

Guia de preparação SAP-C02

Tudo o que a prova cobra, organizado por domínio: conceitos-chave, comparações que confundem, pegadinhas reais, gatilhos do enunciado, questões comentadas e um plano para chegar pronto ao exame.

4 capítulos por domínio	**28** questões comentadas	**75%** nota de corte
6 provas completas no curso		

ANTES DE COMEÇAR

Como usar este guia

Este guia foi feito para ser estudado junto com as 6 provas completas do curso SAP-C02: o guia explica o que cai e onde estão as armadilhas; as provas treinam o ritmo e mostram o que você ainda não domina.

Quando	O que fazer
Semana 1	Faça o teste diagnóstico (página seguinte) e leia os capítulos 1 e 2. No fim da semana, faça a Prova 1 do curso no tempo do exame.
Semana 2	Leia os demais capítulos, começando pelos domínios em que você errou mais no diagnóstico. Faça as provas 2 e 3.
Semana 3	Releia os alertas de pegadinha e as tabelas de comparação. Faça as provas 4 e 5 e revise cada erro pela explicação.
Semana 4	Estude a folha de cola, faça a Prova 6 e refaça as provas abaixo de 80%. Agende o exame quando estiver acima de 80% em todas (a nota de corte é 75%).

Mapa do exame

Cada prova do curso tem 75 questões e 180 minutos, no formato do exame. Referência de aprovação: 75%.

Domínio (capítulo)	Peso	Neste guia
1. Aceleração da migração e da modernização de workloads	—	17 conceitos · 12 pegadinhas
2. Design de novas soluções	—	20 conceitos · 14 pegadinhas
3. Otimização contínua	—	18 conceitos · 14 pegadinhas
4. Soluções de design de complexidade organizacional	—	18 conceitos · 14 pegadinhas

Como ler os capítulos: os **tópicos-chave** são o que mais cai; a **tabela de memorização** separa o que a prova costuma confundir; os **alertas de prova** mostram as armadilhas das alternativas erradas; os **gatilhos** ligam palavras do enunciado à resposta; e o **checklist** diz se você está pronto para o domínio.

Estude um pouco todo dia, de graça

Instagram @strecht.simulados

Um post de prova por dia: questão comentada, pegadinhas e o "não confunda" de cada certificação.

Seguir no Instagram →

YouTube Strecht Simulados

Shorts diários com pegadinhas de prova e, aos sábados, um simulado comentado.

Inscrever-se no canal →

DIAGNÓSTICO

Você já sabe isso?

Responda às 12 questões sem consultar nada. As respostas, com a explicação e o capítulo para revisar, estão no fim do guia. Comece o estudo pelos capítulos em que errou.

D1 · Aceleração da migração e da modernização de workloads

Na descoberta para a migração, a empresa encontrou aplicações sem usuários há mais de um ano, mantidas apenas por hábito. Qual estratégia de migração deve ser aplicada a elas?

A. Replatform, levando as aplicações para contêineres gerenciados na AWS
B. Rehost, migrando as aplicações para Amazon EC2 sem nenhuma mudança
C. Refactor, reescrevendo as aplicações como microsserviços modernos
D. Retire, desligando-as após confirmar que não são necessárias

D2 · Aceleração da migração e da modernização de workloads

Uma fintech de médio porte está migrando seu core bancário monolítico de um data center on-premises para a AWS. O aplicativo é sensível à latência e usa Oracle Database com centenas de procedures PL/SQL. A diretoria exige que a migração seja feita em poucos meses, com o mínimo possível de refatoração inicial, mas o time de arquitetura quer uma trajetória clara para modernizar gradualmente o core em microsserviços e adotar bancos gerenciados. Qual abordagem o arquiteto de soluções deve recomendar para acelerar a migração agora e, ao mesmo tempo, preparar a modernização futura?

A. Executar um replatform imediato do monólito para Amazon ECS com Fargate, migrando o Oracle para Amazon RDS for Oracle e planejando uma refatoração total.
B. Migrar diretamente o banco Oracle para Amazon Aurora PostgreSQL com conversão automática de schema e procedures via AWS Schema Conversion Tool (AWS SCT) e reescrever o monólito para microsserviços em AWS Lambda antes de migrar para a nuvem.
C. Fazer lift-and-shift do monólito com o AWS MGN e do Oracle com o DMS e colocar uma camada de APIs na frente do monólito para extrair microsserviços aos poucos
D. Manter o Oracle on-premises em longo prazo, criar uma conexão VPN site-to-site com a AWS e desenvolver novos microsserviços em Amazon EKS consumindo o banco remoto

D3 · Aceleração da migração e da modernização de workloads

Uma empresa de marketing digital está migrando dezenas de aplicativos legados para a AWS e deseja acelerar a migração com o mínimo de refatoração inicial, mas mantendo a possibilidade de modernizar os workloads gradualmente no futuro. Qual abordagem é MAIS adequada para atender a esse requisito?

A. Reescrever de pronto os aplicativos para uma arquitetura de microsserviços baseada em contêineres no Amazon EKS, eliminando cada dependência de servidores legados.
B. Migrar os bancos de dados para Amazon Aurora Serverless e manter os aplicativos on-premises acessando remotamente os dados, para só depois mover o código para a AWS.
C. Utilizar AWS Snowball para migrar os dados em massa e manter os aplicativos legados executando-se em instâncias spot da Amazon EC2 sem planejamento de modernização.
D. Usar o AWS MGN para o lift-and-shift e, depois de estabilizar, modernizar aos poucos com RDS, ECS ou Lambda

D4 · Design de novas soluções

Uma aplicação criptografa dados sensíveis no cliente antes de gravá-los e replica esses dados para outra Região. Em caso de desastre, a Região secundária precisa decifrar os dados sem chamar a Região primária. Qual solução é mais adequada?

