangan biaya lebih lanjut.

Namun, dimulai dengan rehost lengkap dari semua aplikasi menunda manfaat modernisasi yang lebih besar. Kuncinya adalah menemukan keseimbangan antara strategi migrasi sehingga aplikasi strategis bisnis diprioritaskan untuk modernisasi sementara aplikasi lain dapat di-host ulang atau direplatform terlebih dahulu kemudian dimodernisasi.

Bagaimana cara menentukan strategi migrasi untuk aplikasi Anda?

Pada tahap penilaian ini, fokusnya adalah untuk menggabungkan model awal untuk memandu pemilihan strategi migrasi. Untuk memvalidasi strategi migrasi untuk aplikasi awal, gunakan model bersama dengan driver bisnis dan kriteria prioritas. Logika default dari pohon keputusan akan membantu Anda menentukan perlakuan awal untuk ruang lingkup. Di pohon, pendekatan yang paling kompleks, seperti refactor, atau arsitek ulang, disediakan untuk beban kerja strategis Anda.


Key business drivers
Cost reduction / avoidance
(Licence renewal, hardware refresh, increased operational efficiency)
Business transformation
(Increased innovation and agility, reduced time to market)
Risk reduction
(Regulatory requirements, end of life H/W, unsupported application, audit compliance)
Identify in-scope applications
Is app being transformed on premises?
Yes
Re-assess application
No
Planned decommission prior to migration deadline?
Yes
No
In scope?
Yes
No
Analyze compute performance information
Peak compute utilization below 3%?
Yes
Analyze application/infrastructure dependencies (i.e., inbound/outbound network communication traffic)
No
Retire?
Yes
No
Revenue generating or strategic application?
Yes
No
Rate of functional change?
Twice or more per year
Once or less per year
Current operational complexity level?
High
Medium / Low
Legacy code base or architecture?
Yes
No
Investment available to refactor (re-architect)?
Yes
No
Can complete prior to migration deadline?
Yes
No
Third-party or in-house solution?
Third-party
In-house
All licenses and products supported in AWS?
No
Assess SaaS alternatives considering benefits, available budget, required effort and migration deadlines
SaaS option available and preferred?
Yes
No
Is upgrade supported or move to AWS managed service possible?
Yes
No
Assess investment and timeline/effort for modernization

Langkah pertama untuk model awal adalah memperbarui driver bisnis di bagian atas pohon dengan yang ditentukan oleh organisasi Anda. Selanjutnya, terapkan pohon ke komponen aplikasi daripada aplikasi secara keseluruhan. Misalnya, dalam kasus aplikasi tiga tingkat yang memiliki tiga komponen (front-end, lapisan aplikasi, dan database), setiap komponen harus transit pohon secara independen dan diberi strategi dan pola tertentu. Ini karena dalam beberapa kasus Anda mungkin ingin meng-host ulang atau memplatform ulang tingkat tertentu dan refactor (arsitek ulang) tingkatan lain.

Sebelum menggunakan pohon keputusan untuk membuat strategi migrasi, uji logika dengan beberapa aplikasi dan konfirmasikan bahwa Anda umumnya setuju dengan hasilnya. Pohon keputusan adalah panduan yang tidak menggantikan analisis yang diperlukan untuk menentukan kebenarannya. Logika pohon mungkin tidak berlaku untuk kasus tertentu. Perlakukan kasus-kasus itu sebagai pengecualian, dan lanjutkan untuk mengganti keputusan yang didorong oleh pohon dengan mendokumentasikan alasan untuk penggantian daripada mengubah logika pohon. Ini mencegah beberapa versi pohon keputusan, yang bisa menjadi sulit untuk dikelola. Panduan umum adalah bahwa pohon harus valid untuk setidaknya 70-80 persen dari beban kerja. Selebihnya, akan ada pengecualian. Pada tahap penilaian ini, setiap penyesuaian pada logika pohon harus difokuskan pada pembentukan model awal untuk memungkinkan perencanaan. Iterasi dan penyempurnaan lebih lanjut terjadi pada tahap selanjutnya, seperti analisis portofolio dan perencanaan migrasi.

Membuat kasus bisnis terarah

Pemangku kepentingan dari seluruh bisnis harus memahami dan membeli ke dalam kasus bisnis untuk transformasi setiap langkah di sepanjang jalan.

Pada tahap awal, penting untuk dengan cepat menunjukkan nilai potensial yang cukup dari program migrasi, sehingga Anda dapat mengamankan sumber daya yang dibutuhkan untuk merencanakan dan membuat program. Kasus bisnis terarah dirancang untuk memberikan kepercayaan yang masuk akal dalam mencapai nilai bisnis yang menarik dengan data terbatas yang dapat dikumpulkan lebih awal.

Setelah program ditetapkan, kasus bisnis dikembangkan lebih lanjut. Kasus terperinci memberikan akurasi yang lebih besar, gambaran yang lebih lengkap tentang nilai program, dan wawasan tentang prioritas perencanaan. Ini mendefinisikan dan mengukur hasil bisnis yang direncanakan yang dibeli organisasi, dan menetapkan dasar di mana kantor tata kelola program Anda kemudian dapat mengarahkan program dan mengukur pencapaiannya.

Memperbaiki ruang lingkup kasus bisnis terarah

Kasus bisnis terarah biasanya dirakit dengan cepat, dalam waktu 2-4 minggu. Ini perlu menghasilkan kepercayaan diri yang cukup sehingga Anda dapat mengamankan sumber daya untuk membentuk tim inti, melibatkan AWS Mitra jika diperlukan, dan seminimal mungkin, menyelesaikan penilaian aplikasi yang diprioritaskan dan analisis portofolio dan tahap perencanaan migrasi.

Biasanya, kasus bisnis terarah yang mendukung migrasi portofolio dibuat sebagai salah satu dari berikut ini:

- Perbandingan biaya kepemilikan total (TCO) sederhana antara lanskap infrastruktur apa adanya dan arsitektur Layanan AWS pasca-migrasi. Perbandingan menunjukkan perbedaan laju lari yang diharapkan untuk volume beban kerja tertentu.
- Kasus bisnis yang menunjukkan nilai sekarang bersih (NPV), laba atas investasi (ROI), periode pengembalian modal, tingkat pengembalian internal yang dimodifikasi (MIRR) dan analisis arus kas 3-5 tahun untuk bermigrasi ke AWS inklusif biaya migrasi vs tetap apa adanya.

Lingkup kasus bisnis terarah biasanya terbatas pada salah satu dari berikut ini:

- Perbandingan biaya teknologi infrastruktur
- Perbandingan teknologi infrastruktur dan biaya operasi

Secara umum, semakin besar portofolio, semakin kurang berkembang kasusnya. Ini karena asumsi yang lebih luas dapat dibuat tanpa mempengaruhi hasilnya secara signifikan. Untuk portofolio yang lebih kecil, setiap perubahan akan memiliki dampak yang lebih besar, sehingga diperlukan lebih banyak detail.

Mulailah dengan membangun perbandingan biaya infrastruktur dasar. Kemudian putuskan apakah perbandingannya cukup menarik sebelum Anda melanjutkan. Biasanya, portofolio lebih dari 400 server akan menunjukkan kasus bisnis yang positif pada pengurangan biaya infrastruktur saja dalam waktu 3 tahun beroperasi AWS, atau 250 server dalam waktu 5 tahun, meskipun ini dapat bervariasi. Untuk portofolio yang lebih kecil, detail lebih lanjut mungkin diperlukan.

Sebaliknya, jarang berguna untuk memeriksa komponen nilai bisnis lainnya pada tahap ini, seperti nilai yang berasal dari peningkatan ketahanan atau kelincahan bisnis, kecuali total ruang lingkup migrasi kurang dari sekitar 5 beban kerja atau 50 server.

Driver nilai fokus

Perbandingan TCO teknologi infrastruktur membandingkan model biaya infrastruktur apa adanya dengan model dasar Layanan AWS tagihan bahan yang dibutuhkan untuk menjalankan beban kerja Anda dengan kinerja dan ketersediaan yang setara. Banyak optimasi dapat dilakukan. Pada tahap ini, bagaimanapun, fokusnya adalah pada daftar berikut karena mereka lebih mudah untuk menilai dan biasanya menghasilkan sekitar 30 persen penghematan TCO, yang cukup untuk bergerak maju:

- Compute elastisitas — Server peta yang penggunaannya tidak 100 persen, seperti server pengembangan atau UAT yang menjalankan 8x5 (penggunaan 24 persen), 10x5 (30 persen), atau 10x6 (36 persen), dan server pemulihan bencana (DR) yang berjalan pada 2 persen, untuk layanan sesuai permintaan yang hanya ditagih saat digunakan.
- Pengadaan dengan rencana penghematan - Rencanakan untuk mendapatkan server produksi dan server lain dengan penggunaan tinggi (lebih dari 36 persen) dengan rencana penghematan yang sesuai untuk mengurangi biaya hingga 75 persen. Opsi termasuk komitmen 1 tahun dan 3 tahun, dengan tingkat pembayaran di muka yang berbeda untuk mendapatkan diskon yang lebih besar.
- Hapus zombie — Identifikasi server dengan pemanfaatan CPU kurang dari 2 persen yang dapat Anda konfirmasi tidak lagi diperlukan, dan hapus dari analisis biaya.
- Hitung ukuran kanan — Gunakan data deret waktu pemanfaatan CPU dan memori untuk menilai setiap server daya komputasi dan memori yang dibutuhkan. Kemudian pilih instans Amazon Elastic Compute Cloud (Amazon EC2) agar sesuai.
- Sistem manajemen basis data relasional (RDBMS) lisensi ukuran kanan - Menilai kembali kebutuhan lisensi RDBMS Anda setelah menghitung ukuran kanan pada server database Anda, bandingkan Bring Your Own License (BYOL) dan lisensi Pengadaan dari, dan jelajahi AWS potensi Amazon Relational Database Service (Amazon RDS) untuk meningkatkan penghematan.
- Penyimpanan — Right-size total volume penyimpanan yang dibutuhkan, dan mengidentifikasi kebutuhan input/output operasi per detik (IOPS) di seluruh portofolio. Tentukan berapa banyak yang dapat dipindahkan ke penyimpanan objek dengan SLA dan biaya yang berbeda.

Kebutuhan data

Tabel dalam Memahami persyaratan data penilaian awal menunjukkan data yang diperlukan untuk membangun setiap bagian dari kasus bisnis terarah, dan apakah itu wajib atau opsional.

Untuk membangun kasus ini, Anda memerlukan subset infrastruktur dari data perencanaan awal ditambah data biaya. Menentukan cara mengidentifikasi infrastruktur yang akan disertakan tergantung pada tujuan bisnis Anda:

- Jika tujuan program adalah untuk memigrasi dan memodernisasi aplikasi tertentu, bangun portofolio infrastruktur berdasarkan apa yang dibutuhkan aplikasi, dengan mempertimbangkan infrastruktur yang dibagikan.
- Jika tujuan program adalah infrastruktur-sentris, seperti migrasi keluar dari pusat data yang sewanya akan kedaluwarsa, pemetaan aplikasi tidak diperlukan untuk perbandingan TCO infrastruktur.

Data yang ditandai sebagai opsional (seperti CPU dan pemanfaatan puncak memori untuk server) biasanya dapat diganti dengan nilai benchmark standar. Anda dapat mendiskusikan hal ini dengan AWS Mitra atau Layanan AWS Profesional. Atau Anda dapat mengekstrapolasi nilai dari titik data yang tersedia di bagian portofolio Anda (seperti data yang dikumpulkan oleh hypervisor). Semakin besar portofolio, semakin akurat ini.

Membangun infrastruktur perbandingan TCO

Alat sangat penting untuk membangun perbandingan TCO infrastruktur. AWS Layanan Profesional atau AWS Mitra dapat memberikan bantuan dengan semua jenis kasus terarah, terutama jika Anda berencana untuk melibatkan mereka untuk membantu dalam proses migrasi yang lebih luas.

Ada alat yang tersedia untuk melakukan hal berikut:

- Kumpulkan data inventaris.
- Kumpulkan data pemanfaatan.
- Menyediakan data benchmarking biaya infrastruktur yang akurat.
- Identifikasi dan hapus zombie.
- Buat penilaian ukuran yang tepat.
- Merekomendasikan opsi pembelian.
- Bandingkan opsi lisensi perangkat lunak.
- Menghasilkan analisis arus kas grafis sederhana.

Migration Evaluator dari AWS adalah salah satu opsi. Ini menyediakan semua kemampuan ini sebagai layanan terkelola gratis. Anda dapat meminta Penilai Migrasi melalui manajer AWS akun

atau Mitra Kompetensi AWS Migrasi atau dengan mengirimkan permintaan secara online. Migration Evaluator telah dirancang khusus sebagai solusi titik untuk menghasilkan perbandingan TCO teknologi infrastruktur dan cepat.

Keuntungan utama:

- Bebas biaya
- Agent-less penemuan atau konfigurasi manual data inventaris di mana penemuan berbasis alat dibatasi
- Dukungan khusus untuk membantu penyebaran, konfigurasi, pengumpulan data, dan membangun kasus dasar, atau kasus bisnis terarah
- Kenyamanan operasi SaaS, tetapi dapat menjalankan pengumpulan data sepenuhnya dalam jaringan pelanggan untuk mendukung scrubbing sebelum memuat ke mesin analitik
- Dukungan kuat untuk ukuran kanan lisensi Microsoft
- Kemampuan ekspor data lengkap

Keterbatasan utama:

- Menilai hanya server arsitektur x86 (Windows dan Linux)
- Opsi terbatas untuk mengonfigurasi atau mengkalibrasi data biaya apa adanya benchmark
- Tidak ada dukungan untuk pengoptimalan biaya operasi pemodelan
- Tidak ada dukungan untuk pemodelan biaya migrasi
- Tidak ada dukungan langsung untuk membangun kasus bisnis di luar perbandingan TCO

Jika Anda memutuskan untuk menggunakan alat penemuan komersial untuk penemuan portofolio dan kemampuan analisis seperti tumpukan aplikasi dan penemuan interdependensi, biasanya akan memberikan perbandingan TCO infrastruktur juga. Untuk panduan tentang penggunaan alat untuk penemuan dan penilaian portofolio, lihat Mengevaluasi kebutuhan alat penemuan. Untuk meninjau dan membandingkan kemampuan utama alat terdepan di pasar, lihat Alat migrasi Discovery, Planning, dan Rekomendasi.

Membangun optimalisasi biaya operasional

Peningkatan produktivitas operasi TI seringkali merupakan kontributor nilai yang signifikan untuk migrasi. Rata-rata, setelah migrasi ke AWS, produktivitas staf operasional TI meningkat sebesar 62

persen melalui migrasi, menurut whitepaper International Data Corporation (IDC) Fostering Business and Organizational Transformation to Generate Business Value with Amazon Web Services. Namun, ada dua tantangan dengan ukuran dan termasuk manfaat ini dalam kasus terarah.

Pertama, menilai berbagai keuntungan produktivitas membutuhkan pengumpulan data yang ekstensif dan lebih tepat untuk kasus bisnis yang terperinci. Tantangan ini dapat diatasi dengan berfokus pada beberapa elemen yang lebih mudah dinilai dan diukur dengan data benchmark sederhana tetapi masih menunjukkan keuntungan yang signifikan.

Kedua, berfokus pada produktivitas sebagai sumber pengurangan biaya dapat menimbulkan kekhawatiran dan negativitas di antara pemangku kepentingan pelanggan utama dan anggota program. Pastikan bahwa Anda memberikan kejelasan tentang bagaimana manfaat akan direalisasikan dan apa artinya bagi orang-orang yang terkena dampak. Masalah seperti itu dapat dihindari dengan mengklarifikasi bahwa ini hanya akan meningkatkan peran tim:

- Program migrasi mencakup jalur untuk mengembangkan dan memindahkan staf operasi internal ke peran baru, seperti bergabung dengan DevSecOps tim yang membangun infrastruktur sebagai otomatisasi kode dan otomatisasi pengujian yang akan mendorong pertumbuhan tim.
- Manfaat dapat direalisasikan dengan mengubah dan mengubah ukuran kontrak outsourcing operasi, sehingga staf internal dapat meningkatkan fokus mereka pada kegiatan yang bernilai lebih tinggi

Pendekatan membangun elemen kasus bisnis ini berdasarkan transformasi operasi apa yang ingin Anda pertimbangkan:

- Jika Anda memiliki tim operasi internal yang ada, tingkatkan keterampilan anggota tim, dan tunjukkan peningkatan produktivitas yang diharapkan.
- Atau, bermigrasi dari solusi operasi Anda saat ini ke AWS Managed Services (AMS) atau ke penawaran layanan terkelola alternatif dari AWS Mitra.

Untuk transformasi pertama, untuk mendapatkan perkiraan keuangan konservatif dari peningkatan produktivitas yang dapat dimasukkan dalam kasus ini, kami merekomendasikan yang berikut:

1. Fokus pada produktivitas operasi manajemen server secara khusus. Ini cenderung menjadi proporsi yang signifikan dari upaya operasi, dapat lebih mudah dinilai, dan lebih mudah diverifikasi nanti.

2. Hitung kepegawaian yang dibutuhkan berdasarkan tolok ukur jumlah server yang dapat dikelola oleh setiap karyawan full-time equivalent (FTE). Di tempat, jumlah itu sekitar 150 severs. Pada AWS, itu sekitar 400 server.
3. Terapkan metrik ini ke jumlah server lokal dibandingkan dengan jumlah instans EC2.
4. Lipat gandakan waktu yang dihemat dengan tingkat biaya campuran untuk seluruh tim operasi.

Anda kemudian dapat memeriksa hasil Anda dengan salah satu pendekatan dengan memverifikasi hasilnya tidak jauh melebihi perolehan produktivitas rata-rata berdasarkan peran yang disediakan dalam tabel berikut (data yang bersumber dari whitepaper IDC Mendorong Bisnis dan Transformasi Organisasi untuk Menghasilkan Nilai Bisnis dengan Amazon Web Services.