A. Usar chaves multi-Região do AWS KMS, com réplica da chave na Região secundária
B. Guardar a chave de criptografia em texto aberto junto aos dados replicados
C. Usar uma chave do KMS diferente em cada Região e decifrar pela Região primária
D. Exportar a chave do KMS para um arquivo e copiá-lo manualmente entre Regiões

D5 · Design de novas soluções

Uma empresa de healthtech está projetando uma nova plataforma de telemedicina altamente disponível e resiliente em múltiplas Regiões AWS. Os requisitos incluem: (1) RPO de até 5 minutos e RTO de até 15 minutos para o banco de dados transacional; (2) capacidade de operar mesmo em caso de falha total de uma Região; (3) tráfego HTTP(S) global distribuído para a Região mais saudável/mais próxima; (4) necessidade de minimizar o esforço operacional contínuo. Qual arquitetura atende melhor a esses requisitos?

A. Utilizar um único cluster Amazon Aurora MySQL em uma Região com Multi-AZ, e configurar AWS Global Accelerator apontando para um Application Load Balancer (ALB) nessa mesma Região. Usar backups automáticos de Aurora para recuperação em outra Região em caso de desastre.
B. Configurar um cluster Amazon Aurora MySQL Multi-AZ em uma Região e replicar para instâncias Amazon EC2 com MySQL em outra Região usando scripts customizados de cópia de dados. Utilizar um Network Load Balancer (NLB) regional na Região primária e um Classic Load Balancer na Região secundária, com failover manual via alteração de registros DNS externos.
C. Criar dois clusters independentes Amazon RDS for MySQL, um em cada Região, com replicação assíncrona baseada em binlog configurada manualmente entre eles. Utilizar Route 53 failover routing.
D. Usar o Aurora Global Database com escrita na região primária e leitura na secundária, roteamento por latência no Route 53 para ALBs nas duas regiões e um runbook automatizado de promoção da secundária

D6 · Design de novas soluções

Um sistema de pagamentos usará um banco relacional compatível com PostgreSQL. O plano de continuidade exige RPO de poucos segundos e RTO de cerca de um minuto em caso de perda de uma Região inteira, com o mínimo de administração. Qual solução atende a esses requisitos?

A. Amazon RDS for PostgreSQL Multi-AZ na Região principal, com snapshots copiados a cada hora para outra Região
B. Amazon RDS for PostgreSQL com backups automáticos e restauração em outra Região quando houver desastre
C. PostgreSQL em instâncias Amazon EC2 com replicação lógica configurada entre Regiões pela equipe de banco
D. Amazon Aurora PostgreSQL com Aurora Global Database e uma Região secundária pronta para promoção

D7 · Otimização contínua

Um arquiteto de soluções responsável por um sistema de processamento de pedidos em produção quer implementar um processo de melhoria contínua usando serviços nativos da AWS. O sistema já está estável em produção, mas a diretoria pediu que falhas recorrentes sejam detectadas mais rapidamente e que a equipe consiga testar melhorias de arquitetura com risco mínimo para o ambiente atual. Qual abordagem atende melhor a esses requisitos?

A. Configurar AWS CloudTrail só para auditoria de API e, toda vez que ocorrer uma falha, revisar à mão os logs e aplicar mudanças diretamente nos recursos de produção.
B. Ativar o Config e os logs no CloudWatch, criar alarmes para métricas críticas e testar os ajustes em um ambiente idêntico antes da produção
C. Desligar cada automação existente, concentrar os esforços em criar documentação detalhada em formato de wiki interna e implementar um processo manual de revisão trimestral dos recursos da conta.
D. Migrar de pronto os componentes para uma arquitetura de microsserviços em Amazon EKS e confiar só em logs de aplicação.

D8 · Otimização contínua

O Amazon GuardDuty gera achados de instâncias se comunicando com endereços maliciosos, mas a equipe demora horas para isolar as instâncias. Qual solução melhora a resposta com a menor intervenção manual?

A. Desativar o GuardDuty nas contas que geram mais achados por semana
B. Regra do EventBridge que aciona a automação de isolamento
C. Relatório diário dos achados enviado por e-mail à equipe de segurança
D. Revisar os achados semanalmente em uma reunião do comitê de risco

D9 · Otimização contínua

Instâncias em sub-redes privadas transferem dezenas de terabytes por mês para o Amazon S3 na mesma Região, e a conta mostra custos altos de processamento de dados em gateways NAT. Qual mudança reduz esse custo com a menor alteração?

A. Mover as instâncias para sub-redes públicas com endereços IP públicos
B. Trocar o Amazon S3 por volumes Amazon EBS anexados às instâncias
C. Criar um endpoint de gateway do S3 nas tabelas de rotas privadas
D. Aumentar o número de gateways NAT para distribuir o tráfego entre eles

D10 · Soluções de design de complexidade organizacional

Uma empresa está consolidando seu ambiente multi-conta na AWS e já utiliza AWS Organizations com todas as contas-membro anexadas a uma única organização. A equipe de segurança deseja garantir que: • Nenhuma conta possa desativar o AWS CloudTrail organizacional. • Todas as contas usem logs centralizados em uma conta dedicada de segurança. • Políticas de segurança sejam aplicadas de forma consistente, impedindo que administradores de contas-membro as enfraqueçam. Qual combinação de ações atende melhor a esses requisitos com o menor esforço operacional contínuo?