Peran	Keuntungan efisiensi
Manajemen infrastruktur TI	62%
Dukungan TI	59%
Manajemen aplikasi	43%
Manajemen basis data	19%
Pengembangan aplikasi	25%

Untuk transformasi kedua, Anda dapat menambahkan penghematan biaya operasional dengan langsung membandingkan total operasi saat ini dan biaya dukungan untuk portofolio dalam ruang lingkup dengan biaya layanan terkelola yang dipertimbangkan.

Untuk mendapatkan biaya layanan terkelola, berikan kepada manajer AWS akun atau AWS Managed Services Mitra Anda dengan AWS Bill of Material yang Anda usulkan, pilihan tingkat layanan Anda (Plus atau Premium), dan paket AMS Anda (AMS Accelerate atau AMS Advanced). Ini akan memberi Anda total biaya layanan terkelola untuk Layanan AWS komponen solusi yang diubah. Demikian pula, Anda dapat memperoleh harga dari AWS Mitra yang menawarkan paket layanan terkelola sendiri berdasarkan parameternya sendiri.

Memperluas ke kasus bisnis terarah penuh

Secara umum, untuk merakit kasus bisnis terarah penuh, membangun perbandingan TCO, dengan atau tanpa elemen produktivitas TI, dan perkirakan semua biaya migrasi dan modernisasi. Kemudian buat arus kas yang mencakup pasangan skenario migrasi-dan-modernisasi dan jangan bermigrasi-dan-modernisasi.

Kasus yang paling mendasar adalah persiapan sepasang skenario, di mana skenario jangan bermigrasi-dan-modernisasi adalah situasi Anda saat ini dan skenario migrasi-dan-modernisasi memiliki karakteristik sebagai berikut:

- Tidak ada pertumbuhan atau penyusutan volume transaksional, komputasi, atau kapasitas jaringan
- Pertumbuhan volume rendah yang stabil dalam persyaratan penyimpanan
- Quality-of-service kemampuan (seperti ketersediaan, daya tahan, throughput, dan kinerja) yang sesuai dengan kemampuan sistem yang ada

Untuk semua kecuali portofolio yang sangat kecil, ini sesuai dengan tujuan membangun kasus terarah dengan baik. Ini menunjukkan nilai yang cukup cepat untuk mendapatkan mandat untuk bergerak maju.

Untuk portofolio yang lebih kecil, akan bermanfaat untuk menambahkan pasangan skenario migrasi-dan-modernisasi dan jangan bermigrasi dan memodernisasi yang menunjukkan aspek lain dari peningkatan nilai migrasi cloud, seperti:

- Campuran persyaratan pertumbuhan kapasitas sedang dan tinggi di seluruh beban kerja di mana pertumbuhan itu diharapkan
- Dimasukkannya ketahanan yang ditingkatkan, seperti ketersediaan tinggi, DR, dan toleransi kesalahan
- Peningkatan kinerja global dengan komputasi tepi, jaringan pengiriman konten (CDN), replikasi database multi-wilayah.
- Kualitas layanan spesifik lainnya yang telah Anda jadikan prioritas bisnis untuk program ini

Untuk skenario ini, pastikan bahwa biaya dan implikasi arus kas dari peningkatan arsitektur infrastruktur non-cloud saat ini agar sesuai dengan spesifikasi baru diperkirakan secara akurat. Cara paling langsung untuk mendapatkan perkiraan ini dapat meminta kutipan dari integrator sistem, terutama jika mereka juga merupakan Mitra AWS Konsultasi dengan Kompetensi Migrasi, yang dapat mendukung Anda dengan skenario migrasi-dan-modernisasi dan larangan migrasi-dan-modernisasi.

Untuk setiap pasangan skenario, kumpulkan kasus yang terdiri dari hal-hal berikut:

- Biaya skenario jangan bermigrasi-dan-modernisasi. Dalam kasus yang paling mendasar, ini termasuk:
  - Total biaya kepemilikan selama jangka waktu kasus bisnis untuk konfigurasi infrastruktur saat ini
  - Peningkatan berkala dalam komputasi, penyimpanan, dan konsumsi lalu lintas jaringan
- Biaya migrasi dan modernisasi; skenario, termasuk:
  - Menyiapkan program, yang mencakup penemuan terperinci, perencanaan migrasi, pengembangan kasus bisnis terperinci, pembentukan tim inti dan peningkatan keterampilan mereka, membangun landing zone jika belum ada, dan membangun manajemen keamanan dan integrasi operasi untuk beban kerja yang bermigrasi
  - Biaya migrasi dan modernisasi beban kerja
  - Biaya infrastruktur migrasi, termasuk koneksi jaringan, layanan migrasi data seperti AWS Snowball Edgedan AWS DataSync, dan biaya AWS utilitas untuk arsitektur yang diperlukan selama proses migrasi itu sendiri (misalnya, untuk pengujian)
  - Peningkatan biaya AWS utilitas selama migrasi saat gelombang ditayangkan, dan menurunnya biaya infrastruktur yang ada karena digantikan oleh layanan AWS berbasis dan dinonaktifkan
- Biaya penonaktifan dan penghapusan untuk setiap aset yang terdampar

Memperkirakan pengaturan program migrasi dan modernisasi

Untuk menyiapkan program untuk sukses, Anda mungkin perlu menjalankan serangkaian kegiatan dasar untuk membangun kemampuan dasar dan rencana terperinci jika ini belum dilakukan sebelumnya. Kegiatan dasar ini meliputi:

1. Melakukan penemuan portofolio terperinci, perencanaan migrasi, dan pengembangan kasus bisnis terperinci, seperti yang dijelaskan di bagian Analisis portofolio dan perencanaan migrasi, ditambah pendokumentasian biaya alat penemuan apa pun yang digunakan.
2. Membangun bisnis cloud dan tim inti teknis dan mengembangkan keterampilan internal melalui pelatihan dan perekrutan. Identifikasi anggota organisasi TI yang akan membutuhkan pelatihan, dan alokasikan anggaran pelatihan untuk setiap orang.
3. Menetapkan landing zone dan mengonfigurasinya untuk mendukung kemampuan tata kelola biaya, operasional, dan keamanan yang Anda perlukan.

AWS Mitra Konsultasi dapat membantu memberikan perkiraan untuk item 1 dan 3.

Memperkirakan biaya migrasi dan modernisasi

Untuk memenuhi tujuan kasus bisnis terarah dan menunjukkan potensi komersial yang cukup untuk melanjutkan ke fase berikutnya, pertahankan estimasi biaya migrasi dan modernisasi sedasar mungkin.

Untuk tujuan ini, kami menyarankan Anda menyiapkan kasus bisnis terarah dengan berfokus pada aplikasi yang termasuk dalam strategi migrasi berikut:

- Pensiun
- Mempertahankan
- Pindah
- Rehost
- Platform Ulang
- Pembelian kembali

Biasanya, sekitar 70 persen beban kerja dapat direhost, dipindahkan atau direplatform, dan 5 persen lainnya dapat dihentikan. Menilai aplikasi dengan strategi migrasi biasanya membahas inti dari kasus pengurangan biaya.

Memperkirakan biaya untuk refactoring, atau re-architecting, bisa jadi rumit. Tidak praktis untuk mencoba ini dalam jangka waktu yang diberikan untuk mempersiapkan kasus bisnis terarah. Seperti yang dibahas sebelumnya dalam Menentukan tipe R untuk migrasi, pertimbangkan untuk menggunakan strategi rehost, relokasi, atau replatform untuk fase pertama migrasi dan modernisasi Anda. Strategi R ini kemungkinan akan mempercepat pengembalian awal, mengurangi risiko implementasi, dan meningkatkan kasus bisnis dalam jangka pendek. Hal ini juga secara material lebih mudah bagi tim aplikasi Anda untuk memodernisasi aplikasi yang berjalan dalam AWS lingkungan daripada yang tidak. Perkiraan untuk refactoring (re-architecting) aplikasi spesifik paling baik ditambahkan ketika kasus bisnis terperinci disiapkan.

Memperkirakan upaya migrasi dengan strategi

Setiap migrasi berbeda. Sebelum melakukan anggaran atau rencana apa pun, perkiraan beban kerja benih untuk aktivitas migrasi dari tim yang akan bertanggung jawab atas proyek, apakah itu tim aplikasi internal Anda, Layanan AWS Profesional, atau organisasi Mitra. AWS

Untuk membantu membangun kasus terarah, tabel berikut memberikan rentang upaya indikatif untuk perawatan yang berbeda. Rentang ini mengasumsikan bahwa portofolio menengah hingga besar

sedang dimigrasikan dan tim migrasi dilatih dan berpengalaman. Untuk portofolio kecil, yang terbaik adalah meminta tim yang bertanggung jawab atas migrasi menyiapkan perkiraan bahkan untuk kasus terarah.

Strategi migrasi	Proses estimasi	Elemen	Jam orang	Jam orang
Mempertahankan	Jangan melakukan apa-apa, tanpa biaya, tanpa manfaat, dan tidak ada pengurangan hutang teknologi.	–	–	–
Pensiun	Perkiraan penonaktifan peralatan perangkat keras yang digunakan, jika ada.	–	–	–
Pindah	Perkiraan menyalin beban kerja dalam VMware menggunakan alat VMware. Ini termasuk menyalin data, pengujian asap untuk memverifikasi, dan penonaktifan perangkat keras apa pun. Upaya untuk merelokasi VM biasanya	–	–	–

	kurang dari pola rehost dengan kompleksitas rendah.

Rehost	Perkirakan penyalinan beban kerja dan data dengan salinan gambar, pengujian asap, pengujian ketersediaan tinggi (HA) dan pemulihan bencana (DR) jika sesuai untuk server produksi, dan penonaktifan perangkat keras apa pun. Praktik terbaik adalah menggunak an alat-alat seperti AWS Transform MGN. Bagilah beban kerja menjadi kompleksitas rendah, sedang, dan tinggi, berdasarkan faktor-faktor seperti apakah database atau perangkat lunak infrastruktur lainnya berjalan, kompleksi tas basis data, apakah	Upaya per aplikasi per server	Migrasi	HA/DR tes
		Rendah	10—14	3—5
		Sedang	16—24	4—6
		Tinggi	26—38	8—12

	dikelompokkan, kompleksitas integrasi, dan volume data.

Platform Ulang	Untuk migrasi replatform yang mencakup peningkatan ke sistem operasi atau versi RDBMS, ambil perkiraan untuk rehost, dan tambahkan waktu untuk menjalankan uji rebuild dan asap pada platform baru. Jika replatform termasuk mengubah teknologi platform, perkirakan waktu tambahan untuk penggunaan alat konversi, seperti AWS Schema Conversion Tooldan AWS Database Migration Service, dan tes aplikasi yang lebih lengkap. Contoh perubahan teknologi adalah bermigrasi	Upaya per aplikasi per server	Versi naik	Perubahan teknologi
		Rendah	Tambahkan 1—3	Tambahkan 10—15
		Sedang	Tambahkan 2—5	Tambahkan 20—30
		Tinggi	Tambahkan 4-8	Tambahkan 40—60

	dari database komersial berpemilik ke pengganti open source.			
Pembelian kembali	Perkirakan ekstraksi data, transformasi, dan pengungga han ke dalam penggantian layanan SaaS yang baru dibeli, dan penonaktifan perangkat keras apa pun.	–	–	–

Memperkirakan biaya infrastruktur migrasi

Sertakan perkiraan untuk infrastruktur yang akan Anda gunakan selama migrasi. Biasanya, perkiraan ini terdiri dari:

- Anggaran untuk konektivitas dan layanan pertukaran data untuk beban kerja dan migrasi data dari lingkungan saat ini ke AWS
- Anggaran untuk Layanan AWS (terutama komputasi dan penyimpanan) yang diperlukan untuk menghosting beban kerja yang dimigrasi selama proses migrasi, pengujian, dan pemotongan
- Peningkatan biaya AWS utilitas karena setiap gelombang migrasi selesai
- Biaya penonaktifan infrastruktur yang ada yang tidak akan lagi menjalankan beban kerja yang dimigrasi

Untuk pertukaran data, periksa total volume data Anda dan nilai kelayakan menggunakan jaringan. Jika Anda telah menyediakan AWS Direct Connecttautan atau Site-to-Site VPNdari AWS satu titik di WAN Anda sebelumnya untuk penggunaan operasional setelah migrasi, Anda dapat menggunakan sumber daya tersebut hingga kuota layanannya.

Jika kapasitas jaringan Anda tidak mencukupi, peningkatan bandwidth internet jangka pendek dengan jaringan pribadi virtual (VPN) seringkali merupakan solusi yang sangat hemat biaya. Jika tidak, perangkat pertukaran AWS media seperti AWS Snowball Edgedan AWS Snowball Edgemenawarkan solusi di sebagian besar Wilayah AWS. Juga, untuk migrasi data volume sangat tinggi, pertimbangkan untuk memasukkan anggaran untuk AWS DataSync, yang meningkatkan keandalan dan dapat mempercepat transfer terlepas dari media yang digunakan.

Pemodelan ramp up Layanan AWS dan ramp down infrastruktur yang ada penting untuk elemen analisis arus kas dari kasus bisnis. Pada tahap ini, Anda tidak mungkin memiliki rencana gelombang untuk menentukan dengan tepat kapan biaya akan dikeluarkan. Sebaiknya lakukan hal berikut:

- Meningkatkan biaya untuk AWS pada tingkat konstan selama migrasi.
- Mengurangi biaya untuk infrastruktur yang ada yang Anda rencanakan untuk dinonaktifkan pada tingkat konstan selama durasi yang sama.

Memulai kenaikan AWS biaya 1-2 bulan sebelum infrastruktur yang ada turun. Ini menyediakan 1 bulan penggunaan AWS utilitas untuk melakukan migrasi untuk setiap gelombang. Ini termasuk waktu untuk pengujian, dan waktu tambahan untuk menyelesaikan pekerjaan penonaktifan yang diperlukan untuk menghentikan biaya pada infrastruktur yang diganti.

Memperkirakan biaya penonaktifan

Menonaktifkan peralatan yang tidak dapat dikerahkan kembali, dan membuangnya dengan cara yang legal dan ramah lingkungan, dapat menimbulkan biaya kecil. Namun, untuk kasus bisnis terarah, biasanya satu-satunya jumlah material yang berpotensi adalah biaya penghapusan nilai buku yang tersisa dari aset yang diganti.

Untuk kasus bisnis terarah, kami sarankan Anda melakukan hal berikut:

- Tinjau daftar aset Anda.
- Identifikasi mereka yang akan dinonaktifkan.
- Untuk mengurangi penghapusan, periksa peluang untuk beralih perangkat sehingga perangkat yang lebih baru dalam daftar dapat digunakan untuk menggantikan aset yang lebih tua dan lebih terdepresiasi sepenuhnya.
- Buat penilaian nilai buku future dari aset yang akan dinonaktifkan pada saat itu.
- Sertakan ini sebagai biaya migrasi penonaktifan.

Merakit dan menyesuaikan kasus bisnis terarah penuh

Setelah Anda menyiapkan set lengkap biaya untuk setiap pasangan skenario, buatlah laporan arus kas diskon untuk masing-masing dan buat grafik. Kami merekomendasikan untuk membangun kasus bisnis terarah selama periode yang sama dengan siklus penyegaran perangkat keras. Ini biasanya 5 tahun untuk server, penyimpanan, dan perangkat jaringan. Saat Anda menggunakan periode yang sama dengan siklus penyegaran perangkat keras, biaya tepat satu penyegaran disertakan dalam biaya apa adanya untuk setiap skenario.

Kemudian hitung metrik keuangan utama yang Anda butuhkan untuk mendapatkan persetujuan untuk pindah ke fase berikutnya dari program. Kami biasanya menyertakan yang berikut:

- Net present value (NPV) untuk mengukur nilai absolut dari pengurangan biaya dan keuntungan produktivitas yang dinilai
- Periode pengembalian modal dalam beberapa bulan untuk memverifikasi bahwa pengembalian cukup cepat
- Perbandingan run-rate akhir untuk memverifikasi apakah proses tersebut mengeluarkan biaya yang cukup selama jangka waktu tersebut
- Pengembalian investasi (ROI) dan tingkat pengembalian investasi yang dimodifikasi (MIRR) untuk menilai kinerja keuangan relatif dari program di atas tuntutan modal lain yang mungkin diprioritaskan oleh organisasi Anda

Gunakan iterasi pertama dari kasus ini untuk menentukan apakah kinerja keuangan yang diharapkan berarti bahwa penyempurnaan harus dilakukan, seperti dalam contoh berikut:

- Jika pengembalian terlalu lambat, pertimbangkan opsi untuk mempercepat dan mengurangi biaya migrasi, seperti berikut ini:
  - Gunakan AWS Mitra atau Layanan AWS Profesional untuk memperluas sumber daya yang tersedia dan selanjutnya memparalelkan beban kerja migrasi dengan pola yang lebih mendasar.
  - Untuk beban kerja yang berjalan di VMware, bandingkan strategi relokasi dengan strategi rehost atau replatform, setidaknya untuk fase awal. Menggunakan strategi relokasi dapat mengurangi biaya migrasi dan meningkatkan kecepatan migrasi.
  - Jika layak secara teknis, dorong beban kerja yang membutuhkan strategi replatform atau refactor (re-architect) yang lebih kompleks ke fase future, di luar lingkup kasus bisnis awal.
- Jika ROI dan MIRR terlalu rendah, pertimbangkan hal berikut:

    - Apakah skenario yang Anda anggap terlalu konservatif? Apakah Anda memiliki skenario yang mencerminkan pertumbuhan kapasitas dan kebutuhan elastisitas yang paling mungkin? Apakah Anda memiliki skenario yang membandingkan biaya termasuk peningkatan kualitas layanan dalam tujuan Anda?
    - Dapatkah Anda menyempurnakan cakupan portofolio aplikasi yang akan dimigrasikan pada tahap pertama untuk fokus pada beban kerja yang akan menghasilkan pengembalian yang lebih kuat, seperti yang memiliki pemanfaatan arus yang lebih rendah atau kebutuhan pemulihan bencana (DR) yang mahal?
    - Dapatkah menyempurnakan ruang lingkup portofolio aplikasi untuk awalnya mengecualikan beban kerja tertentu yang mencapai kurang komersial? Misalnya, dapatkah Anda menunda beban kerja yang lisensi perangkat lunak pihak ketiga menjadi lebih mahal karena persyaratan yang berbeda untuk penyebaran di infrastruktur cloud publik?
- Jika perbandingan run-rate akhir tidak memenuhi target yang diharapkan, jelajahi hal berikut:
    - Pertama, konfirmasikan bahwa metrik lainnya memenuhi harapan. Kasus bisnis terarah terutama untuk menunjukkan bahwa ada peluang keuangan yang cukup untuk membenarkan memulai fase persiapan migrasi berikutnya.
    - Identifikasi daftar peluang untuk terus meningkatkan kinerja biaya AWS setelah fase awal migrasi.