A. Ativar o recurso de consolidação de faturamento em AWS Organizations e configurar um único CloudTrail na conta de gerenciamento.
B. Habilitar AWS Config em todas as contas-membro, enviar os logs para a conta de segurança e confiar que o AWS Config substituirá a necessidade de CloudTrail organizacional.
C. Criar um CloudTrail separado em cada conta-membro, armazenando os logs em buckets S3 locais, e usar IAM policies em cada conta para impedir que usuários desativem o CloudTrail.
D. Criar um trail organizacional com os logs entregues a um bucket da conta de segurança, com bucket policy restritiva, e SCPs que impedem as contas-membro de alterar o CloudTrail

D11 · Soluções de design de complexidade organizacional

Uma organização utiliza AWS Organizations com dezenas de contas, cada uma gerenciada por uma unidade de negócios diferente. A equipe central de Cloud deseja padronizar o uso de tags obrigatórias (por exemplo, 'Owner', 'CostCenter' e 'Environment') em recursos de produção em todas as contas, garantindo que recursos sem essas tags não possam ser criados. A solução deve exigir o mínimo de esforço operacional contínuo e seguir boas práticas de governança multi-conta. Qual abordagem é MAIS adequada?

A. Definir um padrão de nomenclatura de recursos em um documento interno de governança e exigir que as equipes incluam as tags obrigatórias à mão durante o provisionamento.
B. Configurar AWS Config em cada conta individualmente, criando regras personalizadas para validar tags e aplicar ações de remediação manual quando recursos sem tags forem detectados.
C. Criar um script central em uma conta de ferramentas que liste periodicamente os recursos nas contas, identifique os sem tags obrigatórias e notifique os donos das contas por e-mail.
D. Usar políticas de controle de serviço (SCPs) em nível de organização para negar a criação de recursos que não contenham as tags obrigatórias, combinando condições de tags nas políticas.

D12 · Soluções de design de complexidade organizacional

Uma empresa de consultoria em TI está migrando diversos workloads de clientes para AWS. Cada cliente possui sua própria conta AWS, mas a empresa deseja aplicar políticas de segurança e conformidade padronizadas, além de consolidar o faturamento. A empresa também quer permitir que alguns clientes tenham autonomia para gerenciar suas próprias contas dentro de limites pré-definidos. Qual solução na AWS melhor atende a esses requisitos organizacionais?

A. Usar o Organizations com uma conta por cliente, SCPs nas OUs e faturamento consolidado
B. Configurar AWS Control Tower apenas na conta principal da empresa, mantendo todos os clientes como VPCs separadas dentro dessa conta.
C. Criar múltiplas contas AWS independentes e gerenciar o acesso de cada conta através de usuários federados no AWS IAM Identity Center, sem usar AWS Organizations.
D. Criar uma única conta AWS e utilizar tags para separar os recursos de cada cliente, aplicando políticas de segurança via IAM em nível de recurso.

01

CAPÍTULO 1

Aceleração da migração e da modernização de workloads

Aqui eu preciso dominar o programa de migração em escala: descobrir e avaliar o portfólio, escolher a estratégia por aplicação, organizar ondas, executar com as ferramentas certas e só depois modernizar. O padrão que domina a prova é migrar primeiro e modernizar depois: rehost rápido com o AWS MGN (ou VMware Cloud on AWS quando há muito VMware), bancos com DMS e CDC, e modernização gradual com strangler fig, App2Container e Refactor Spaces. Os enunciados trazem prazo agressivo para fechar o data center, centenas de servidores e pouca vontade de refatorar, e as alternativas erradas propõem reescrever tudo de uma vez, usar a ferramenta de outro tipo de carga (DMS para servidor, DataSync para servidor) ou processos manuais. Onde mais erro: trocar MGN por DMS, escolher DataSync quando o link é o gargalo e esquecer que aplicação sem uso é retire.

Neste capítulo
17 tópicos-chave · 12 pegadinhas · 6 macetes · 14 gatilhos · 4 questões comentadas

Tópicos-chave

Estratégias de migração (7 Rs)

Rehost, Replatform, Repurchase, Refactor, Retire, Retain e Relocate. Aplicação sem uso é Retire, desligando após confirmar que não é necessária; migrar ou reescrever algo sem uso gera custo sem valor. Em portfólios grandes, a estratégia varia por perfil de risco: rehost para críticas, replatform para internas, refactor para experimentais.

AWS Application Migration Service (MGN)

Principal serviço de rehost: replicação contínua de servidores físicos e virtuais para EC2 e cutover com mínima mudança e parada curta. Exportar imagens de disco à mão é lento, e o DMS migra só bancos, não servidores inteiros.

AWS Application Discovery Service

Coleta inventário, uso e dependências dos servidores on-premises (com agentes) para planejar ondas e decidir estratégias. Inspector procura vulnerabilidades e Macie descobre dados sensíveis, não dependências.

AWS Migration Hub e Strategy Recommendations

O Migration Hub é a fonte única de inventário, planejamento de ondas e acompanhamento do progresso, integrado ao MGN e ao DMS. O Strategy Recommendations analisa as aplicações e sugere a estratégia de cada workload com base nos dados coletados.

Migration Hub Orchestrator

Modelos de runbook reutilizáveis para padronizar e automatizar migrações e modernizações repetitivas (por exemplo, WebSphere ou JBoss com Oracle) por onda. É a peça da fábrica de migração.

Migration Hub Refactor Spaces e strangler fig

O strangler fig moderniza um monólito sem virada única: uma fachada (API Gateway, ALB) roteia aos poucos funcionalidades para microsserviços em ECS, Fargate ou Lambda. O Refactor Spaces gerencia esse roteamento incremental sem interromper o monólito.

AWS App2Container

Empacota aplicações Java e .NET existentes em containers para ECS ou EKS sem reescrever código, muitas vezes rodando no ECS com Fargate. É o passo de modernização rápida depois do rehost ou para tirar a gestão de servidores.