Sertakan penilaian daftar peluang saat menyiapkan kasus bisnis terperinci. Selain itu, sertakan penilaian peluang dalam pemeliharaan kasus yang sedang berlangsung dan proses pengoptimalan biaya bulanan setelah migrasi selesai.

Penilaian aplikasi yang diprioritaskan

Salah satu hasil utama dari tahap sebelumnya, penemuan portofolio dan perencanaan awal, adalah memprioritaskan subset aplikasi untuk penilaian terperinci. Bagian ini mengeksplorasi penilaian rinci aplikasi.

Melihat detail beberapa aplikasi sejak dini akan mendorong akselerasi. Proses penilaian dan desain arsitektur calon memunculkan pemblokir potensial dan mengklarifikasi tugas-tugas penting yang merupakan prekursor migrasi dengan cakupan yang lebih besar. Tugas-tugas ini termasuk mengumpulkan persyaratan untuk membangun AWS fondasi, seperti landing zone on AWS, atau untuk memperluas dan memvalidasi landing zone yang ada. Penilaian ini juga merupakan waktu untuk mempertimbangkan langkah-langkah dan strategi migrasi.

Hasil utama dari tahap ini adalah sebagai berikut:

- Daftar aplikasi yang diprioritaskan yang divalidasi
- Arsitektur keadaan saat ini yang didokumentasikan
- Arsitektur target terdokumentasi dan strategi migrasi untuk kandidat migrasi
- Pola dan perkakas migrasi yang teridentifikasi
- Persyaratan platform yang terdokumentasi (keamanan, AWS infrastruktur, dan operasi)
- Pertimbangan cutover terdokumentasi untuk perencanaan migrasi
- Perkiraan laju AWS lari

Memahami persyaratan data penilaian terperinci

Tabel berikut menjelaskan informasi yang diperlukan untuk mendapatkan tampilan portofolio lengkap dari aplikasi dalam migrasi dan infrastruktur terkait.

Tabel menggunakan singkatan berikut:

- R, untuk yang dibutuhkan
- O, untuk opsional
- N/A, karena tidak berlaku

Aplikasi

Nama atribut	Deskripsi	Penemuan, desain, dan strategi migrasi	Perkiraan laju lari	Tingkat kesetiaan yang disarankan (minimum)
Pengidentifikasi unik	Misalnya, ID aplikasi. Biasanya tersedia pada CMDB yang ada atau inventaris internal dan sistem kontrol lainnya. Pertimbangkan untuk membuat ID unik setiap kali ini tidak ditentukan dalam organisasi Anda.	R	O	Tinggi
Nama aplikasi	Nama yang dengannya aplikasi ini diketahui organisasi Anda. Sertakan vendor komersial off-the-shelf (COTS) dan nama produk bila berlaku.	R	R	Tinggi
Apakah COTS?	Ya atau Tidak. Apakah	R	R	Tinggi

	ini aplikasi komersial atau pengembangan internal			
Produk dan versi COTS	Nama dan versi produk perangkat lunak komersial	R	R	Tinggi
Deskripsi	Fungsi dan konteks aplikasi utama	R	O	Tinggi
Kekritisan	Misalnya, aplikasi strategis atau menghasilkan pendapatan, atau mendukung fungsi kritis	R	O	Tinggi
Tipe	Misalnya, database, manajemen hubungan pelanggan (CRM), aplikasi web, multimedia, layanan bersama TI	R	O	Tinggi
Lingkungan	Misalnya, produksi, pra-produksi, pengembangan, pengujian, kotak pasir	R	R	Tinggi

Kepatuhan dan peraturan	Kerangka kerja yang berlaku untuk beban kerja (misalnya , HIPAA, SOX,, ISO, SOC PCI-DSS, FedRAMP) dan persyaratan peraturan	R	O	Tinggi
Dependensi	Dependensi hulu dan hilir ke aplikasi atau layanan internal dan eksternal	R	N/A	Tinggi
Pemetaan infrastruktur	Pemetaan ke aset and/or virtual fisik yang membentuk aplikasi	R	R	Tinggi
Lisensi	Jenis lisensi perangkat lunak komoditas (misalnya, Microsoft SQL Server Enterpris e)	R	R	Tinggi
Biaya	Biaya untuk lisensi perangkat lunak, operasi perangkat lunak, dan pemelihar aan	O	R	Medium-high

Unit bisnis	Misalnya, pemasaran , keuangan, penjualan	R	O	Tinggi
Detail pemilik	Informasi kontak untuk pemilik aplikasi	R	O	Tinggi
Jenis arsitektur	Misalnya, aplikasi web, 2-tier, 3-tier, microservices, service-oriented architecture (SOA)	R	R	Tinggi
Tujuan titik pemulihan (RPO), tujuan waktu pemulihan (RTO), dan/ service-level agreement (SLA)	Atribut manajemen layanan saat ini	R	R	Tinggi
Revenue-generating Aplikasi atau aplikasi strategis bisnis?	Ya, jika aplikasi secara langsung atau tidak langsung mempengar uhi pendapata n perusahaan atau dianggap strategis oleh bisnis.	R	O	Medium-high

Jumlah pengguna (bersamaan)	Misalnya, pengguna internal, atau eksternal atau, and/or eksternal internal users/ customers	R	R	Medium-high
Lokasi pengguna	Asal sesi pengguna	R	R	Medium-high
Risiko dan masalah	Risiko dan masalah yang diketahui	R	O	Medium-high
Pertimbangan migrasi	Informasi tambahan apa pun yang mungkin relevan untuk migrasi	R	R	Medium-high
Strategi migrasi	Misalnya, salah satu dari AWS 6 Rs untuk migrasi	R	R	Medium-high
Rincian basis data	Misalnya, partisi, enkripsi, replikasi , ekstensi, dukungan Secure Sockets Layer (SSL)	R	R	Tinggi
Tim Support	Misalnya, nama tim operasi aplikasi	R	O	Medium-high

Solusi pemantauan	Produk yang digunakan untuk memantau aplikasi ini	R	O	Medium-high
Persyaratan Backup	Jadwal pencadangan yang diperlukan di AWS	R	R	Medium-high
Informasi DR	Misalnya, komponen pemulihan bencana untuk aplikasi ini	R	R	Medium-high
AWS Persyaratan target	Misalnya, komponen, penempatan akun, jaringan, keamanan	R	R	Tinggi

Infrastruktur

Nama Atribut	Deskripsi	Penemuan, desain, dan strategi migrasi	Perkiraan laju lari	Tingkat kesetiaan yang disarankan (minimum)
Pengidentifikasi unik	Misalnya, ID server. Biasanya tersedia pada CMDB yang ada atau inventaris internal dan sistem	R	O	Tinggi

	kontrol lainnya. Pertimbangkan untuk membuat ID unik setiap kali ini tidak ditentukan dalam organisasi Anda.			
Nama jaringan	Nama aset dalam jaringan (misalnya, nama host)	R	O	Tinggi
Nama DNS (nama domain yang sepenuhnya memenuhi syarat, atau FQDN)	Nama DNS	R	O	Medium-high
Alamat IP dan netmask	Alamat IP and/or publik internal	R	R	Tinggi
Jenis aset	Misalnya, server fisik atau virtual, hypervisor, wadah, perangkat, contoh database	R	R	Tinggi
Nama produk	Vendor komersial dan nama produk (misalnya, VMware ESXi, IBM Power Systems, Exadata)	R	R	Tinggi

Sistem operasi	Misalnya, REHL 8, Windows Server 2019, AIX 6.1	R	R	Tinggi
Konfigurasi	CPU yang dialokasikan, jumlah inti, utas per inti, total memori, penyimpanan, kartu jaringan	R	R	Tinggi
Pemanfaatan	CPU, memori, dan puncak penyimpanan dan rata-rata . Database instance throughput.	R	R	Tinggi
Lisensi	Jenis lisensi komoditas (misalnya, Standar RHEL)	R	R	Tinggi
Pemetaan aplikasi	Aplikasi atau komponen aplikasi yang berjalan di infrastruktur ini	R	O	Tinggi
Data komunikasi	Misalnya, server ke server pada tingkat proses	R	N/A	Medium-high

AWS Persyaratan target	Misalnya, jenis contoh, akun, subnet, grup keamanan, perutean	R	R	Tinggi
Strategi, pola, dan alat migrasi	Misalnya, salah satu dari 6 Rs untuk migrasi, pola teknis tertentu, perkakas migrasi	R	O	Tinggi
Risiko dan masalah	Risiko dan masalah yang diketahui	R	O	Medium-high

Penilaian aplikasi terperinci

Tujuan dari penilaian aplikasi terperinci adalah pemahaman lengkap tentang aplikasi yang ditargetkan dan infrastruktur terkait (komputasi, penyimpanan, dan jaringan). High-fidelity data diperlukan untuk menghindari jebakan. Misalnya, adalah umum bagi organisasi untuk berasumsi bahwa mereka sepenuhnya memahami aplikasi. Ini wajar, dan itu benar dalam banyak kasus. Namun, untuk meminimalkan risiko terhadap bisnis, penting untuk memvalidasi pengetahuan kelembagaan dan dokumentasi statis dengan memperoleh data terprogram sebanyak mungkin. Ini akan menangani proses penemuan yang berat. Anda dapat fokus pada elemen data yang berasal dari sumber alternatif, seperti informasi spesifik bisnis, peta jalan strategis, dan lain-lain.

Kuncinya adalah menghindari perubahan menit terakhir selama dan setelah migrasi. Misalnya, saat bermigrasi, penting untuk meminimalkan perubahan berdasarkan dependensi tak dikenal yang mungkin memerlukan penyertaan server ke dalam gelombang migrasi yang sedang berlangsung. Tak lama setelah migrasi, penting untuk meminimalkan perubahan berdasarkan persyaratan platform terkait untuk memungkinkan lalu lintas atau menyebarkan layanan tambahan. Perubahan yang tidak direncanakan ini meningkatkan risiko masalah keamanan dan operasional. Kami sangat merekomendasikan penggunaan alat penemuan terprogram untuk memvalidasi pola lalu lintas dan dependensi saat melakukan penilaian aplikasi terperinci.

Di awal penilaian, Anda harus mengidentifikasi pemangku kepentingan aplikasi. Ini biasanya sebagai berikut:

- Unit bisnis memimpin
- Pemilik aplikasi
- Arsitek
- Operasi dan dukungan
- Cloud-enablement tim
- Tim platform tertentu seperti komputasi, penyimpanan, dan jaringan

Ada dua pendekatan untuk penemuan terperinci. Top-down penemuan dimulai dengan aplikasi, atau bahkan dengan pengguna, dan berjalan sampai ke infrastruktur. Ini adalah pendekatan yang disarankan ketika identifikasi aplikasi jelas. Sebaliknya, penemuan bottom-up dimulai dengan infrastruktur dan berjalan sampai ke aplikasi atau layanan dan penggunanya. Pendekatan ini berguna ketika program migrasi didorong oleh tim infrastruktur dan ketika pemetaan aplikasi-ke-infrastruktur tidak jelas. Secara umum, Anda cenderung menggunakan kombinasi keduanya.

Untuk menyelam jauh ke dalam aplikasi, diagram arsitektur yang ada adalah awal yang baik. Jika ini tidak tersedia, buat satu berdasarkan pengetahuan saat ini. Jangan meremehkan pentingnya tugas ini, bahkan untuk strategi migrasi rehost sederhana. Merencanakan diagram arsitektur membantu Anda mengidentifikasi inefisiensi yang dapat dengan cepat diatasi dengan perubahan kecil saat berada di cloud.

Bergantung pada apakah Anda melakukan pendekatan top-down atau bottom-up, diagram awal akan memplot komponen aplikasi dan layanan atau komponen infrastruktur seperti server dan penyeimbang beban. Setelah komponen dan antarmuka utama diidentifikasi, validasi dengan data terprogram dari alat penemuan dan alat pemantauan kinerja aplikasi. Alat harus mendukung analisis ketergantungan dan memberikan informasi komunikasi antar komponen. Setiap komponen yang membentuk aplikasi ini harus diidentifikasi. Selanjutnya, dokumentasikan dependensi ke aplikasi dan layanan lain, baik internal maupun eksternal.

Dengan tidak adanya perkakas untuk memvalidasi dependensi dan pemetaan, diperlukan pendekatan manual. Misalnya, Anda dapat masuk ke komponen infrastruktur dan menjalankan skrip untuk mengumpulkan informasi komunikasi seperti port terbuka dan koneksi yang sudah mapan. Demikian juga, Anda dapat mengidentifikasi proses yang berjalan dan perangkat lunak yang diinstal. Jangan meremehkan upaya yang diperlukan untuk penemuan manual. Alat AI agen, seperti Kiro,

dapat membantu Anda membangun skrip dan menganalisis output data. Perkakas terprogram dapat menangkap dan melaporkan sebagian besar dependensi dalam beberapa hari, kecuali yang terjadi pada interval yang lebih besar (biasanya persentase kecil). Penemuan manual dapat memakan waktu berminggu-minggu untuk mengumpulkan dan menggabungkan semua titik data, dan masih rentan terhadap kesalahan dan data yang hilang.

Lanjutkan untuk mendapatkan informasi yang ditentukan di bagian persyaratan data untuk setiap aplikasi yang diprioritaskan dan infrastruktur yang dipetakan. Selanjutnya, gunakan kuesioner berikut untuk memandu Anda melalui proses penilaian terperinci. Bertemu dengan pemangku kepentingan yang teridentifikasi untuk mendapatkan dan mendiskusikan jawaban atas pertanyaan-pertanyaan ini.

Umum

- Apa tingkat kekritisan aplikasi ini? Apakah itu menghasilkan pendapatan? Apakah ini aplikasi bisnis-strategis atau mendukung bisnis? Apakah ini layanan infrastruktur inti yang dimiliki oleh sistem lain?
- Apakah ada proyek transformasi yang sedang berlangsung untuk aplikasi ini?
- Apakah ini aplikasi yang dihadapi secara internal atau eksternal?

Arsitektur

- Apa jenis arsitektur saat ini (misalnya, SOA, layanan mikro, monolit)? Berapa banyak tingkatan yang dimiliki arsitektur? Apakah itu digabungkan dengan erat atau digabungkan secara longgar?
- Apa saja komponennya (misalnya, komputasi, database, penyimpanan jarak jauh, penyeimbang beban, layanan caching)?
- Apa tumpukan teknologi? Jelaskan sistem operasi, runtime, paket, kerangka kerja, dan middleware.
- Apa itu API? Jelaskan ini, termasuk nama API, operasi, URL, port, dan protokol.
- Apa aliran datanya? Jelaskan pekerjaan batch, ETL, pipeline, dan transfer file.
- Apa latensi maksimum yang ditoleransi antara komponen dan antara ini dan aplikasi atau layanan lainnya?

Operasi

- Di lokasi apa aplikasi ini beroperasi?

- Siapa yang mengoperasikan aplikasi dan infrastruktur? Apakah ini dioperasikan oleh tim internal atau AWS Mitra?
- Apa yang terjadi jika aplikasi ini turun? Siapa yang terpengaruh? Apa dampaknya?
- Di mana pengguna atau pelanggan berada? Bagaimana mereka mengakses aplikasi? Berapa jumlah pengguna bersamaan?
- Kapan penyegaran teknologi terakhir? Apakah penyegaran dijadwalkan di masa depan? Jika demikian, kapan?
- Apa risiko dan masalah yang diketahui untuk aplikasi ini? Bagaimana sejarah pemadaman dan insiden tingkat keparahan sedang dan tingkat keparahan tinggi?
- Apa siklus penggunaan (dalam jam kerja)? Apa zona waktu operasi?
- Apa periode pembekuan perubahan?
- Solusi apa yang digunakan untuk memantau aplikasi ini?

Performa

- Apa yang ditunjukkan oleh informasi kinerja yang dikumpulkan? Apakah penggunaan runcing atau konstan dan dapat diprediksi?
- Berapa kerangka waktu, interval, dan tanggal data kinerja yang tersedia?

Siklus hidup perangkat lunak

- Berapa tingkat perubahan saat ini (mingguan, bulanan, triwulanan, atau tahunan)?
- Apa siklus hidup pengembangan (misalnya, pengujian, pengembangan, QA, UAT, pra-produksi, produksi)?
- Apa metode penyebaran untuk aplikasi dan infrastruktur?
- Apa itu perkakas penyebaran?
- Apakah aplikasi atau infrastruktur ini menggunakan continuous integration (CI) /continuous delivery (CD)? Apa tingkat otomatisasi? Apa tugas manualnya?
- Apa persyaratan lisensi untuk aplikasi dan infrastruktur?
- Apa itu Service Level Agreement (SLA)?
- Apa mekanisme pengujian saat ini? Apa tahapan tesnya?

Migrasi

- Apa pertimbangan migrasi?
- Apa kompleksitas teknisnya?
- Apa risiko bisnisnya?
- Apa tingkat kepercayaan dalam data yang dikumpulkan?

Pada titik ini, perhatikan pertimbangan apa pun saat memigrasikan aplikasi ini. Untuk penilaian yang lebih lengkap dan akurat, dapatkan jawaban atas pertanyaan ini dari berbagai pemangku kepentingan. Kemudian kontraskan pengetahuan dan pendapat mereka.

Ketahanan

- Apa metode pencadangan saat ini? Produk apa yang digunakan untuk cadangan? Apa jadwal pencadangan? Apa kebijakan retensi cadangan?
- Apa tujuan titik pemulihan (RPO) dan tujuan waktu pemulihan (RTO) saat ini?
- Apakah aplikasi ini memiliki rencana pemulihan bencana (DR)? Jika demikian, apa solusi DR?
- Kapan tes DR terakhir?

Keamanan dan kepatuhan

- Apa saja kerangka kepatuhan dan peraturan yang berlaku untuk aplikasi ini? Apa tanggal audit terakhir dan berikutnya?
- Apakah aplikasi ini menyimpan data sensitif? Apa klasifikasi data?
- Apakah data dienkripsi saat transit atau diam, atau keduanya? Apa mekanisme enkripsi?
- Apakah aplikasi ini menggunakan sertifikat SSL? Apa otoritas penerbit?
- Apa metode otentikasi untuk pengguna, komponen, dan aplikasi dan layanan lainnya?

Basis Data

- Database apa yang digunakan aplikasi ini?
- Berapa jumlah tipikal koneksi bersamaan ke database? Berapa jumlah minimum dan jumlah maksimum koneksi?
- Apa metode koneksi (misalnya, JDBC, ODBC)?

- Apakah string koneksi didokumentasikan? Jika demikian, di mana?
- Apa skema database?
- Apakah database menggunakan tipe data khusus?

Dependensi

- Apa ketergantungan antar komponen? Perhatikan dependensi apa pun yang tidak dapat diselesaikan dan yang memerlukan migrasi komponen bersama-sama.
- Apakah komponen dibagi di seluruh lokasi? Apa konektivitas antara lokasi-lokasi ini (misalnya, WAN, VPN)?
- Apa ketergantungan aplikasi ini ke aplikasi atau layanan lain?
- Apa dependensi operasional? Misalnya, siklus pemeliharaan dan rilis seperti menambal jendela.

AWS desain aplikasi dan strategi migrasi

Merancang dan mendokumentasikan keadaan masa depan aplikasi Anda adalah faktor keberhasilan migrasi utama. Sebaiknya buat desain untuk semua jenis strategi migrasi, tidak peduli seberapa sederhana atau kompleksnya. Membuat desain akan memunculkan potensi pemblokir, dependensi, dan peluang untuk mengoptimalkan aplikasi bahkan dalam kasus di mana arsitektur tidak diharapkan berubah. Selain itu, desain yang dihasilkan akan berfungsi sebagai dasar untuk evolusi lebih lanjut setelah migrasi.

Selain pentingnya proses desain, itu juga dapat menjadi pemblokir dan berdampak pada kecepatan migrasi. Ini terjadi karena proses peninjauan arsitektur lama yang disusun untuk sistem lokal. Kami menyarankan Anda meninjau tinjauan arsitektur dan proses persetujuan Anda dan fokus pada penetapan pola atau jenis arsitektur yang dapat disetujui sebelumnya. Jangan meremehkan upaya meninjau arsitektur untuk ratusan atau ribuan aplikasi, dan mengambil kesempatan untuk memodernisasi proses.

Daftar berikut berisi sumber daya untuk membantu proses desain:

- AWS Pusat Arsitektur menggabungkan alat dan panduan, seperti AWS Well-Architected Kerangka Kerja. Selain itu, ia menyediakan arsitektur referensi yang dapat Anda gunakan untuk aplikasi Anda.
- Amazon Builders' Library berisi beberapa sumber tentang bagaimana Amazon membangun dan mengoperasikan perangkat lunak.

- AWS Solutions Library menawarkan koleksi solusi berbasis cloud, yang diperiksa oleh AWS, untuk puluhan masalah teknis dan bisnis. Ini mencakup banyak koleksi arsitektur referensi.
- AWS Panduan Preskriptif menyediakan strategi, panduan, dan pola yang membantu proses desain dan praktik terbaik migrasi.
- AWS Documentationberisi informasi tentang AWS layanan, termasuk Panduan Pengguna dan Referensi API.
- Getting Started Resource Center menyediakan beberapa tutorial langsung dan penyelaman mendalam untuk mempelajari dasar-dasarnya sehingga Anda dapat mulai membangun. AWS

Tergantung di mana Anda berada dalam perjalanan cloud, AWS yayasan mungkin sudah ada. AWS Yayasan ini meliputi:

- Wilayah AWS telah diidentifikasi.
- Akun telah dibuat atau dapat diperoleh sesuai permintaan.
- Jaringan umum telah diterapkan.
- AWS Layanan dasar telah digunakan di dalam akun.

Sebaliknya, Anda mungkin berada di awal proses, dan AWS fondasi belum didirikan. Kurangnya fondasi yang mapan dapat membatasi ruang lingkup desain aplikasi Anda atau memerlukan pekerjaan lebih lanjut untuk mendefinisikannya. Jika ini masalahnya, kami sarankan untuk mendefinisikan dan menerapkan desain dasar dari landing zone secara paralel dengan pekerjaan desain aplikasi. Desain aplikasi membantu mengidentifikasi persyaratan seperti Akun AWS struktur, jaringan, virtual private cloud (VPC), rentang Classless Inter-Domain Routing (CIDR), layanan bersama, keamanan, dan operasi cloud.

AWS Control Towermenyediakan cara termudah untuk mengatur dan mengatur AWS lingkungan multi-akun yang aman, yang disebut landing zone. AWS Control Tower membuat landing zone Anda menggunakan AWS Organizations, yang menyediakan pengelolaan dan tata kelola akun yang berkelanjutan serta implementasi pengalaman berbasis praktik AWS terbaik bekerja dengan ribuan pelanggan saat mereka pindah ke cloud.

Aplikasi future state

Mulailah dengan menetapkan strategi migrasi awal untuk aplikasi ini. Pada titik ini, strategi dianggap awal karena dapat berubah sebagai bagian dari desain negara masa depan, yang dapat

mengungkap potensi keterbatasan. Untuk memvalidasi asumsi awal, lihat pohon keputusan 6 Rs. Juga, dokumentasikan fase migrasi potensial. Misalnya, apakah aplikasi ini akan dimigrasikan dalam satu acara (semua komponen dimigrasikan secara bersamaan)? Atau apakah ini migrasi bertahap (beberapa komponen dimigrasikan nanti)?

Perhatikan bahwa strategi migrasi untuk aplikasi tertentu mungkin tidak unik. Ini karena beberapa tipe R dapat digunakan untuk memigrasikan komponen aplikasi. Misalnya, pendekatan awal bisa mengangkat dan menggeser aplikasi tanpa perubahan. Namun, komponen aplikasi mungkin berada di aset infrastruktur yang berbeda yang mungkin memerlukan perawatan yang berbeda. Misalnya, aplikasi terdiri dari tiga komponen, masing-masing berjalan di server terpisah, dan salah satu server menjalankan sistem operasi lama yang tidak didukung di cloud. Komponen itu akan memerlukan pendekatan replatform, sementara dua komponen lainnya, berjalan dalam versi server yang didukung, dapat dihosting ulang. Ini adalah kunci untuk menetapkan strategi migrasi ke setiap komponen aplikasi dan infrastruktur terkait yang sedang dimigrasikan.

Selanjutnya, dokumentasikan konteks dan masalah, dan tautkan artefak yang ada yang menentukan status saat ini:

- Mengapa aplikasi ini dimigrasikan?
- Apa saja perubahan yang diusulkan?
- Apa manfaatnya?
- Apakah ada risiko atau pemblokir utama?
- Apa kerugian saat ini?
- Apa yang ada di ruang lingkup dan di luar ruang lingkup?

Pengulangan

Sepanjang pekerjaan desain, pertimbangkan bagaimana solusi dan arsitektur untuk aplikasi ini dapat digunakan kembali untuk aplikasi lain. Bisakah solusi ini digeneralisasi?

Persyaratan

Dokumentasikan persyaratan fungsional dan nonfungsional untuk aplikasi ini, termasuk keamanan. Ini termasuk persyaratan status saat ini dan masa depan, tergantung pada strategi migrasi yang dipilih. Gunakan informasi yang dikumpulkan selama penilaian aplikasi terperinci untuk memandu proses ini.

To-be arsitektur

Jelaskan arsitektur future untuk aplikasi ini. Pertimbangkan untuk membuat templat diagram yang dapat digunakan kembali yang berisi blok bangunan untuk lingkungan sumber Anda (lokal) dan AWS lingkungan target (misalnya, target, akun, VPC Wilayah AWS, dan Availability Zones).

Buat tabel komponen yang sedang dimigrasikan dan komponen yang akan baru. Sertakan aplikasi dan layanan lain (baik di tempat atau di cloud) yang berinteraksi dengan aplikasi ini.

Tabel berikut mencantumkan contoh komponen. Ini tidak mewakili arsitektur referensi atau konfigurasi yang diperiksa.

Nama	Deskripsi	Detail
Aplikasi	Layanan eksternal (koneksi masuk)	Layanan menggunakan data dari API yang terbuka.
DNS	Resolusi nama (internal)	Amazon Route 53 digunakan sebagai bagian dari pengatura n akun dasar
Penyeimbang Beban Aplikasi	Mendistribusikan lalu lintas di antara layanan backend	Menggantikan penyeimbang beban lokal. Migrasi Kolam A.
Keamanan aplikasi	Perlindungan DDoS	Diimplementasikan dengan menggunakan AWS Shield
Grup keamanan	Firewall virtual	Batasi akses ke instance aplikasi pada port 443 (inbound).
Server A	Front-end	Rehost, menggunakan Amazon Elastic Compute Cloud (Amazon EC2).
Peladen B	Front-end	Rehost menggunakan Amazon EC2.
Server C	Logika aplikasi	Rehost menggunakan Amazon EC2.

Peladen D	Logika aplikasi	Rehost menggunakan Amazon EC2.
Layanan Database Relasional Amazon (Amazon RDS) - Amazon Aurora	Basis Data	Menggantikan server E dan F
Pemantauan dan peringatan	Ubah kontrol	Amazon CloudWatch
Pencatatan audit	Ubah kontrol	AWS CloudTrail
Penambalan dan akses jarak jauh	Maintenance	AWS Systems Manager
Akses sumber daya	Kontrol akses yang aman	AWS Identity and Access Management (IAM)
Autentikasi	Akses pengguna	Amazon Cognito
Sertifikat	SSL/TLS	AWS Certificate Manager
API 1	API Eksternal	Amazon API Gateway
Penyimpanan objek	Hosting gambar	Amazon Simple Storage Service (Amazon S3)
Kredensial	Manajemen dan hosting kredensional	AWS Secrets Manager
Fungsi AWS Lambda	Pengambilan kredensi database dan kunci API	AWS Lambda
gateway internet	Akses internet outbound	Gerbang internet ke VPC
Subnet pribadi 1	Backend dan DB	Zona Ketersediaan 1 — VPC 1
Subnet pribadi 2	Backend dan DB	Zona Ketersediaan 2 — VPC 1
Subnet publik 1	Front-end	Zona Ketersediaan 1 — VPC 1
Subnet publik 2	Front-end	Zona Ketersediaan 2 — VPC 1

Layanan Backup	Database dan cadangan instans EC2	AWS Backup
DR	Ketahanan Amazon EC2	AWS Elastic Disaster Recovery

Setelah komponen diidentifikasi, plot mereka dalam diagram menggunakan alat pilihan Anda. Bagikan desain awal dengan pemangku kepentingan aplikasi utama, termasuk pemilik aplikasi, arsitek perusahaan, dan tim platform dan migrasi. Pertimbangkan untuk mengajukan pertanyaan-pertanyaan berikut:

- Apakah tim umumnya setuju dengan desainnya?
- Bisakah tim operasi mendukungnya?
- Bisakah desain dikembangkan?
- Apakah ada pilihan lain?
- Apakah desain sesuai dengan standar arsitektur dan kebijakan keamanan?
- Apakah ada komponen yang hilang (misalnya, repositori kode, CI/CD perkakas, titik akhir VPC)?

Keputusan arsitektur

Sebagai bagian dari proses desain, Anda mungkin akan menemukan lebih banyak opsi untuk keseluruhan arsitektur atau bagian tertentu darinya. Dokumentasikan opsi ini di samping alasan untuk opsi yang disukai atau dipilih. Keputusan ini dapat didokumentasikan sebagai keputusan arsitektur.

Pastikan bahwa opsi utama terdaftar dan dijelaskan dengan cukup detail bagi pembaca baru untuk memahami opsi dan alasan di balik keputusan untuk menggunakan satu opsi di atas yang lain.

Lingkungan siklus hidup perangkat lunak

Dokumentasikan setiap perubahan pada lingkungan saat ini. Misalnya, lingkungan pengujian dan pengembangan akan dibuat ulang AWS dan tidak dimigrasikan.

Penandaan

Jelaskan penandaan wajib dan direkomendasikan untuk setiap komponen infrastruktur serta nilai penandaan untuk desain ini.

Strategi migrasi

Pada titik desain ini, asumsi awal tentang strategi migrasi harus divalidasi. Konfirmasikan bahwa ada konsensus tentang strategi R yang dipilih. Dokumentasikan strategi migrasi aplikasi secara keseluruhan dan strategi untuk komponen aplikasi individual. Seperti disebutkan sebelumnya, komponen aplikasi yang berbeda mungkin memerlukan tipe R yang berbeda untuk migrasi.

Selain itu, selaraskan strategi migrasi dengan pendorong dan hasil bisnis utama. Juga, jelaskan setiap pendekatan bertahap untuk migrasi, seperti pergerakan komponen dalam peristiwa migrasi yang berbeda.

Untuk informasi lebih lanjut tentang menentukan 6 Rs Anda, lihat rekomendasi AWS Migration Hub strategi.

Pola dan alat migrasi

Dengan strategi migrasi yang ditentukan untuk komponen aplikasi dan infrastruktur, Anda sekarang dapat menjelajahi pola teknis tertentu. Misalnya, strategi rehost dapat diimplementasikan dengan alat migrasi seperti. AWS Transform MGN Jika Anda tidak perlu mereplikasi status atau data, Anda dapat mencapai hasil yang sama dengan menerapkan ulang aplikasi menggunakan Amazon Machine Image (AMI) dan pipeline penerapan aplikasi.

Demikian pula, untuk memplatform ulang atau memfaktorkan ulang (arsitek ulang) aplikasi, Anda dapat menggunakan alat seperti AWS App2Container, AWS Database Migration Service (AWS DMS), (),AWS Schema Conversion Tool.AWS SCTAWS DataSync Untuk containerizing, Anda dapat menggunakan Amazon Elastic Container Service (Amazon ECS), Amazon Elastic Kubernetes Service (Amazon EKS), atau. AWS Fargate Saat membeli kembali, Anda dapat menggunakan AMI untuk produk tertentu atau solusi perangkat lunak sebagai layanan (SaaS) dari. AWS Marketplace

Evaluasi berbagai pola dan opsi yang tersedia untuk mencapai tujuan. Pertimbangkan pro dan kontra, dan kesiapan operasional migrasi. Untuk membantu analisis Anda, gunakan pertanyaan-pertanyaan berikut:

- Dapatkah tim migrasi mendukung pola-pola ini?
- Apa keseimbangan antara biaya dan manfaat?
- Bisakah aplikasi, layanan, atau komponen ini dipindahkan ke layanan terkelola?
- Apa upaya untuk menerapkan pola ini?
- Apakah ada peraturan atau kebijakan kepatuhan yang mencegah penggunaan pola tertentu?

- Bisakah pola ini digunakan kembali? Pola yang dapat digunakan kembali lebih disukai. Namun, terkadang sebuah pola hanya akan digunakan satu kali. Pertimbangkan keseimbangan antara upaya pola sekali pakai atas pola alternatif yang dapat digunakan kembali.

AWS Panduan Preskriptif berisi berbagai pola dan teknik migrasi.

Manajemen dan operasi layanan

Saat membuat desain aplikasi untuk migrasi ke AWS, pertimbangkan kesiapan operasional. Saat mengevaluasi persyaratan kesiapan dengan tim aplikasi dan infrastruktur Anda, pertimbangkan pertanyaan-pertanyaan berikut:

- Apakah mereka siap mengoperasikannya?
- Apakah prosedur respons insiden didefinisikan?
- Apa perjanjian tingkat layanan yang diharapkan (SLA)?
- Apakah pemisahan tugas diperlukan?
- Apakah tim yang berbeda siap untuk mengoordinasikan tindakan dukungan?
- Siapa yang bertanggung jawab untuk apa?

Pertimbangan cutover

Mempertimbangkan strategi dan pola migrasi, apa yang penting untuk diketahui saat aplikasi dimigrasikan? Perencanaan cutover adalah kegiatan pasca-desain. Namun, dokumentasikan pertimbangan apa pun untuk kegiatan dan persyaratan yang dapat diantisipasi. Misalnya, mendokumentasikan persyaratan untuk melakukan bukti konsep, jika berlaku, dan menguraikan persyaratan pengujian, audit, atau validasi.

Risiko, asumsi, masalah, dan ketergantungan

Dokumentasikan setiap risiko terbuka, asumsi, dan potensi masalah yang belum terselesaikan. Tetapkan kepemilikan yang jelas untuk item-item ini, dan lacak kemajuan sehingga desain dan strategi keseluruhan dapat disetujui untuk implementasi. Selain itu, dependensi kunci dokumen untuk mengimplementasikan desain ini.

Memperkirakan biaya operasional

Untuk memperkirakan biaya arsitektur AWS target Anda, gunakan file AWS Kalkulator Harga. Tambahkan komponen infrastruktur Anda seperti yang ditentukan oleh desain Anda, dan dapatkan perkiraan biaya operasional. Faktor dalam lisensi perangkat lunak yang diperlukan untuk komponen aplikasi Anda dan yang belum termasuk dalam Layanan AWS yang akan Anda gunakan.