AWS DMS com CDC e conversão de esquema

Migração de banco com parada mínima: carga completa mais replicação contínua de mudanças (CDC) até o cutover. Entre motores diferentes (Oracle para Aurora PostgreSQL) usa-se a ferramenta de conversão de esquema e código (SCT) junto com o DMS.

Babelfish for Aurora PostgreSQL

Leva aplicações SQL Server para Aurora PostgreSQL com pouca mudança, eliminando licenças. Ficar no RDS for SQL Server mantém as licenças e DynamoDB exigiria reescrever a aplicação.

VMware Cloud on AWS com HCX

Migração em massa de VMs VMware sem conversão, mantendo o stack vSphere, dependências L2 e rollback, quando o prazo é agressivo e há pouco apetite para refatorar. A modernização para ECS, EKS e RDS é planejada em paralelo.

Snowball Edge

Muitos TB com link fraco e prazo curto: transferência física. DataSync, Transfer Acceleration ou upload pelo console continuam presos ao link local; o Snowball não mantém dados sincronizados para cutover com parada mínima.

AWS DataSync

Cópia agendada e recorrente de compartilhamentos NFS ou SMB para EFS ou S3, com verificação e monitoramento gerenciados. Substitui rsync manual; não migra servidores nem bancos.

Armazenamento e middleware compatíveis

NAS on-premises vai para FSx for NetApp ONTAP; NFS estendido pode usar Storage Gateway; Oracle com customizações vai para RDS Custom ou fica em EC2 na primeira fase; JMS ou ESB vai para Amazon MQ; Hadoop, Spark e Hive vão para EMR quase sem mudança.

AWS MAP e AWS CAF

O MAP (Migration Acceleration Program) estrutura o programa em avaliação, mobilização e migração, com playbooks e financiamento. O CAF responde como a organização precisa se transformar em pessoas, processos e governança para adotar a nuvem.

Landing zone antes da migração

Programas grandes começam pela fundação: Control Tower (com Landing Zone Accelerator), SCPs, contas e VPCs padronizadas, Service Catalog e StackSets para autoatendimento das equipes. Migrar sem essa base gera contas despadronizadas.

Planejamento de ondas

Comece pelas aplicações de menor complexidade para ganhar experiência e agrupe na mesma onda aplicações com forte dependência entre si. Ondas por ordem alfabética ou com as mais críticas primeiro sem testes são erradas.

Containers no Fargate

Para tirar a gestão de servidores sem reescrever, coloque a aplicação em containers no ECS com Fargate. EC2 com o mesmo sistema mantém a gestão de servidores e reescrever em Lambda é caro e arriscado.

Macetes para memorizar

Padrão-mestre

Migrar rápido, modernizar depois: MGN agora, strangler fig depois.

Ferramenta por carga

Servidor = MGN; Banco = DMS (+SCT); Arquivos = DataSync; Link fraco = Snowball; VMware inteiro = VMware Cloud on AWS.

Fluxo do programa

Descobrir, Avaliar, Planejar, Migrar, Modernizar: Discovery, Strategy Recommendations, Migration Hub, MGN e DMS, App2Container e Refactor Spaces.

MAP x CAF

MAP = como migrar (e quem paga); CAF = como a empresa muda.

Ondas

Simples primeiro, dependentes juntos.

Strangler fig

A figueira cresce em volta da árvore velha até substituí-la: fachada na frente, extrai uma rota por vez.

Tabela de memorização: não confunda

Quando é...	...e quando é
AWS MGN Servidores inteiros (sistema operacional e aplicação) físicos ou virtuais para EC2.	**AWS DMS** Bancos de dados, com carga completa e CDC, inclusive entre motores diferentes com SCT.
AWS MGN Rehost para EC2 nativo com replicação contínua e cutover.	**VMware Cloud on AWS** Muitas VMs VMware, prazo agressivo, manter vSphere e dependências L2 sem conversão.
Snowball Edge Volume grande com link fraco e prazo curto: o gargalo é a rede.	**AWS DataSync** Cópia online recorrente e agendada de NFS ou SMB para EFS ou S3 quando a rede comporta.
App2Container Empacotar a aplicação Java ou .NET como está em container, sem mudar código.	**Refactor Spaces / strangler fig** Extrair funcionalidades do monólito aos poucos para microsserviços.
AWS MAP Como estruturar e financiar o programa de migração (avaliação, mobilização, migração).	**AWS CAF** Como a organização se transforma em pessoas, processos e governança.
Application Discovery Service Coletar inventário, uso e dependências dos servidores.	**Migration Hub Strategy Recommendations** Recomendar a estratégia de migração e modernização de cada aplicação.

Números que caem

Item	Valor
Peso do domínio no SAP-C02	**20% da pontuação do conteúdo**
Estratégias de migração	**7: Rehost, Replatform, Repurchase, Refactor, Retire, Retain, Relocate**

Alertas de prova (pegadinhas)

⚠ **Alerta de prova:** Usar o DMS para migrar servidores inteiros.
Como não cair: O DMS migra bancos de dados; servidores com sistema operacional e aplicações são MGN.

⚠ **Alerta de prova:** Usar o MGN, o DMS ou scripts rsync manuais para copiar compartilhamentos NFS para o EFS de forma recorrente.
Como não cair: O MGN replica servidores inteiros, o DMS migra bancos e rsync não tem verificação nem monitoramento gerenciados; cópia recorrente de NFS para EFS ou S3 é DataSync com tarefas agendadas.

⚠ **Alerta de prova:** DataSync ou Transfer Acceleration pela conexão de 100 Mbps já saturada.
Como não cair: Nenhum dos dois supera o limite do link local; com prazo curto e muitos TB use Snowball Edge.