Analisis portofolio dan perencanaan migrasi

Tahap penilaian ini berfokus pada penyelesaian penemuan dan analisis tingkat portofolio yang dimulai di bagian penemuan Portofolio dan perencanaan awal. Tujuannya adalah untuk mengulangi dan menetapkan dasar yang solid untuk portofolio aplikasi dan infrastruktur. Garis dasar ini mencakup identifikasi semua dependensi, mengulangi model rasionalisasi untuk migrasi, membuat kasus bisnis terperinci, dan menguraikan rencana gelombang migrasi. Akibatnya, kesetiaan data yang dibutuhkan lebih tinggi. Tahap ini akan membutuhkan investasi waktu. Untuk mempercepat hasil penilaian, kami sarankan untuk menggunakan sebanyak mungkin sumber data terprogram, seperti alat penemuan.

Hasil utama dari tahap ini meliputi:

- Aplikasi dengan kesetiaan tinggi dan inventaris infrastruktur
- Strategi migrasi yang ditentukan untuk setiap aplikasi
- Rencana gelombang migrasi dengan kepercayaan tinggi
- Kasus bisnis yang terperinci

Memahami persyaratan data penilaian lengkap

Tabel berikut menjelaskan informasi yang diperlukan untuk mendapatkan tampilan portofolio lengkap dari aplikasi dalam migrasi dan infrastruktur terkait.

Tabel menggunakan singkatan berikut:

- R, untuk yang dibutuhkan
- O, untuk opsional
- N/A, karena tidak berlaku

Aplikasi

Nama atribut	Deskripsi	Inventarisasi dan prioritas	Kasus bisnis terperinci	Tingkat kesetiaan yang disarankan (minimum)

Pengidentifikasi unik	Misalnya, ID aplikasi. Biasanya tersedia pada CMDB yang ada atau inventaris internal dan sistem kontrol lainnya. Pertimbangkan untuk membuat ID unik setiap kali ini tidak ditentukan dalam organisasi Anda.	R	R	Tinggi
Nama aplikasi	Nama yang dengannya aplikasi ini diketahui organisasi Anda. Sertakan vendor komersial off-the-shelf (COTS) dan nama produk bila berlaku.	R	R	Tinggi
Apakah COTS?	Ya atau Tidak. Apakah ini aplikasi komersial atau pengembangan internal	R	R	Tinggi

Produk dan versi COTS	Nama dan versi produk perangkat lunak komersial	R	R	Tinggi
Deskripsi	Fungsi dan konteks aplikasi utama	R	R	Tinggi
Kekritisan	Misalnya, aplikasi strategis atau menghasilkan pendapatan, atau mendukung fungsi kritis	R	R	Tinggi
Tipe	Misalnya, database, manajemen hubungan pelanggan (CRM), aplikasi web, multimedia, layanan bersama TI	R	R	Tinggi
Lingkungan	Misalnya, produksi, pra-produksi, pengembangan, pengujian, kotak pasir	R	R	Tinggi

Kepatuhan dan peraturan	Kerangka kerja yang berlaku untuk beban kerja (misalnya , HIPAA, SOX,, ISO, SOC PCI-DSS, FedRAMP) dan persyaratan peraturan	R	R	Tinggi
Dependensi	Dependensi hulu dan hilir ke aplikasi atau layanan internal dan eksternal. Non-technical dependensi seperti elemen operasional (misalnya, siklus pemeliharaan).	R	O	Tinggi
Pemetaan infrastruktur	Pemetaan ke aset and/or virtual fisik yang membentuk aplikasi	R	R	Tinggi
Lisensi	Jenis lisensi perangkat lunak komoditas (misalnya, Microsoft SQL Server Enterpris e)	R	R	Medium-high

Biaya	Biaya untuk lisensi perangkat lunak, operasi perangkat lunak, dan pemelihar aan	O	R	Medium-high
Unit bisnis	Misalnya, pemasaran , keuangan, penjualan	R	R	Tinggi
Detail pemilik	Informasi kontak untuk pemilik aplikasi	R	R	Tinggi
Informasi DR	Komponen pemulihan bencana	R	R	Tinggi
Strategi migrasi	Misalnya, salah satu dari 6 Rs untuk migrasi ke AWS	R	R	Tinggi
Tiket Support	Data 12-24 bulan untuk membantu menilai produktiv itas dan dampak keuangan dari pemadaman, perlambatan, pembatasan transaksi, dan pembengkakan jendela batch	O	R	Sedang

Infrastruktur

Nama atribut	Deskripsi	Inventarisasi dan prioritas	Kasus bisnis	Tingkat kesetiaan yang disarankan (minimum)
Pengidentifikasi unik	Misalnya, ID server. Biasanya tersedia pada CMDB yang ada atau inventaris internal dan sistem kontrol lainnya. Pertimbangkan untuk membuat ID unik setiap kali ini tidak ditentukan dalam organisasi Anda.	R	R	Tinggi
Nama jaringan	Nama aset dalam jaringan (misalnya, nama host)	R	R	Tinggi
Nama DNS (nama domain yang sepenuhnya memenuhi syarat, atau FQDN)	Nama DNS	R	O	Tinggi
Alamat IP dan netmask	Alamat IP and/or publik internal	R	R	Tinggi

Jenis aset	Misalnya, server fisik atau virtual, hyperviso r, wadah, perangkat, contoh database	R	R	Tinggi
Nama produk	Vendor komersial dan nama produk (misalnya, VMware ESXi, IBM Power Systems, Exadata)	R	R	Tinggi
Sistem operasi	Misalnya, REHL 8, Windows Server 2019, AIX 6.1	R	R	Tinggi
Konfigurasi	CPU yang dialokasikan, jumlah inti, utas per inti, total memori, penyimpanan, kartu jaringan	R	R	Tinggi
Pemanfaatan	CPU, memori, dan puncak penyimpanan dan rata-rata . Throughpu t instance database.	R	R	Tinggi

Lisensi	Jenis lisensi komoditas (misalnya, Standar RHEL)	R	R	Tinggi
Pemetaan aplikasi	Aplikasi atau komponen aplikasi yang berjalan di infrastruktur ini	R	R	Tinggi
Biaya	Biaya terisi penuh untuk server bare-metal, termasuk perangkat keras, pemeliharaan, operasi, penyimpanan (SAN, NAS, objek), lisensi sistem operasi, pembagian ruang rak, dan overhead pusat data	O	R	Medium-high
Perkiraan volume transfer data (in/out)	Misalnya, per aset infrastruktur per hari selama periode 30 hari	O	R	Sedang

Jaringan

Nama atribut	Deskripsi	Inventarisasi dan prioritas	Kasus bisnis	Tingkat kesetiaan yang

				disarankan (minimum)
Ukuran pipa (Mb/s), redundansi () Y/N	Spesifikasi tautan WAN saat ini (misalnya , 1000 Mb/s redundan)	R	R	Medium-high
Lokasi yang terhubung	Lokasi bernama yang dihubungkan oleh tautan ini	R	O	Tinggi
Pemanfaatan tautan	Puncak dan pemanfaatan rata-rata, transfer data keluar () GB/month	R	R	Medium-high
Latensi (ms)	Latensi saat ini antara lokasi yang terhubung.	R	O	Tinggi
Biaya	Biaya saat ini per bulan	N/A	R	Medium-high

Migrasi

Nama atribut	Deskripsi	Inventarisasi dan prioritas	Kasus bisnis	Tingkat kesetiaan yang disarankan (minimum)
Biaya rehost	Upaya pelanggan dan mitra untuk setiap beban	N/A	R	Medium-high

	kerja (orang-hari), tarif biaya pelanggan dan Mitra per hari, biaya alat, jumlah beban kerja			
Biaya replatform	Upaya pelanggan dan mitra untuk setiap beban kerja (orang-hari), tarif biaya pelanggan dan mitra per hari, jumlah beban kerja	N/A	R	Medium-high
Biaya refactor	Upaya pelanggan dan mitra untuk setiap beban kerja (orang-hari), tarif biaya pelanggan dan mitra per hari, jumlah beban kerja	N/A	R	Medium-high
Biaya pensiun	Jumlah server, biaya dekomisi rata-rata	N/A	R	Medium-high

Zona pendaratan	Re-use existing (Y/N), daftar Wilayah AWS kebutuhan, biaya	N/A	R	Medium-high
Orang dan perubahan	Jumlah staf yang akan dilatih dalam operasi dan pengembangan cloud, biaya pelatihan per orang, biaya waktu pelatihan per orang	N/A	R	Medium-high
Durasi	Durasi migrasi beban kerja dalam lingkup (bulan)	O	R	Medium-high
Biaya paralel	Kerangka waktu dan tingkat biaya apa adanya dapat dihapus selama migrasi	N/A	R	Medium-high
	Kerangka waktu dan tingkat di mana produk dan layanan AWS, serta biaya infrastruktur lainnya, diperkenalkan selama migrasi	N/A	R	Medium-high

Menetapkan baseline untuk portofolio aplikasi

Untuk membuat rencana gelombang migrasi dengan kepercayaan tinggi, Anda harus menetapkan dasar untuk portofolio aplikasi dan infrastruktur terkaitnya. Garis dasar portofolio memberikan pandangan komprehensif tentang ruang lingkup migrasi, termasuk dependensi teknis dan strategi migrasi. Garis dasar portofolio memberikan kejelasan tentang aplikasi mana yang berada dalam lingkup migrasi dan bahwa titik data yang diuraikan dalam bagian Memahami persyaratan data penilaian lengkap dikumpulkan. Demikian juga, semua infrastruktur terkait (komputasi, jaringan penyimpanan) dipahami dan dipetakan ke aplikasi.

Ketergantungan teknis dapat dijelaskan dalam empat kategori:

- Application-to-infrastructuredependensi membangun hubungan antara perangkat lunak dan perangkat keras fisik atau virtual. Misalnya, ada ketergantungan antara aplikasi CRM dan mesin virtual tempat ia diinstal.
- Application-componentdependensi menggambarkan bagaimana komponen yang berjalan di aset infrastruktur yang berbeda berinteraksi. Contoh ketergantungan komponen aplikasi adalah front end web yang berjalan pada mesin virtual, dengan lapisan aplikasi yang berjalan pada mesin virtual yang berbeda, dan database yang berjalan pada cluster database.
- Application-to-applicationdependensi berhubungan dengan interaksi antara aplikasi yang berbeda atau komponennya. Contoh ketergantungan aplikasi-ke-aplikasi adalah aplikasi pemrosesan pembayaran dan aplikasi manajemen saham. Aplikasi ini independen, tetapi mereka terus-menerus berinteraksi menggunakan operasi API yang ditentukan atau kumpulan data bersama.
- Application-to-infrastructure dependensi layanan secara teknis adalah dependensi aplikasi-ke-aplikasi, mengingat bahwa layanan infrastruktur itu sendiri merupakan aplikasi. Namun, kami sarankan untuk mengkategorikan ini secara terpisah. Alasan utamanya adalah bahwa layanan infrastruktur biasanya dibagi oleh banyak aplikasi, sehingga mereka memiliki jejak ketergantungan yang panjang. Mereka juga biasanya mengikuti strategi dan pola migrasi yang berbeda.

  Misalnya, penyeimbang beban dapat berisi kolam penyeimbang untuk beberapa aplikasi. Yang penting adalah ketergantungan pada kumpulan, yang kemungkinan akan dimigrasikan secara individual, di samping aplikasi dependen, sementara penyeimbang beban itu sendiri dipertahankan atau dihentikan.

  Selain itu, individualisasi dependensi layanan aplikasi-ke-infrastruktur membantu menghindari grup dependensi palsu. Kelompok ketergantungan palsu adalah ketika beberapa aplikasi bisnis dikelompokkan bersama, menyiratkan bahwa memiliki ketergantungan umum pada layanan

infrastruktur memaksa mereka untuk dimigrasi pada saat yang sama. Misalnya, layanan otentikasi, seperti Active Directory, cenderung dikaitkan dengan kelompok besar aplikasi. Kuncinya adalah memfilter dependensi layanan infrastruktur, dan mengaktifkan layanan ini masuk. AWS

Untuk informasi selengkapnya tentang cara bekerja dengan dependensi untuk membuat grup migrasi, lihat bagian Perencanaan Gelombang.

Saat Anda menetapkan baseline untuk portofolio, sebaiknya Anda mengonfirmasi strategi migrasi untuk setiap komponen aplikasi. Strategi migrasi akan menjadi salah satu dari 6 Rs untuk migrasi (lihat bagian Mengulangi strategi migrasi 6 Rs). Dalam baseline portofolio, salah satu dari 6 Rs harus dikaitkan dengan setiap aplikasi. Strategi 6 R juga harus dikaitkan dengan masing-masing komponen infrastruktur aplikasi.

Untuk menetapkan versi dasar portofolio, termasuk dependensi dan strategi migrasi, gunakan alat penemuan otomatis (lihat Mengevaluasi kebutuhan alat penemuan). Lengkapi data dengan informasi yang dikumpulkan dari pemangku kepentingan utama seperti pemilik aplikasi dan tim infrastruktur. Terus kumpulkan data sampai Anda mendapatkan inventaris portofolio lengkap yang sesuai dengan atribut dan tingkat kesetiaan yang diuraikan di bagian persyaratan data untuk tahap ini. Dataset yang dihasilkan akan berperan penting dalam mendorong migrasi.

Pertimbangkan bahwa, tergantung pada luas cakupan migrasi Anda dan alat yang tersedia, aktivitas ini dapat memakan waktu beberapa minggu untuk diselesaikan.

Mengulangi kriteria prioritas

Sebelum Anda membuat rencana gelombang migrasi, kami sarankan Anda mengulangi kriteria prioritas aplikasi untuk beralih dari pemilihan aplikasi pilot ke perencanaan gelombang jangka panjang.

Di bagian sebelumnya, kami memperkenalkan kriteria prioritas default yang akan memprioritaskan aplikasi sederhana yang siap cloud (lihat Memprioritaskan aplikasi). Ini karena pada tahap awal kami merekomendasikan memulai dengan aplikasi nonkritis untuk menyempurnakan proses migrasi dan menggabungkan pelajaran yang dipetik. Namun, pada tahap ini, dan untuk membuat rencana jangka panjang, urutan migrasi aplikasi harus diselaraskan dengan driver bisnis. Menerapkan kriteria baru akan menghasilkan peringkat aplikasi baru yang akan menjadi masukan utama untuk perencanaan gelombang.

Tinjau poin data yang tersedia dari portofolio aplikasi, dan pilih atribut yang akan menentukan prioritas aplikasi berdasarkan driver bisnis.

Pertama, validasi driver bisnis Anda (lihat Penggerak bisnis dan prinsip panduan teknis). Selanjutnya, berdasarkan driver bisnis Anda, pilih atribut yang akan membantu memprioritaskan aplikasi untuk migrasi.

Tabel berikut menunjukkan contoh kriteria prioritas yang selaras dengan pendorong bisnis untuk inovasi.

Atribut atau titik data	Kemungkinan nilai	Skor (0-99)	Pentingnya atau relevansi faktor pengalikan
Sistem operasi	AIX	80	Tinggi (1x)
	Solaris	80	
	HP-UX	80	
	Mainframe	70	
	Windows	50	
	Linux	20	
Kekritisan bisnis	Tinggi	60	Tinggi (1x)
	Sedang	40	
	Rendah	20	
Arsitektur	Dipasangkan dengan erat	60	Tinggi (1x)
	Berpasangan secara longgar	20	
Model operasi	Tradisional - tidak CI/CD	60	Medium-high (0,8x)
	Dasar CI/CD	40	
	Penuh DevOps	20	

Jumlah instans komputasi	1-3	60	Medium-high (0,8x)
	4-10	40	
	11 atau lebih	20	
Strategi migrasi	Refactor (arsitek ulang)	70	Sedang (0.6x)
	Platform Ulang	40	
	Pembelian kembali	30	
	Rehost	10	

Tabel berikut menunjukkan contoh kriteria prioritas yang selaras dengan pendorong bisnis untuk pengurangan biaya cepat.

Atribut atau titik data	Kemungkinan nilai	Skor (0-99)	Pentingnya atau relevansi faktor pengalikan
Produk basis data	Oracle	70	Tinggi (1x)
	Microsoft SQL	70	
	Lainnya	20	
Sistem operasi	Windows	70	Tinggi (1x)
	Linux	70	
	Lainnya	20	
Pemanfaatan CPU (rata-rata)	Lebih dari 36%	60	Tinggi (1x)
	Kurang dari 36%	40	
Jumlah instans komputasi	11 atau lebih	60	Medium-high (0,8x)

	4-10	40	
	1-3	20	
Strategi migrasi	Pensiun	80	Sedang (0.6x)
	Rehost	70	
	Platform Ulang	50	
	Refactor (arsitek ulang)	10	

Uji kriteria prioritas dan ulangi sampai Anda umumnya setuju dengan output. Dibutuhkan setidaknya tiga atau empat iterasi untuk mendapatkan versi dasar.

Mengulangi pemilihan strategi migrasi 6 Rs

Pada tahap ini, kami menyarankan Anda mengulangi dan mengembangkan pohon keputusan 6 Rs. Bagian Menentukan tipe R untuk migrasi memperkenalkan pohon keputusan default. Kami merekomendasikan untuk merevisi pohon, mempertimbangkan pembelajaran selama migrasi aplikasi percontohan awal, dan memastikan bahwa itu masih selaras dengan driver bisnis, kriteria prioritas, dan keadaan unik Anda. Validasi pohon keputusan dengan aplikasi sampel, dan verifikasi bahwa itu masih menghasilkan strategi yang diharapkan. Jika tidak, perbarui logika yang sesuai. Pohon yang dihasilkan akan menjadi kunci dalam menetapkan garis dasar untuk portofolio aplikasi dan dalam mengalokasikan strategi migrasi untuk setiap komponen aplikasi.

Pastikan bahwa strategi migrasi dialokasikan untuk setiap komponen aplikasi dan infrastruktur yang terkait. Informasi ini merupakan faktor kunci ketika memperkirakan upaya, kapasitas, dan keterampilan yang dibutuhkan, dan saat membuat rencana gelombang migrasi.

Perencanaan gelombang

Intinya, rencana gelombang adalah jadwal migrasi, dan mirip dengan kegiatan perencanaan proyek lainnya. Kami menyarankan Anda menggunakan gelombang migrasi sebagai sarana untuk membuat grup yang dapat dikelola, mengurangi risiko, dan mengatur aktivitas di sekitar grup tersebut.

Untuk membuat rencana gelombang migrasi yang efektif dan percaya diri tinggi, Anda harus mendapatkan tampilan lengkap dari portofolio aplikasi, infrastruktur terkait (komputasi, penyimpanan, jaringan), pemetaan ketergantungan, dan strategi migrasi.

Selain aplikasi bisnis, yang merupakan bentuk pengelompokan kumpulan komponen perangkat lunak dan infrastruktur, Anda dapat menggunakan level grup lainnya. Gelombang adalah level grup tertinggi. Dalam gelombang, Anda dapat membuat grup ketergantungan. Jenis sub-grup ini dapat berisi lebih dari satu aplikasi. Misalnya, dua atau lebih aplikasi yang perlu dimigrasikan pada saat yang sama karena ketergantungan teknis, seperti latensi rendah, atau faktor lainnya. Kemudian kelompok ketergantungan itu dikelola secara keseluruhan. Beberapa kelompok ketergantungan dapat ditugaskan ke gelombang. Selanjutnya, Anda dapat menetapkan tanggal migrasi ke seluruh gelombang atau ke grup ketergantungan individu dalam gelombang. Tanggal migrasi adalah tanggal dan waktu ketika grup tersebut akan dihentikan di lokasi saat ini dan dibuat aktif AWS.

Gelombang migrasi memiliki banyak aktivitas. Kami menyarankan Anda mengatur gelombang secara bertahap dan menetapkan durasi yang diharapkan untuk setiap fase. Fase di bawah ini berfungsi sebagai contoh:

- Desain: Dalam fase gelombang ini, desain target untuk setiap aplikasi dalam gelombang dikonfirmasi dan disetujui.
- Perencanaan cutover: Fase gelombang ini mencakup pembuatan atau iterasi runbook cutover dan perencanaan semua langkah yang diperlukan untuk mengalihkan aplikasi ke AWS (termasuk skenario rollback).
- Pre-migration: Fase ini mencakup aktivitas penyebaran landing zone, seperti penyediaan akun, konfigurasi, pengujian premi, penyiapan perkakas migrasi, dan replikasi data.
- Cutover: Fase ini adalah saat migrasi sebenarnya terjadi. Selama waktu ini, aplikasi dihentikan di lokasi saat ini, data disinkronkan untuk terakhir kalinya, pengujian bisnis dilakukan, dan migrasi diselesaikan. Fase ini termasuk serah terima operasional.
- Hypercare atau Post-migration: Fase ini adalah periode waktu di mana tim migrasi tersedia untuk mendukung operasi jika terjadi masalah. Selain itu, optimasi dapat diterapkan sesuai kebutuhan.
- Penutupan gelombang: Pada fase ini, Anda meninjau metrik dan pelajaran yang dipetik dan menutup gelombang secara formal.

Tidak ada durasi yang telah ditentukan untuk gelombang migrasi, dan itu akan tergantung pada tingkat upaya dan kompleksitas. Kami merekomendasikan untuk menjaga gelombang migrasi dalam

waktu 6 hingga 10 minggu. Kasus di mana lebih banyak waktu diperlukan, misalnya, ketika benar-benar menulis ulang komponen aplikasi, biasanya lebih baik ditangani di luar gelombang migrasi.

Untuk mengukur keberhasilan dan melacak kemajuan, gelombang harus diselaraskan dengan hasil dan pendorong bisnis. Ini juga akan mempengaruhi durasi gelombang dan kelompok ketergantungan yang terkandung dalam gelombang. Penyelesaian gelombang harus mencerminkan pencapaian yang terukur.

Ada beberapa cara untuk mengatur gelombang migrasi. Tabel berikut menjelaskan opsi organisasi gelombang yang paling umum. Ini biasanya digabungkan.

Jenis organisasi gelombang	Deskripsi	Pro	Kontra
Dengan strategi migrasi atau tumpukan teknologi	Tetapkan aplikasi dengan strategi atau pola migrasi umum ke gelombang. Misalnya, gelombang yang hanya berisi aplikasi rehost.	Tim khusus per pola atau tumpukan dapat diberikan seluruh gelombang. Durasi aktivitas yang homogen.	Membutuhkan lebih banyak analisis dependensi, terutama untuk aplikasi yang mengikuti pola yang berbeda.
Berdasarkan domain bisnis	Buat gelombang per domain bisnis. Misalnya, gelombang manajemen pesanan atau gelombang pembayaran.	Data bersama biasanya dalam domain tertentu. Keterlibatan tim yang konsisten.	Peningkatan risiko karena seluruh domain bisnis terpengaruh.
Dengan kemampuan teknis	Kelompokkan aplikasi yang menggunak an satu atau lebih kemampuan. Misalnya, gelombang komputasi saja atau gelombang penyeimbang beban komputasi+.	Migrasi dimulai lebih cepat karena kemampuan teknis diaktifkan dari waktu ke waktu. Menghapus ketergantungan untuk landing zone yang beroperasi penuh.	Menciptakan kantong kompleksitas di gelombang selanjutn ya.

Berdasarkan lingkungan	Gelombang berisi lingkungan tertentu untuk satu set aplikasi. Misalnya, gelombang pengembangan atau gelombang produksi.	Non-production Gelombang mendapat manfaat dari fleksibilitas selama eksekusi. Mengurangi risiko migrasi produksi.	Membutuhkan fokus pada analisis ketergantungan untuk menghindari dependensi yang hilang yang tidak ada di lingkungan non-produksi.
Berdasarkan prioritas bisnis	Membuat grup murni berdasarkan kriteria prioritas yang diberikan.	Mengatasi hasil bisnis.	Biasanya banyak tim yang terlibat; sulit untuk dikoordinasikan.

Bagian tentang menetapkan garis dasar untuk portofolio aplikasi menjelaskan empat kategori dependensi teknis. Dependensi ini berkontribusi pada penciptaan gelombang migrasi dan definisi kelompok ketergantungan. Kelompok ketergantungan akan ditentukan oleh kekritisan ketergantungan. Selain itu, dependensi nonteknis harus dipertimbangkan. Misalnya, jadwal rilis aplikasi, jendela pemeliharaan, dan tanggal bisnis utama (seperti pemrosesan akhir bulan atau akhir kuartal) dapat memengaruhi rencana gelombang.

Tentukan apakah ketergantungannya lunak atau keras. Ketergantungan lunak adalah hubungan antara dua atau lebih aset, atau dari aset ke kendala, yang tidak tergantung pada lokasi komponen. Misalnya, dua sistem yang beroperasi di jaringan lokal yang sama (atau infrastruktur yang sama) dapat dipisahkan dengan memindahkan salah satu sistem tersebut ke cloud sementara yang lain tetap di tempat. Ketergantungan keras adalah hubungan antara dua atau lebih aset, atau dari aset ke kendala, yang bergantung pada lokasi. Misalnya, dua sistem yang beroperasi di jaringan lokal yang sama, dan yang sangat bergantung pada latensi rendah untuk komunikasi antara server aplikasi dan server database, memiliki ketergantungan keras. Memindahkan hanya satu dari sistem ini ke cloud akan menyebabkan masalah fungsionalitas atau kinerja yang tidak dapat diselesaikan. Demikian juga, alasan nonteknis, seperti ketersediaan sumber daya (seperti tim yang melakukan migrasi) atau kendala operasional (seperti jendela pemeliharaan di mana dua sistem hanya dapat dimigrasikan dalam jendela waktu tertentu), dapat membuat ketergantungan keras untuk aset ini.

Untuk membuat rencana gelombang migrasi, tentukan grup dependensi Anda dengan menganalisis dependensi, idealnya dari sumber data yang sangat tepercaya seperti alat penemuan khusus. Gabungkan informasi ini dengan kriteria prioritas aplikasi Anda dan keadaan operasional.

Menentukan dependensi teknis itu menantang. Beberapa titik data diperlukan, dan tidak ada sumber data yang berisi semuanya. Misalnya, meskipun Anda dapat memperoleh informasi komunikasi proses-ke-proses dari alat penemuan, sulit untuk mengklasifikasikannya menjadi dependensi lunak dan keras. Toleransi latensi juga sulit ditentukan dari data jaringan saja.

Teknik-teknik berikut dapat membantu Anda menangani ambiguitas dalam menentukan dependensi nyata:

- Kumpulkan semua data seperti yang dijelaskan di bagian persyaratan data dan titik data lainnya yang Anda pertimbangkan sesuai kebutuhan.
- Filter informasi ketergantungan (atau data komunikasi) dan kecualikan layanan bersama, seperti Active Directory, cadangan, dan pemantauan lalu lintas. Layanan bersama teknis cenderung merekatkan seluruh ruang lingkup.
- Klasifikasi semua informasi. Jika tersedia, gunakan frekuensi jaringan dan volume transfer data antar komponen.
- Bertemu dengan pemilik aplikasi, arsitek, dan tim pendukung. Diskusikan jenis koneksi. Apakah mereka sinkron atau asinkron? Apakah mereka menyadari persyaratan latensi minimum? Apa koneksi kritis, dan apa yang terjadi jika tidak tersedia? Apakah Anda kehilangan koneksi penting? Pertimbangkan bahwa proses batch mungkin terjadi secara sporadis dan hilang dalam kumpulan data.
- Jika alat penemuan Anda menyediakan grafik data, cari aplikasi tunggal yang menjembatani kelompok besar aplikasi. Titik koneksi tunggal ini dapat membantu memecah data menjadi kelompok-kelompok yang lebih kecil.

AWS Transform dapat membantu Anda menganalisis dependensi dan melakukan perencanaan gelombang.

Membuat rencana gelombang

Prasyarat untuk memigrasikan gelombang aplikasi adalah data portofolio aplikasi dan penilaian aplikasi terperinci dari kelompok aplikasi yang akan dimigrasikan dalam gelombang. Penilaian terperinci harus mencakup daftar aplikasi dalam gelombang, detail infrastruktur terkait, desain target, dan strategi migrasi untuk setiap aplikasi.

Menetapkan kepemilikan dan tata kelola gelombang adalah kunci untuk mengelola dan melacak pekerjaan gelombang, ketergantungan program, manajemen perubahan, masalah, dan risiko. Pastikan bahwa kerangka kerja tata kelola ada untuk mengelola rencana.

Untuk menguraikan rencana gelombang, mulailah dengan konstruksi gelombang default. Apa yang terjadi dalam gelombang? Setelah input awal ditentukan, gelombang dapat dimulai. Biasanya, kegiatannya adalah:

1. Perbaiki rencana cutover. Kegiatan ini harus menguraikan runbook dan langkah-langkah yang harus diambil pada saat migrasi, termasuk koordinasi dengan tim internal dan eksternal lainnya.
2. Perbaiki rencana rollback. Apa yang harus dilakukan untuk memutar kembali aplikasi jika ada yang salah?
3. Siapkan infrastruktur target. Misalnya, Anda dapat membuat atau memperluas AWS landing zone (Akun AWS, keamanan, jaringan, layanan infrastruktur, infrastruktur pendukung lainnya).
4. Uji infrastruktur target.
5. Mengoperasikan perkakas migrasi. Misalnya, instal agen replikasi dan mulai transfer data.
6. Lakukan rencana cutover dan runbook dry run. Kelompokkan semua anggota tim yang berpartisipasi, dan tinjau semua langkah sebelumnya.
7. Memantau replikasi data dan penyebaran infrastruktur.
8. Konfirmasikan kesiapan untuk pengoperasian infrastruktur dan aplikasi di AWS.
9. Konfirmasikan kesiapan keamanan.
10. Konfirmasikan kepatuhan dan persyaratan peraturan (misalnya, validasi beban kerja sebelum migrasi dan pasca-migrasi) jika berlaku.
11. Migrasikan aplikasi ke AWS dan lakukan pengujian pra go-live.
12. Berikan dukungan pasca-migrasi untuk jangka waktu tertentu, seperti 3 hari, saat tim operasi dan tim migrasi sepenuhnya tersedia untuk menyelesaikan masalah, dan menerapkan pengoptimalan.
13. Lakukan tinjauan pasca-migrasi. Dokumentasikan pelajaran yang dipetik, dan gabungkan ke dalam gelombang masa depan.
14. Lakukan penutupan gelombang dengan mengonfirmasi serah terima operasional dan perolehan metrik untuk pelaporan.

Berapa lama masing-masing kegiatan ini akan ditentukan oleh kompleksitas ruang lingkup, kapasitas gelombang, orang-orang yang terlibat, dan keadaan unik Anda. Jika memungkinkan, gelombang yang lebih kecil lebih disukai karena itu akan mengurangi dampak penundaan atau penghambat migrasi. Tentukan, dengan tim Anda, berapa durasi default gelombang.

Selanjutnya, lanjutkan untuk menganalisis tanggal untuk membuat struktur gelombang kosong tingkat tinggi awal (tanpa aplikasi yang ditetapkan). Pertimbangkan pertanyaan-pertanyaan berikut:

- Berapa total panjang program migrasi?
- Apa tenggat waktu?
- Apakah ada tanggal keluar pusat data tetap?
- Apakah ada tanggal akhir kontrak kolokasi?
- Apa siklus penyegaran aplikasi dan infrastruktur?
- Apa siklus pemeliharaan dan rilis aplikasi?
- Apakah ada tanggal ketika migrasi harus dihindari (misalnya, siklus rilis dan pemeliharaan, akhir tahun, hari libur, pemrosesan akhir bulan)?

Dengan pertimbangan ini, plot gelombang ke dalam rencana. Untuk mempercepat proses migrasi, kami merekomendasikan gelombang yang tumpang tindih jika memungkinkan. Kunci untuk gelombang yang tumpang tindih adalah mendefinisikan dan mempertimbangkan apa yang terjadi dalam gelombang. Biasanya, aktivitas penyebaran, validasi infrastruktur target, dan sinkronisasi data akan terjadi selama paruh pertama gelombang. Babak kedua akan fokus pada migrasi aktual, pengujian, dan serah terima operasional. Ini berarti bahwa tim yang berbeda terlibat dalam setiap setengah proses, dan Anda dapat memperoleh beberapa efisiensi. Misalnya, segera setelah tim yang terlibat dalam persiapan infrastruktur target menyelesaikan pekerjaan mereka, mereka dapat mulai mengerjakan persyaratan gelombang berikutnya. Secara umum, lebih disukai bahwa sebagian besar gelombang memiliki panjang dan struktur yang sama untuk memfasilitasi pendekatan migrasi seperti pabrik. Namun, selama proses perencanaan gelombang, ukuran gelombang tertentu dapat diperpanjang untuk memenuhi dependensi atau persyaratan operasional.

Selanjutnya, berdasarkan kelompok ketergantungan yang telah diidentifikasi, tentukan ukuran maksimum gelombang dalam hal jumlah kelompok ketergantungan yang dapat dikandungnya. Ukuran gelombang biasanya ditentukan oleh selera risiko (misalnya, berapa banyak perubahan paralel yang dapat ditoleransi), dan ketersediaan sumber daya (misalnya, berapa banyak perubahan paralel yang dapat dilakukan dengan sumber daya, keterampilan, dan anggaran yang tersedia). Namun, selama perencanaan awal, jangan dibatasi oleh persyaratan dan ketersediaan sumber daya. Gelombang yang mengandung lebih dari satu kelompok ketergantungan dapat didekomposisi menjadi gelombang yang lebih kecil dalam iterasi future.

Setelah kelompok dependensi untuk gelombang tertentu telah dikonfirmasi, tinjau persyaratan sumber daya untuk memigrasikan gelombang. Pertimbangkan untuk menyesuaikan ukuran gelombang (jumlah kelompok ketergantungan yang dikandungnya) berdasarkan kebutuhan sumber daya. Hal ini dapat menyebabkan gelombang yang lebih kecil atau lebih besar. Ulangi rencana gelombang sesuai kebutuhan sampai semua gelombang telah ditentukan. Pertimbangkan untuk

bekerja dengan Layanan AWS Profesional atau Mitra Kompetensi AWS Migrasi, yang dapat menyediakan spesialis untuk membantu Anda selama proses berlangsung.

Mengelola perubahan

Portofolio aplikasi dan infrastruktur terkait akan berubah selama siklus hidup program migrasi. Long-running Program migrasi hidup berdampingan dengan evolusi dan perubahan bisnis normal. Aplikasi terus berkembang saat menunggu untuk dimigrasi. Server ditambahkan atau dihapus, infrastruktur baru digunakan di tempat. Diharapkan bahwa ruang lingkup gelombang atau kelompok ketergantungan akan membutuhkan perubahan. Perubahan diperlukan terutama ketika, lebih dekat ke tanggal migrasi, ketergantungan yang sebelumnya tidak diketahui diidentifikasi, atau server baru disertakan dalam inventaris. Terkadang ini bisa terjadi selama migrasi itu sendiri.

Perubahan lingkup mempengaruhi kelompok ketergantungan dan gelombang. Untuk menangani perubahan dan meminimalkan dampak, penting untuk menetapkan mekanisme kontrol ruang lingkup. Mekanisme kontrol perubahan ruang lingkup membutuhkan definisi satu sumber kebenaran untuk ruang lingkup. Ini bisa menjadi alat untuk mengelola ruang lingkup, atau file.csv, spreadsheet, atau database, seperti yang didefinisikan oleh tata kelola program migrasi. Anda harus mengidentifikasi perubahan, menganalisis dampak, dan mengkomunikasikan perubahan kepada pemangku kepentingan yang relevan sehingga mereka dapat mengambil tindakan. Rencana gelombang akan diulang sebagai hasilnya.