⚠ **Alerta de prova:** Rehost, replatform ou refactor de aplicação que ninguém usa.
Como não cair: Migrar ou reescrever algo sem uso gera custo sem valor; a estratégia é Retire.

⚠ **Alerta de prova:** Reescrever o sistema inteiro e virar tudo num fim de semana.
Como não cair: Virada única concentra o risco; use strangler fig e migração em duas fases.

⚠ **Alerta de prova:** Congelar o monólito ou só aumentar os servidores enquanto se estuda a nova arquitetura.
Como não cair: Congelar atrasa entregas e aumentar servidores não moderniza; extraia funcionalidades aos poucos.

⚠ **Alerta de prova:** Migrar as aplicações mais críticas primeiro, sem testes, ou separar dependentes em ondas distantes.
Como não cair: Comece pelo mais simples e agrupe dependentes na mesma onda.

⚠ **Alerta de prova:** Exportar o banco para CSV e importar no fim de semana, ou levar com Snowball toda semana.
Como não cair: Parada longa e sem sincronização; use DMS com CDC e conversão de esquema.

⚠ **Alerta de prova:** Continuar no RDS for SQL Server para eliminar licenças.
Como não cair: Mantém as licenças; Babelfish leva para Aurora PostgreSQL com pouca mudança.

⚠ **Alerta de prova:** Usar Inspector, Macie ou Artifact para descobrir dependências entre servidores.
Como não cair: Inspector acha vulnerabilidades, Macie dados sensíveis e Artifact fornece relatórios de conformidade; dependências são Application Discovery Service.

⚠ **Alerta de prova:** Modernizar tudo de uma vez (aplicação, dados e integração) com prazo agressivo de saída do data center.
Como não cair: Prazo curto pede rehost ou replatform primeiro e modernização depois da estabilização.

⚠ **Alerta de prova:** Migrar para EC2 com o mesmo sistema operacional quando o objetivo é eliminar a gestão de servidores.
Como não cair: Em EC2 a equipe continua gerenciando servidores; containers no Fargate resolvem sem reescrever.

Como cai na prova

📝 **Como cai:** Prazo agressivo para fechar o data center, centenas de servidores e desejo de modernizar no futuro.
Como resolver: Procure a alternativa em duas fases: rehost com MGN (ou VMware Cloud on AWS) e bancos com DMS para cumprir o prazo, depois modernização gradual com strangler fig, App2Container e bancos gerenciados. Descarte big bang e manter tudo on-premises.

📝 **Como cai:** Portfólio grande com pouca visibilidade do ambiente.
Como resolver: Comece por Application Discovery Service, Migration Hub como fonte única e Strategy Recommendations; depois ondas e execução com MGN e DMS, e Orchestrator para runbooks repetitivos.

📝 **Como cai:** Pergunta curta pedindo a ferramenta de transferência ou migração.
Como resolver: Identifique o tipo de carga (servidor, banco, arquivo) e a restrição (link, prazo, parada) e associe: MGN, DMS com CDC, DataSync, Snowball, Babelfish.

📝 **Como cai:** Monólito crítico 24x7 que precisa virar microsserviços sem parada.
Como resolver: Strangler fig com fachada no API Gateway ou Refactor Spaces, microsserviços no ECS com Fargate, DMS com CDC replicando dados para o banco novo e troca gradual das integrações.

📝 **Como cai:** Múltipla resposta sobre planejamento de ondas ou ferramentas de descoberta.
Como resolver: Marque as que reduzem risco e respeitam dependências (começar simples, agrupar dependentes; Discovery Service e Migration Hub). Descarte ordem arbitrária e ferramentas de segurança.

Gatilhos do enunciado

Se o enunciado trouxer...	Pense em
Lift-and-shift de muitos servidores com mínima mudança	**AWS Application Migration Service (MGN) com replicação contínua**

Centenas de VMs VMware, prazo agressivo, manter vSphere	**VMware Cloud on AWS com HCX**
Descobrir dependências e acompanhar a migração	**Application Discovery Service e Migration Hub**
Recomendar estratégia por aplicação	**Migration Hub Strategy Recommendations**
Padronizar runbooks repetitivos por onda	**Migration Hub Orchestrator**
Modernizar monólito sem virada única	**Strangler fig com Refactor Spaces ou API Gateway**
Containerizar Java ou .NET sem reescrever	**App2Container para ECS ou EKS com Fargate**
Oracle para Aurora com parada mínima	**Conversão de esquema (SCT) e DMS com CDC**
SQL Server para Aurora PostgreSQL com pouca mudança	**Babelfish**
Muitos TB, link fraco e prazo curto	**Snowball Edge**
Cópia recorrente de NFS para EFS ou S3	**AWS DataSync com tarefas agendadas**
Programa de migração com metodologia e financiamento	**AWS MAP**
Aplicação sem uso no portfólio	**Retire**
NAS ou JMS on-premises acoplados à aplicação	**FSx for NetApp ONTAP e Amazon MQ**

🎯 **Se você lembrar de só uma coisa deste capítulo...**

- MGN migra servidores; DMS migra bancos; DataSync copia arquivos.
- Quando o link é o gargalo, nenhuma ferramenta online resolve: Snowball.
- Aplicação sem uso é Retire.
- Prazo agressivo pede rehost primeiro; modernização vem depois e aos poucos.

Essenciais para a prova (checklist)

☐ Sei os 7 Rs e quando cada um é a resposta.

☐ Sei o papel de Application Discovery Service, Migration Hub, Strategy Recommendations e Orchestrator.

☐ Sei quando usar MGN e quando usar VMware Cloud on AWS.

☐ Sei migrar bancos com DMS e CDC, SCT e Babelfish.

☐ Sei escolher entre Snowball Edge e DataSync pela restrição de rede.