Kasus bisnis terperinci

Pada tahap ini, kami merekomendasikan untuk memvalidasi dan memperluas ruang lingkup kasus bisnis untuk memberikan tingkat detail yang lebih besar untuk mendukung program transformasi. Kasus bisnis terarah awal yang dirakit dengan cepat dirancang untuk memberikan kepercayaan yang cukup untuk berinvestasi dalam langkah-langkah dasar dan tingkat perencanaan terperinci berikutnya.

Mengembangkan kasus bisnis terperinci mendukung proses perencanaan ini dengan cara-cara berikut:

- Memberikan analisis keuangan yang menginformasikan keputusan tentang apa yang harus dimigrasikan dan dimodernisasi, opsi mana yang harus dipilih dan bagaimana melakukan fase dan memprioritaskan pekerjaan
- Memvalidasi, menyempurnakan, dan mengembangkan kasus keuangan terarah asli dengan memeriksa ulang secara rinci:

    - Potensi pengurangan biaya infrastruktur
    - Produktivitas TI internal dan efisiensi operasi outsourcing
    - Perkiraan untuk investasi yang diperlukan untuk pengaturan program, migrasi, dan modernisasi
- Mengidentifikasi, memperkirakan skala, dan menyiapkan proses untuk melacak driver nilai lebih lanjut yang dibawa migrasi

Dalam kasus bisnis terperinci, Anda menetapkan yang berikut:

- Dasar obyektif untuk mengamankan mandat dan investasi untuk melaksanakan setidaknya tahap pertama migrasi
- Harapan kinerja keuangan minimum dasar untuk program
- Kejelasan atas dasar keuangan di mana berbagai desain migrasi dan keputusan prioritas dibuat, sehingga ketika keadaan dan orang-orang berubah selama program, kepemimpinan baru dapat membuat pilihan berdasarkan informasi.
- Wawasan tentang area tambahan pengoptimalan biaya yang akan dieksplorasi setelah data penggunaan awal tersedia saat beban kerja dimigrasikan dan mulai beroperasi
- Perkiraan nilai yang dibawa transformasi cloud ke bisnis dari peningkatan ketahanan dan kelincahan
- KPI, metrik, dan asumsi terkait digunakan untuk memperkirakan pengembalian finansial dari peningkatan ketahanan dan kelincahan, yang kemudian membentuk dasar untuk mendorong realisasi manfaat utama keluar dari program

Tentukan skenario yang diperlukan untuk kasus ini

Ketika membangun kasus bisnis terperinci, biasanya perlu untuk mengembangkan beberapa skenario untuk mendukung berbagai tujuan yang digunakan untuk kasus bisnis.

Skenario perubahan minimum — Untuk menilai ekspektasi kinerja keuangan minimum, siapkan skenario yang mengasumsikan perubahan minimum yang diharapkan pada status quo. Skenario ini, sebagai backstop kasus terburuk, memberikan dukungan yang berguna saat mendapatkan mandat untuk berinvestasi dalam migrasi. Skenario ini memodelkan tingkat pertumbuhan kapasitas minimum yang diharapkan dan perubahan minimum untuk kebutuhan kualitas layanan lainnya, seperti ketersediaan dan ketahanan. Perubahan terkecil menciptakan biaya terendah dan inefisiensi sumber daya paling sedikit untuk model operasi saat ini.

Skenario yang paling mungkin — Untuk menginformasikan strategi program dan keputusan prioritas, siapkan skenario yang mencerminkan apa yang diharapkan bisnis terjadi. Skenario ini harus mencakup kemungkinan pertumbuhan atau pengurangan pemanfaatan puncak dan biaya peningkatan untuk memenuhi permintaan akan kualitas layanan tingkat tinggi (terutama ketersediaan dan ketahanan) dari bisnis.

Skenario spesifik lainnya — Di mana masih perlu membuat asumsi yang dapat berdampak besar pada kasus bisnis, kembangkan skenario di mana asumsi tersebut benar atau tidak. Namun, kami sarankan untuk menjaga jumlah skenario alternatif ini pada minimum absolut. Lebih dari tiga-empat skenario dalam kemajuan total lambat, dan menjadi mahal, membingungkan, dan sulit dipertahankan. Jika memungkinkan, lakukan eksperimen dan bekerja untuk menghilangkan asumsi yang lebih besar.

Memvalidasi dan menyempurnakan infrastruktur dan model biaya migrasi

Setelah Anda menyelesaikan analisis portofolio dan menyiapkan desain dan ukuran target Layanan AWS, perbaiki perkiraan biaya operasional untuk model operasi saat ini (COM) dan model operasi future (FOM) AWS untuk setiap skenario. Biasanya perlu untuk menyempurnakan perkiraan untuk hal-hal berikut:

- Biaya infrastruktur COM dari server host hypervisor, server bare-metal, penyimpanan, perangkat jaringan, penyegaran perangkat keras alat keamanan, instalasi, dan pemeliharaan. Hitung ini dengan harga aktual dan tingkat diskon untuk kapasitas yang dibutuhkan untuk skenario tersebut.
- Biaya pusat data COM dan fasilitas kolokasi, termasuk ruang, pendinginan, daya, rak, catu daya tak terputus (UPS), pemasangan kabel, sistem keamanan fisik, ukuran untuk pertumbuhan dan ditentukan untuk memenuhi kapasitas, dan tingkat ketersediaan dan pemulihan bencana (DR) yang tinggi untuk skenario tersebut.
- Biaya layanan jaringan COM, termasuk biaya untuk tautan WAN, jaringan pengiriman konten, dan jaringan pribadi virtual (VPN), dihitung menggunakan harga kontrak untuk konektivitas, bandwidth, throughput, dan kebutuhan latensi untuk skenario tersebut.
- Biaya aplikasi dan perangkat lunak infrastruktur COM berdasarkan kontrak yang ada untuk memberikan pertumbuhan atau pengurangan penggunaan untuk skenario tersebut.
- Biaya AWS utilitas FOM, termasuk dukungan teknis dan layanan terkelola sesuai kebutuhan, berdasarkan arsitektur layanan yang disempurnakan, ukuran instans, model harga pilihan, penggunaan yang diharapkan, dan volatilitas penggunaan.
- Lisensi aplikasi FOM berdasarkan desain aplikasi akhir, konfigurasi infrastruktur yang menjalankan aplikasi, pertumbuhan dari waktu ke waktu, dan aturan transferabilitas lisensi.

- Perkiraan biaya migrasi dan modernisasi FOM, disempurnakan untuk mencerminkan rencana gelombang migrasi dasar untuk skenario tersebut, dan dirinci untuk menyediakan biaya untuk setiap beban kerja, terutama bagi mereka yang akan direplatform, dibeli kembali, atau difaktorkan ulang.
- Biaya penonaktifan FOM, termasuk perkiraan penghapusan aset dan biaya penghentian awal kontrak, direvisi untuk mencerminkan waktu penonaktifan dalam rencana gelombang migrasi dasar, verifikasi aset apa yang dapat digunakan kembali dan aset apa yang dapat dialihkan untuk meminimalkan penghapusan, dan biaya pembuangan aset fisik dan media.
- Biaya proses paralel migrasi disempurnakan untuk mencerminkan waktu setiap pemotongan migrasi dan setiap penonaktifan layanan yang ada.

Menyempurnakan produktivitas TI dan operasi TI serta mendukung model nilai efisiensi

Seperti halnya kasus bisnis terarah, ada dua pendekatan utama untuk menyempurnakan dan mengembangkan model nilai seputar operasi dan dukungan TI. Pendekatan yang Anda pilih tergantung pada apakah COM dikelola secara internal atau dengan kontraktor atau layanan outsourcing:

Peningkatan produktivitas tim internal

Di mana operasi dan dukungan TI dikelola di rumah, fokus kasus bisnis adalah sebagai berikut:

- Mengidentifikasi dan mengukur keuntungan produktivitas dari migrasi dan otomatisasi operasional apa pun yang termasuk dalam ruang lingkup
- Memvalidasi bahwa waktu yang dibebaskan untuk tim internal dapat dengan mudah dan produktif diterapkan pada kegiatan lain yang biasanya bernilai lebih tinggi, memberikan peluang untuk kemajuan dan penghargaan yang lebih besar kepada tim dan nilai lebih bagi organisasi

Menilai analisis kebutuhan manfaat tentang berapa banyak waktu yang dihabiskan setiap anggota dalam setiap peran dalam tim untuk berbagai kegiatan rutin mereka, dan panduan tentang pengurangan beban kerja yang diharapkan untuk kegiatan yang berbeda.

Tabel berikut memberikan panduan awal untuk tingkat tipikal pengurangan beban kerja berdasarkan aktivitas untuk tugas-tugas yang menghabiskan sebagian besar operasi TI dan upaya dukungan di berbagai peran dalam tim. Tabel ini mencakup deskripsi tentang bagaimana produktivitas dicapai.

> **Note**
>
> Kegiatan yang tercantum biasanya dilakukan oleh anggota tim dalam beberapa peran yang berbeda, sehingga penghematan produktivitas untuk setiap tugas harus dinilai di seluruh set lengkap peran dalam tim. Misalnya, dalam tim operasi TI yang diselenggarakan oleh menara infrastruktur (seperti komputasi, penyimpanan, dan jaringan), perencanaan belanja modal dan penganggaran mungkin umum untuk menara lead untuk setiap menara.

Kegiatan operasional dan dukungan	Tingkat tabungan	Penggerak produktivitas
Desain infrastruktur	Sedang	Desain disederhanakan, dengan parameter yang lebih sedikit untuk dipertimbangkan.
Perencanaan dan penganggaran belanja modal	Tinggi	OPEX-centric layanan elastis menghapus hampir semua masalah penganggaran dan perencanaan.
Pembelian	Tinggi	Pengadaan sangat disederhanakan setelah Akun AWS ditetapkan.
Perencanaan kapasitas	Medium-very tinggi	Jaringan dan beban kerja manajemen kapasitas komputasi biasanya dihilangkan, dan untuk penyimpanan sangat disederhanakan
Penyetelan	High-very tinggi	Tuning tidak diperlukan untuk layanan terkelola dan hampir tidak diperlukan untuk layanan lain karena instance dapat diubah ukurannya kapan saja.

Mengelola kegagalan perangkat keras	Sangat tinggi	Semua aspek penangana n perangkat keras di cloud ditangani secara transparan oleh. AWS
Memantau ketersediaan dan komunikasi server	Tinggi	Pemantauan dan komunikas i disederhanakan secara ekstensif dengan dukungan AWS alat dan otomatisasi.
Manajemen keamanan	Sedang	Beban kerja berkurang secara signifikan dengan kemampuan AWS keamanan dan dengan AWS memiliki tanggung jawab keamanan untuk AWS Cloud perangkat keras, perangkat lunak, jaringan, dan fasilitas.
Peningkatan jaringan dan penyimpanan, pemeliharaan, dan tambalan.	Sangat tinggi	Semua aspek pemeliharaan jaringan dan penyimpanan di cloud ditangani secara transparan oleh. AWS
Racking dan susun — logistik perangkat keras	Sangat tinggi	Semua aspek pengelolaan perangkat keras di cloud ditangani secara transparan oleh. AWS
Pencadangan	Sedang	Backup disederhanakan secara ekstensif dengan AWS alat, sistem penyimpanan yang fleksibel, dan otomatisa si.

Layanan terkelola (seperti Amazon S3, Amazon RDS, AWS Lambda dan) AWS Fargate	Sangat tinggi	Layanan terkelola berjalan di lingkungan yang dikelola sepenuhnya oleh AWS, sehingga tidak memerlukan pemeliharaan, penambalan, pemantauan, atau aktivitas manajemen penyediaan.
Pengaturan dan commissio ning perangkat dan layanan	High-very tinggi	Aktivitas untuk pengatura n perangkat keras untuk perkebunan yang dimigrasi ke AWS biasanya dikurangi , kecuali untuk perangkat konektivitas WAN untuk membangun VPN atau AWS Direct Connect koneksi ke AWS pusat data.
Perlindungan titik akhir dan perlindungan antivirus	Tinggi	Aplikasi dan pemeliharaan perlindungan endpoint dan layanan antivirus biasanya secara ekstensif otomatis sebagai bagian dari desain migrasi.
Ancaman, kerentanan, dan penilaian risiko	Tinggi	AWS memberikan dukungan untuk elemen-elemen ini, berfokus pada platform inti dan mekanisme yang AWS menyediakan untuk mengamankan arsitektur menyederhanakan penilaian.

Manajemen proyek infrastruktur pusat data	Tinggi	Manajemen proyek untuk pekerjaan instalasi untuk perluasan, penyegaran, atau penonaktifan layanan infrastruktur. Sementara beberapa manajemen perangkat lunak dan layanan infrastruktur tetap ada, ini jauh lebih sederhana daripada infrastruktur lokal, dan aktivitas perangkat keras dihilangkan.
Manajemen fasilitas pusat data	Medium-very tinggi	Pekerjaan manajemen fasilitas yang dapat dikaitkan dengan semua server, perangkat penyimpanan, peralatan keamanan, dan rak terkait dihapus untuk semua yang dimigrasikan. Namun, beberapa pekerjaan biasanya tetap untuk menyediakan fasilitas untuk perangkat jaringan tautan WAN dan untuk infrastruktur apa pun yang disimpan di tempat dalam arsitektur hibrida.

Arsitektur aplikasi, pengembangan, manajemen, dan pengujian	Rendah	Penggunaan rantai alat pengembangan tangkas, dikombinasikan dengan otomatisasi instantiasi dan penghancuran tumpukan aplikasi untuk membangun lingkungan pengujian sesuai kebutuhan, mengurangi waktu tunggu pengembangan aplikasi dan menghilangkan banyak langkah pengujian manual.
Menginstal dan mengonfigurasi perangkat lunak aplikasi	Sedang	Instalasi dan konfigurasi tumpukan aplikasi lengkap siap diotomatisasi menggunakan layanan seperti AWS CloudFormation dan disederhanakan melalui penggunaan zona pendaratan, yang dapat dengan mudah dikonfigurasi dengan menggunakan. AWS Control Tower

Dukungan TI	Sedang	Pengurangan dukungan L1 dan L2 dicapai dengan mengurangi masalah kapasitas dan kinerja melalui penggunaan kemampuan Service Catalog untuk penyediaan layanan mandiri, peningkatan penggunaan arsitektur ketersediaan tinggi berbiaya rendah (mengurangi pemadaman dan mengonfig urasi penskalaan otomatis dan komputasi tepi).
Administrasi basis data	Minimal-low	Kegiatan ini sebagian besar tetap tidak berubah. Mereka biasanya sumber daya pada tingkat yang AWS sama untuk infrastruktur lokal.
Pengambilan, analisis, dan desain persyaratan infrastru ktur dan keamanan	Minimal	
Dokumentasi	Minimal	
Aplikasi dan pemantauan kinerja	Minimal	
Dukungan teknis L3, menjawab pertanyaan, dan pemecahan masalah dan pemecahan masalah	Minimal	
Menginstal dan mengonfig urasi perangkat lunak aplikasi	Minimal	
Dukungan aplikasi L3 (tidak termasuk penganggaran dan perencanaan kapasitas jangka panjang)	Minimal	

Tabel berikut menunjukkan penghematan yang diharapkan untuk setiap tingkat pengurangan beban kerja.

Tingkat	Expected
Sangat tinggi	85% - 100%
Tinggi	60% - 90%
Sedang	30% - 70%
Rendah	10% - 35%
Minimal	0% - 10%

Metrik ini memberikan titik awal untuk menilai peningkatan produktivitas dan memasukkannya ke dalam kasus bisnis terperinci. Keuntungan produktivitas aktual bervariasi berdasarkan situasi tertentu. Ini dapat berguna untuk menghitung penghematan produktivitas di titik tengah dan ujung bawah rentang untuk memperkirakan skenario tipikal dan konservatif.

Seiring kemajuan program, penting untuk menangkap data aktual untuk waktu yang dihabiskan pada setiap aktivitas berdasarkan peran. Data tersebut membangun basis yang lebih baik untuk memperkirakan operasi dan mendukung biaya untuk proyek baru dan perluasan layanan.

Mengalihdayakan operasi TI dan mendukung pengurangan biaya

Di mana operasi dan dukungan TI terutama dialihdayakan atau dikelola dengan kontraktor, alokasi biaya untuk model operasi future (FOM) dapat disiapkan dengan meminta penawaran dari AWS Mitra yang menawarkan solusi layanan terkelola, termasuk. AWS Partner-led AWS Managed Services Anda juga dapat menghubungi manajer AWS akun Anda dan meminta harga untuk AMS secara langsung, seperti yang dijelaskan dalam ayat tentang Membangun optimalisasi biaya operasional dalam bagian Membuat kasus bisnis terarah.

Untuk kasus bisnis terperinci, ganti angka tolok ukur apa pun dengan kutipan berdasarkan Layanan AWS tagihan bahan yang direvisi dan konsumsi layanan yang diharapkan, paket AMS dan opsi apa pun yang diperlukan, dan tingkat layanan yang diperlukan. Biaya akan memiliki komponen implementasi satu kali dan run rate berbasis konsumsi.

Sertakan operasi TI yang tersisa, dukungan yang harus dipertahankan untuk layanan apa pun yang tidak akan dimigrasikan AWS, dan biaya satu kali jika ada penalti kontrak (misalnya, untuk penghentian dini).

Mengembangkan model nilai ketahanan

Pada AWS, Anda dapat membangun berbagai ketersediaan tinggi, pemulihan bencana, dan arsitektur toleran kesalahan. Consumption-based Penetapan harga berarti bahwa layanan hanya dikenakan biaya saat digunakan. Bersama-sama, kedua faktor ini memberikan kinerja biaya yang luar biasa untuk ketahanan.

Selain itu, AWS pelanggan telah menggunakan ini untuk meningkatkan ketahanan beban kerja mereka. Survei IDC 2018 memberikan contoh pelanggan yang berpartisipasi mencapai 73 persen lebih sedikit pemadaman per tahun, pengurangan 58 persen dalam mean time to recover (MTTR) dan penurunan 94 persen dalam produktivitas yang hilang. Survei yang sama menunjukkan bahwa manfaat finansial yang diperoleh melalui peningkatan ketahanan adalah 50 persen lebih besar daripada manfaat pengurangan biaya infrastruktur TI.

Selain itu, ketahanan lebih lanjut dicapai melalui modernisasi siklus hidup pengembangan perangkat lunak untuk aplikasi. Di mana CI/CD jaringan pipa dengan otomatisasi pengujian diperkenalkan untuk mendukung kelincahan bisnis yang lebih besar, cacat perangkat lunak ditangkap lebih awal dalam siklus pengembangan, sangat mengurangi biaya pemeliharaan perangkat lunak.

Untuk menilai dan memasukkan nilai ini dalam kasus bisnis, pertama-tama bekerja dengan pemilik bisnis aplikasi untuk membangun gambaran peluang manfaat total untuk setiap beban kerja yang akan dimigrasikan. Ini mungkin termasuk sebagai item berikut:

- Jumlah, durasi rata-rata, dan sifat interupsi dalam layanan
  - Contoh gangguan layanan termasuk pemadaman, perlambatan kinerja, batch terencana dan pemeliharaan jendela overrunning, bug dalam fungsi utama, dan pembatasan akses selama periode puncak.
- Dampak pada pendapatan oleh gangguan layanan yang menghasilkan pendapatan, seperti sistem e-commerce
  - Kemungkinan jumlah transaksi yang tidak dapat diselesaikan melalui gangguan layanan, berdasarkan waktu interupsi dan tingkat transaksi
  - Nilai rata-rata untuk setiap transaksi yang terkena dampak
- Biaya tambahan waktu insinyur pendukung untuk menyelesaikan cacat dalam sistem produksi dibandingkan dengan biaya untuk menemukannya di awal proses pengembangan

- Dampak pada produktivitas pengguna internal dan biaya waktu yang hilang

Kemudian buat penilaian pengurangan waktu yang diharapkan dan lebih konservatif yang hilang karena gangguan layanan yang harus dihasilkan oleh peningkatan ketahanan. Misalnya, pertimbangkan untuk memasukkan item-item berikut:

- Mengurangi jumlah pemadaman dan MTTR menggunakan arsitektur ketersediaan tinggi dan peningkatan tujuan waktu pemulihan (RTO) dan tujuan titik pemulihan (RPO)
- Pengurangan perlambatan, penghapusan pelambatan kapasitas dan penghindaran dalam kelebihan pemrosesan batch, menggunakan kemampuan seperti penskalaan otomatis
- Mengurangi jumlah bug aplikasi yang hanya ditemukan dalam produksi, melalui implementasi CI/CD jaringan pipa dan pengujian regresi otomatis pada infrastruktur yang diputar dan diputar ke bawah untuk meminimalkan biaya

Satukan ini untuk portofolio aplikasi yang akan dimigrasikan dan dimodernisasi, dan hitung angka nilai bisnis yang diharapkan dan lebih konservatif untuk setiap tahun kasus. Manfaat harus meningkat sejalan dengan jadwal migrasi dan kemudian meningkatkan volume sesuai dengan ekspektasi pertumbuhan penggunaan dari aplikasi yang berkontribusi.

Mengembangkan model nilai kelincahan bisnis

Kelincahan bisnis adalah alasan utama AWS pelanggan bermigrasi. AWSSurvei AWS pelanggan IDC 2018 menunjukkan bahwa bagi mereka, manfaat kelincahan bisnis menyumbang 47 persen dari total manfaat yang diukur dan lebih dari lima kali manfaat yang diperoleh dari pengurangan biaya infrastruktur.

Memprediksi secara akurat semua manfaat kelincahan bisnis yang akan diperoleh dari transformasi apa pun adalah tantangan. Namun, dengan berfokus pada aplikasi yang mendukung sejumlah besar pengguna atau merupakan sumber diferensiasi bisnis, Anda dapat memodelkan dan memasukkan bagian material dari manfaat ini ke dalam kasus bisnis rinci dasar.

Ketika migrasi berlangsung, secara bertahap menyempurnakan dan memperluas model nilai kelincahan bisnis karena lebih banyak manfaat menjadi terukur. Ini membuat kasus bisnis tetap relevan, sehingga dapat digunakan sebagai alat pendukung keputusan utama untuk mengarahkan program.

Untuk membangun model nilai kelincahan bisnis, gunakan panduan berikut:

- Pilih beban kerja yang memiliki kesempatan untuk mendorong peningkatan kinerja bisnis terbesar, seperti:
  - Revenue-generating beban kerja
  - Beban kerja operasi bisnis dengan ruang lingkup untuk mendorong peningkatan efisiensi dan menghilangkan biaya dari bisnis
  - Alat produktivitas bisnis yang mendukung basis pengguna yang besar
- Untuk beban kerja yang menghasilkan pendapatan dan efisiensi, lakukan hal berikut:
  - Buat penilaian yang realistis dan lebih konservatif terhadap pertumbuhan pendapatan atau efisiensi operasional yang diharapkan dapat didorong oleh peningkatan aplikasi besar dan kecil.
  - Perkirakan peningkatan jumlah rilis mayor dan minor per tahun yang AWS meningkatkan kecepatan pengembangan aplikasi dan mengurangi waktu penyebaran infrastruktur memungkinkan. Beberapa metrik dasar untuk ini disediakan dalam laporan IDC.
  - Hitung ekspektasi manfaat yang realistis dan lebih konservatif. Petakan mereka selama periode kasus bisnis, membuat tunjangan untuk meningkatkan efisiensi penuh beberapa saat setelah beban kerja masing-masing dimigrasikan.
- Untuk alat produktivitas bisnis, lakukan hal berikut:
  - Buat penilaian yang realistis dan lebih konservatif tentang penghematan waktu yang diharapkan dapat didorong oleh peningkatan aplikasi besar dan kecil.
  - Perkirakan biaya rata-rata waktu dan upaya orang di seluruh basis pengguna yang terkena dampak.
  - Gunakan angka untuk meningkatkan frekuensi rilis mayor dan minor, dan hitung manfaatnya selama jangka waktu kasus bisnis.

Karena peningkatan produktivitas pengembang dan pengurangan waktu untuk peluncuran tidak memerlukan sumber daya tambahan, tambahkan garis manfaat bersih untuk setiap beban kerja ke dalam model arus kas kasus bisnis untuk dimasukkan dalam arus kas diskon, NPV, ROI, MIRR, dan perhitungan pengembalian.

Penilaian dan peningkatan berkelanjutan

Tahap penilaian ini berfokus pada dua aspek:

- Penilaian aplikasi terperinci yang sedang berlangsung, untuk setiap gelombang aplikasi
- Evolusi berkelanjutan dan peningkatan portofolio Anda

Aspek pertama, penilaian aplikasi terperinci yang sedang berlangsung, berfokus pada penemuan dan analisis terperinci, hingga tingkat arsitektur dan teknologi, untuk sepenuhnya memahami setiap aplikasi dalam gelombang tertentu, AWS desain yang diusulkan, dan strategi migrasi. Penilaian kesiapan migrasi ini merupakan prasyarat untuk memulai gelombang migrasi tertentu.

Aspek kedua, evolusi berkelanjutan dan peningkatan portofolio Anda, berfokus pada manajemen portofolio dan bagaimana Anda berencana untuk meningkatkan aplikasi dari waktu ke waktu, termasuk evolusi dan pelacakan kasus bisnis.

Hasil migrasi utama dari tahap ini meliputi:

- Cakupan migrasi yang divalidasi untuk setiap gelombang
- Arsitektur target yang terdokumentasi dan strategi migrasi untuk aplikasi dalam gelombang migrasi tertentu
- Pola dan perkakas migrasi yang diidentifikasi dan divalidasi
- Persyaratan terdokumentasi (keamanan, AWS infrastruktur, dan operasi) dan pertimbangan cut-over migrasi untuk setiap gelombang

Hasil optimasi utama dari tahap ini meliputi:

- Model rasionalisasi portofolio dan hasil bisnis
- Usulan perubahan arsitektur dan teknologi, dan manfaat yang diharapkan
- Persyaratan platform (keamanan, AWS infrastruktur, dan operasi)
- Rencana implementasi

Memahami persyaratan data penilaian berkelanjutan

Persyaratan data untuk penilaian berkelanjutan dan peningkatan portofolio aplikasi adalah kombinasi dari persyaratan data dari bagian sebelumnya. Untuk terus mengelola migrasi portofolio dan evolusinya, lihat bagian berikut untuk memahami persyaratan data:

- Untuk penilaian gelombang dan optimalisasi aplikasi, gunakan persyaratan data dari bagian penilaian aplikasi yang diprioritaskan.
- Untuk manajemen portofolio berkelanjutan, gunakan persyaratan data dari analisis Portofolio dan bagian perencanaan migrasi.
- Untuk menentukan rencana gelombang, lihat bagian Perencanaan gelombang.

Penilaian gelombang terperinci

Penilaian rinci aplikasi, menjelang gelombang migrasi dan sebagai enabler utama untuk migrasi, memiliki persyaratan dan rekomendasi yang sama dengan tahap penilaian aplikasi yang diprioritaskan. Tujuannya adalah untuk memahami secara rinci keadaan aplikasi saat ini dalam gelombang tertentu, dan untuk menghasilkan desain arsitektur negara masa depan dan strategi migrasi, termasuk aspek operasional, perkakas, dan pola migrasi tertentu.

Terapkan penilaian aplikasi yang diprioritaskan ke kelompok aplikasi dalam gelombang tertentu. Ulangi proses ini sebelum setiap gelombang dalam rencana migrasi Anda. Kuncinya adalah menjadwalkan waktu yang cukup di antara penilaian terperinci dan awal gelombang. Jumlah waktu yang dibutuhkan akan ditentukan oleh persyaratan platform dan tim migrasi yang menerapkan persyaratan gelombang dan melakukan migrasi. Bekerja dengan tim-tim tersebut untuk menjadwalkan penilaian gelombang terperinci dan gelombang. Kami merekomendasikan penerapan model seperti pabrik yang meniru jalur produksi.

Penilaian untuk optimalisasi dan modernisasi

Proses penilaian untuk optimasi beban kerja dan modernisasi yang sudah dimigrasikan mirip dengan penilaian beban kerja yang akan AWS dimigrasikan. AWS Yang akan berubah, terutama, adalah sumber data untuk melakukan penilaian. Di AWS, ada beberapa alat dan layanan out-of-the-box yang dapat Anda gunakan untuk mendapatkan informasi lebih lanjut tentang aplikasi Anda berjalan. AWS

Apa dan bagaimana mengoptimalkan dan memodernisasi aplikasi Anda akan didasarkan pada driver dan keadaan unik Anda. Optimasi berfokus pada penerapan perubahan pada arsitektur dan teknologi

saat ini untuk mengurangi biaya, menyesuaikan persyaratan kinerja, dan untuk menggabungkan pelajaran yang dipetik. Modernisasi berfokus pada membawa aplikasi Anda ke tingkat berikutnya, seperti mengadopsi model tanpa server dan arsitektur layanan mikro.

Ikuti pedoman penilaian aplikasi yang diprioritaskan. Untuk lebih membantu upaya optimasi dan modernisasi Anda, lihat sumber daya berikut:

- AWS pengoptimalan biaya memberikan informasi tentang pengoptimalan TI dan penghematan biaya TI Anda.
- AWS Compute Optimizermerekomendasikan AWS sumber daya untuk beban kerja Anda untuk mengurangi biaya dan meningkatkan kinerja dengan menggunakan pembelajaran mesin untuk menganalisis metrik pemanfaatan historis.
- AWS Layanan dan alat pengoptimalan biaya dan kapasitas membantu mengelola sumber daya komputasi sehingga Anda dapat menghabiskan lebih banyak waktu untuk membangun dan lebih sedikit waktu mengelola biaya komputasi
- Amazon S3 Storage Lens memberikan visibilitas seluruh organisasi ke dalam penggunaan penyimpanan objek dan tren aktivitas. Itu membuat rekomendasi yang dapat ditindaklanjuti untuk meningkatkan efisiensi biaya dan menerapkan praktik terbaik perlindungan data.
- Kebebasan Database memfasilitasi migrasi ke AWS database dan layanan analitik.
- Amazon CodeGuru adalah alat pengembang yang memberikan rekomendasi cerdas untuk meningkatkan kualitas kode dan mengidentifikasi baris kode aplikasi yang paling mahal.
- AWS Layanan cloud hybrid memberikan AWS pengalaman yang konsisten di mana pun Anda membutuhkannya—dari cloud, hingga di tempat, dan di tepi.

Sumber daya tambahan

- Optimalisasi biaya dan inovasi: Pengantar modernisasi aplikasi (posting blog)
- Mengoptimalkan biaya aplikasi web tanpa server (posting blog)
- Windows di AWS (blog)
- Aplikasi modern
- Modernisasi aplikasi (AWS re: Invent 2020)
- AWS panduan microservices

Mengulangi rencana gelombang

Ketika program migrasi bergerak maju dan lebih banyak gelombang bermigrasi, itu adalah kunci untuk mengembangkan rencana gelombang migrasi berdasarkan pelajaran yang dipetik dan mengubah prioritas bisnis. Secara khusus, untuk program migrasi yang berjalan lama, penting untuk menilai kembali pendorong bisnis dan perubahan organisasi, dan untuk memastikan bahwa rencana gelombang migrasi masih valid.

Demikian pula, pelajaran yang dipetik dari migrasi akan mempengaruhi komposisi rencana gelombang dan ruang lingkup setiap gelombang. Untuk menghindari kehilangan visibilitas terhadap apa yang terjadi, perbarui rencana gelombang. Rencana tersebut harus mencerminkan dan melacak apa yang sedang disampaikan, dan harus mengelola dan menilai perubahan pada ruang lingkup migrasi.

Mengembangkan dan melacak kasus bisnis

Ketika migrasi berlangsung, terutama untuk program yang berjalan lama, tidak dapat dihindari bahwa tekanan bisnis akan menyebabkan prioritas migrasi dan modernisasi diperiksa ulang secara teratur.

Kami menyarankan Anda berdua mengembangkan kasus bisnis saat informasi baru tersedia, dan Anda melacak kinerja komersial aktual terhadap ekspektasi yang didokumentasikan dalam kasus bisnis terperinci. Rekomendasi ini meliputi:

- Perubahan struktural baru dalam organisasi yang memengaruhi prioritas bisnis dan memengaruhi strategi TI dan portofolio aplikasi dengannya
- Meningkatnya kepentingan komersial dari satu bagian dari portofolio aplikasi atau perubahan yang ditargetkan untuk dicapai oleh migrasi dan modernisasi
- Ketersediaan data pemanfaatan sumber daya aktual untuk aplikasi yang dimigrasi, termasuk menyempurnakan ukuran dan mengukur dan mengonfirmasi kasus untuk modernisasi inkremental
- Ketersediaan data tentang upaya yang dikonsumsi dalam operasi TI dan kegiatan pendukung, dan analisis kemungkinan peningkatan operasional dan otomatisasi
- Ketersediaan data yang mengukur perubahan dalam pengembangan perangkat lunak dan waktu siklus pemeliharaan, cacat perangkat lunak berdasarkan tahap pengembangan dan informasi ketersediaan layanan, dan analisis akar penyebab untuk area yang terbuka untuk perbaikan lebih lanjut

Dengan melacak kinerja terhadap kasus bisnis, Anda dapat mengembangkan kasus untuk menyertakan perbaikan lebih lanjut yang dapat lebih mudah dinilai dan diukur setelah migrasi dimulai. Organisasi tata kelola program jauh lebih siap untuk menanggapi tekanan bisnis yang berubah dan mengarahkan transformasi ke arah yang mendorong nilai terbesar pada tingkat risiko yang dapat dikelola dan dapat diterima.

Hal ini sangat penting untuk produktivitas TI, ketahanan, dan manfaat kelincahan bisnis dalam kasus ini. Ini biasanya driver yang lebih besar dan lebih sulit untuk dinilai sebelumnya. Dengan melacak kinerja pembalap ini, tim dapat menyelam lebih dalam dan menyelesaikan masalah yang menghambat realisasi manfaat. Atau kasus bisnis dapat disesuaikan untuk memprioritaskan inisiatif yang mencapai optimalisasi kinerja keuangan yang paling berkelanjutan.

Sumber daya

AWS referensi

- Amazon Builders' Library
- Modernisasi aplikasi (AWS re:Invent 2025)
- Strategi penilaian portofolio aplikasi
- AWS Pusat Arsitektur
- AWS Compute Optimizer
- AWS layanan dan alat pengoptimalan biaya dan kapasitas
- AWS optimalisasi biaya
- Optimalisasi biaya dan inovasi: Pengantar modernisasi aplikasi (posting blog)
- Alat migrasi Penemuan, Perencanaan, dan Rekomendasi
- AWS Documentation
- Memulai Pusat Sumber Daya
- AWS Marketplace
- AWS Managed Services
- AWS panduan microservices
- AWS Mitra Kompetensi Migrasi
- Aplikasi modern
- Mengoptimalkan biaya aplikasi web tanpa server (posting blog)
- AWS Bimbingan Preskriptif
- AWS Layanan Profesional
- AWS Pustaka Solusi
- Windows di AWS (blog)

Layanan AWS

- AWS App2Container
- AWS Transform MGN

- AWS Transform
- Kiro
- AWS Control Tower
- AWS Database Migration Service
- AWS DataSync
- AWS Direct Connect
- Amazon ECS
- Amazon EKS
- AWS Fargate
- AWS Managed Services
- Penilai Migrasi
- AWS Zona Pendaratan
- AWS Kalkulator Harga
- AWS Schema Conversion Tool
- Lensa Penyimpanan Amazon S3

Riwayat dokumen

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

Perubahan	Deskripsi	Tanggal
Update	Memperluas bagian perencanaan gelombang. Menyederhanakan bagian strategi migrasi 6 Rs.	Juni 2, 2026
Update	Mengganti nama bagian penemuan Portofolio dan perencanaan awal Akseleras i penemuan dan perencanaan awal; memperbarui diagram pohon keputusan.	Mei 20, 2024
—	Publikasi awal	12 November 2021

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