☐ Sei aplicar strangler fig com Refactor Spaces ou API Gateway.

☐ Sei usar App2Container para levar Java e .NET ao ECS ou EKS.

- ☐ Sei diferenciar MAP de CAF e por que a landing zone vem antes.
- ☐ Sei montar ondas começando pelo simples e agrupando dependentes.

Questões de revisão do capítulo

R1.1 · Aceleração da migração e da modernização de workloads

Uma empresa está migrando um portfólio grande de aplicações legadas para a AWS. A diretoria quer acelerar a migração mantendo riscos baixos, padronizando arquitetura e reduzindo o retrabalho entre times. O arquiteto de soluções recomenda adotar o AWS Cloud Adoption Framework (AWS CAF) e o AWS Migration Acceleration Program (MAP). Qual é a MELHOR forma de usar essas abordagens em conjunto para atingir o objetivo da empresa?

A. Utilizar o AWS CAF só para treinar a equipe técnica em serviços de infraestrutura e usar o AWS MAP só para migrações lift-and-shift.
B. Deixar de lado o AWS CAF e utilizar só o AWS MAP, pois o MAP já contempla as dimensões organizacionais e técnicas necessárias para a migração, incluindo governança e modelo operacional.
C. Usar o AWS CAF para alinhar pessoas, processos e governança e o AWS MAP para conduzir avaliação, mobilização e migração com playbooks
D. Utilizar o AWS CAF só para documentar a arquitetura atual on-premises e, em seguida, iniciar a migração diretamente com ferramentas de replicação de dados, evitando o uso do MAP para reduzir a complexidade do projeto.

R1.2 · Aceleração da migração e da modernização de workloads

Uma empresa de logística está migrando rapidamente seus sistemas de gestão de frotas e rastreamento de cargas para a AWS. O ambiente atual on-premises usa dezenas de VMs VMware com bancos de dados Oracle e aplicações Java legadas em WebLogic. O time de arquitetura definiu que a migração deve ser acelerada, com o mínimo de mudanças de código inicialmente, mas já preparando o ambiente para modernização futura (containerização e adoção de bancos gerenciados). Quais ações em conjunto atendem melhor a esses requisitos?

A. Recriar à mão as VMs no Amazon EC2 com AMIs personalizadas, migrar o Oracle para Amazon EC2 com instalação manual e postergar cada plano de containerização até que todo o ambiente esteja estável na AWS.
B. Migrar as aplicações Java para instâncias spot do Amazon EC2 para reduzir custo, manter o banco Oracle on-premises acessado por VPN como primeira fase.
C. Migrar diretamente as aplicações Java para AWS Lambda usando AWS Serverless Application Model (AWS SAM) e converter o banco Oracle on-premises em Amazon DynamoDB antes da migração para reduzir a dependência de infraestrutura.
D. Fazer lift-and-shift com o AWS MGN, levar o Oracle ao RDS for Oracle quando possível e containerizar aos poucos as aplicações Java com o App2Container para ECS ou EKS

R1.3 · Aceleração da migração e da modernização de workloads

Uma aplicação usa Microsoft SQL Server com muito código T-SQL. A empresa quer sair das licenças do SQL Server indo para o Amazon Aurora PostgreSQL, com o mínimo de mudanças na aplicação. Qual recurso ajuda mais?

A. Amazon RDS for SQL Server Multi-AZ
B. Amazon Redshift com consultas federadas
C. Babelfish for Aurora PostgreSQL
D. Amazon DynamoDB com tabelas globais

R1.4 · Aceleração da migração e da modernização de workloads

Uma indústria farmacêutica está migrando rapidamente seus workloads on-premises para a AWS e quer modernizar aplicações para arquitetura de microsserviços sem interromper os ciclos de desenvolvimento atuais. A equipe deseja automatizar a descoberta de dependências, criar pipelines de CI/CD padronizados e estabelecer padrões de infraestrutura como código reutilizáveis para acelerar a migração e reduzir riscos. Qual abordagem em conjunto atende melhor a esses requisitos?

A. Utilizar AWS Database Migration Service (AWS DMS) para migrar os bancos de dados e depois criar à mão scripts de implantação com AWS CodeDeploy para cada aplicação.
B. Usar o Strategy Recommendations e o Application Discovery Service para o portfólio e pipelines de IaC no CodePipeline
C. Usar AWS Snowball para transferir dados, configurar à mão ambientes em Amazon EC2.
D. Recriar as aplicações diretamente em AWS Lambda, migrando o código à mão e usando AWS CloudFormation só para provisionar funções Lambda e APIs.

Respostas comentadas

R1.1. C) Usar o AWS CAF para alinhar pessoas, processos e governança e o AWS MAP para conduzir avaliação, mobilização e migração com playbooks

Por quê: Esta alternativa é a correta porque reflete o uso complementar do AWS Cloud Adoption Framework (AWS CAF) e do AWS Migration Acceleration Program (MAP). O AWS CAF ajuda a alinhar pessoas, processos, governança e tecnologia, orientando a criação de uma foundation em múltiplas contas (segurança, rede, observabilidade, operações).

✗ A) Incorreta. Restringe o AWS CAF apenas a treinamento técnico e limita o MAP a lift-and-shift, o que não atende ao objetivo de modernizar workloads e padronizar arquitetura desde o início. A abordagem proposta aumenta retrabalho futuro.

✗ B) Incorreta. Superestima o MAP ao assumir que ele substitui o AWS CAF. O MAP é um programa de aceleração de migração, mas não cobre de forma tão abrangente as perspectivas organizacionais, de pessoas e de governança tratadas pelo CAF.

✗ D) Incorreta. Reduz o AWS CAF a mera documentação da arquitetura atual e ignora o MAP, perdendo o benefício de uma abordagem estruturada de migração e modernização.

> **Dica de prova:** Lembre que o AWS CAF responde à pergunta 'como a organização precisa se transformar para adotar nuvem?', enquanto o MAP responde 'como estruturar e acelerar a migração e modernização em fases?'.

R1.2. D) Fazer lift-and-shift com o AWS MGN, levar o Oracle ao RDS for Oracle quando possível e containerizar aos poucos as aplicações Java com o App2Container para ECS ou EKS

Por quê: Esta alternativa combina aceleração da migração com preparação para modernização. O AWS Application Migration Service (AWS MGN) possibilita lift-and-shift rápido das VMs VMware para EC2, minimizando mudanças de código e de arquitetura. Em seguida, o uso de serviços gerenciados como Amazon RDS for Oracle (quando suportado pelo licenciamento) reduz a carga operacional.

✗ A) Incorreta. Depende de recriação manual de VMs e do banco em EC2, o que é mais lento, mais propenso a erros e não utiliza serviços de migração nem bancos gerenciados. Não prepara adequadamente para modernização com containers.

✗ B) Incorreta. Mantém o banco Oracle on-premises via VPN, o que pode introduzir alta latência e complexidade operacional. Foca em redução de custo com EC2 Spot, mas não acelera a modernização nem utiliza serviços gerenciados ou ferramentas...

✗ C) Incorreta. Exige replatforming/rearquitetura agressiva (Lambda + DynamoDB) de todas as aplicações e do banco antes da migração, o que aumenta muito o esforço e risco, contrariando o requisito de mínimas mudanças de código na fase inicial.

> **Dica de prova:** Para provas de arquiteto profissional, associe: AWS MGN a lift-and-shift rápido de VMs; App2Container à containerização de aplicações existentes (Java/.NET); e bancos gerenciados (RDS/Aurora) como etapa natural de modernização após o rehost.

R1.3. C) Babelfish for Aurora PostgreSQL

Por quê: Babelfish for Aurora PostgreSQL acelera migrações de SQL Server para PostgreSQL, com menor esforço de alteração no código.

✗ A) Incorreta. Continuar no SQL Server não elimina as licenças.

✗ B) Incorreta. O Redshift é um data warehouse, não um banco transacional para a aplicação.

✗ D) Incorreta. O DynamoDB exigiria reescrever a aplicação.

> **Dica de prova:** SQL Server para Aurora PostgreSQL com pouca mudança: Babelfish.

R1.4. B) Usar o Strategy Recommendations e o Application Discovery Service para o portfólio e pipelines de IaC no CodePipeline

Por quê: Esta alternativa combina serviços voltados exatamente para acelerar migração e modernização com governança e automação. O AWS Migration Hub Strategy Recommendations e o AWS Application Discovery Service ajudam a entender o portfólio de aplicações e suas dependências, permitindo definir estratégias de migração (rehost, replatform, refactor etc.).

✗ A) Incorreta. Foca apenas em migração de banco de dados com o AWS DMS e ignora descoberta de dependências de aplicações, padronização de IaC e automação completa de pipelines.

✗ C) Incorreta. AWS Snowball ajuda na transferência de grandes volumes de dados, mas não resolve descoberta de dependências, nem padroniza pipelines e infraestrutura.

✗ D) Incorreta. Força um único padrão (Lambda) para todas as aplicações, o que é irrealista em cenários corporativos complexos.

> **Dica de prova:** Para questões de aceleração de migração e modernização, procure combinações que incluam: (1) descoberta e análise de portfólio (Migration Hub, Application Discovery, Strategy Recommendations), (2) padronização de infraestrutura e pipelines (CloudFormation, CDK, AWS Proton) e (3) automação de CI/CD (CodePipeline,...

02

CAPÍTULO 2

Design de novas soluções

Este é o domínio de maior peso e o que mais exige repertório de serviços: eu tenho de escolher a arquitetura certa para requisitos de RTO e RPO, latência global, ingestão em streaming, segurança, multi-tenant e residência de dados. Os cenários misturam várias exigências (multi-Região ativo-ativo, menor esforço operacional, auditoria por 7 anos, soberania de dados) e a resposta certa é a combinação de serviços gerenciados que atende a todas, não a que atende só uma. Também caem perguntas curtas de associação direta: RDS Proxy para Lambda esgotando conexões, DAX para microssegundos, Global Accelerator para UDP com IPs fixos, Step Functions com waitForTaskToken para aprovação humana. Onde eu mais erro: confundir Global Accelerator com CloudFront, Aurora Global Database com DynamoDB global tables, Kinesis Data Streams com Data Firehose e escolher a estratégia de DR mais cara ou mais lenta do que o RTO pede.

Neste capítulo

20 tópicos-chave · 14 pegadinhas · 7 macetes · 14 gatilhos · 4 questões comentadas

Tópicos-chave

Estratégias de DR

Backup e restauração tem RTO de horas; pilot light replica só os dados e ainda precisa iniciar e escalar a aplicação; warm standby mantém uma cópia reduzida e ativa que escala na recuperação, dando RTO de minutos sem pagar duas Regiões cheias; ativo-ativo atende ao menor RTO mas custa capacidade total nas duas Regiões. A prova pede a mais barata que ainda cumpre o RTO e o RPO.

Aurora Global Database

Relacional com RPO de segundos entre Regiões: cluster primário grava, clusters secundários servem leitura local e a secundária é promovida por automação ou runbook no desastre. Restaurar backups em outra Região não atende ao mesmo RTO, e réplicas de leitura não aceitam gravações.

DynamoDB global tables

Tabela multi-Região ativo-ativo com gravação local em cada Região e replicação gerenciada; é a resposta para gravar localmente em várias Regiões com baixa latência. Exige boa chave de partição e combina com failover pelo Route 53 ou Global Accelerator.

DAX e ElastiCache

DAX fica na frente da tabela DynamoDB e entrega leitura em microssegundos; aumentar capacidade ou usar leitura fortemente consistente não reduz latência. ElastiCache guarda sessões fora das instâncias e faz cache de consultas para bancos relacionais.

RDS Proxy

Agrupa e reutiliza conexões quando funções Lambda esgotam o banco RDS. Aumentar o timeout da função, a memória ou mover o banco para EC2 não resolve a explosão de conexões.

AWS Global Accelerator

IPs fixos, entrada pela borda mais próxima, health checks com failover rápido entre Regiões e traffic dials por endpoint group; suporta TCP e UDP. É a resposta para jogos UDP, IPs fixos para clientes com allowlist e ativo-ativo multi-Região com um único endereço.

Amazon CloudFront com OAC e WAF

CDN para HTTP e HTTPS: cache de conteúdo estático e dinâmico, várias origens (S3, ALB, API Gateway), grupo de origens para failover e URLs assinadas. Com S3, o bucket fica privado com controle de acesso de origem (OAC) e bucket policy que só aceita a distribuição, e uma web ACL do WAF protege a borda.

Route 53: políticas de roteamento

Latência para mandar o usuário à Região mais rápida, geolocalização para residência de dados ou conteúdo por país, failover com health checks para DR e ponderado para canary. TTL longo atrasa o failover, porque clientes guardam o endereço antigo.

Kinesis Data Streams

Ingestão de streaming com shards, retenção e replay para reprocessamento e vários consumidores. Enhanced fan-out dá a cada aplicação leitura dedicada; com GetRecords os consumidores compartilham o limite do fragmento e competem entre si.

Amazon Data Firehose

Entrega gerenciada de streaming com buffer por tamanho e tempo direto no S3 (inclusive em Parquet), Redshift ou OpenSearch, sem gerenciar shards. É o caminho de menor esforço para levar eventos ao data lake e consultar pelo Athena.

MSK e Managed Service for Apache Flink

MSK é Kafka gerenciado para ingestão com muitos consumidores; o Flink faz processamento com estado e janelas de tempo em baixa latência. O MSK Connect grava dados brutos no S3 para reprocessar e o Glue Schema Registry gerencia esquemas.

SNS com filas SQS (fanout) e SQS como amortecedor

Um evento para vários sistemas, cada um no seu ritmo: tópico SNS com uma fila SQS assinada por sistema; numa fila única compartilhada cada mensagem vai para um só consumidor. Para proteger um destino limitado em pico, envie a uma fila SQS e limite a concorrência reservada da função consumidora.

Step Functions com waitForTaskToken

Fluxos longos com etapa humana usam um estado de tarefa com token de retorno que aguarda a aprovação. Uma função Lambda tem tempo máximo de 15 minutos e não pode esperar dias; cron com polling não dá orquestração, novas tentativas por etapa nem rastreamento.

Cognito x IAM Identity Center

Login de clientes da aplicação, com provedores sociais, SAML ou OIDC por tenant e MFA, é Amazon Cognito com grupos de usuários. O IAM Identity Center é para a força de trabalho, e usuários do IAM não são feitos para clientes.

AWS PrivateLink

Expor um serviço a outras VPCs ou clientes sem internet e com faixas de IP sobrepostas: serviço de endpoint com um Network Load Balancer. Peering e Transit Gateway exigem IPs sem sobreposição e dão acesso de rede mais amplo que o necessário.

VPC Lattice x ECS Service Connect

Comunicação entre microsserviços dentro do ECS é ECS Service Connect. Entre contas e VPCs, com roteamento ponderado para canary e A/B, HTTP e gRPC, sem sidecar, é VPC Lattice.

Spot confiável

Diversifique tipos de instância com a alocação price-capacity-optimized e salve pontos de controle para retomar após interrupção. O aviso de interrupção não pode ser desativado; estado só em memória se perde. AWS Batch com Spot em vários tipos é o padrão para simulações e jobs em lote.

FSx for Lustre x FSx for Windows File Server

HPC em Linux com dados no S3 é FSx for Lustre vinculado ao bucket. Compartilhamento SMB com Active Directory e permissões NTFS é FSx for Windows File Server; EFS é NFS para Linux e EBS por nó não forma sistema de arquivos compartilhado.

AWS Backup à prova de invasor

Vault Lock em modo de conformidade para imutabilidade e cópia dos backups para um cofre em outra conta da organização. Backups no mesmo cofre da produção ficam ao alcance de quem comprometeu a conta.

Residência de dados

Quando a lei exige que dados fiquem no país ou na Região, desconfie de replicação global automática: use recursos regionais, chave KMS por Região, buckets por país e replicação seletiva só de dados permitidos ou pseudonimizados.

Macetes para memorizar

💡 DR em escada

Backup (horas) < Pilot light (liga o motor) < Warm standby (motor ligado, pequeno) < Ativo-ativo (dois motores cheios). Suba só até o degrau que o RTO exige.

💡 Banco global

Aurora Global: um escreve, todos leem. DynamoDB global: todos escrevem.

💡 Borda

CloudFront guarda (cache HTTP); Global Accelerator encaminha (IP fixo, TCP/UDP).

💡 Streaming

Streams processa e reprocessa; Firehose só entrega no destino.

💡 Mensageria

SNS grita para todos, SQS guarda na fila; fanout = SNS + uma fila por ouvinte.

💡 Microsserviços

Dentro do ECS: Service Connect. Entre contas e VPCs: Lattice.

💡 Arquivos

Lustre = Linux veloz com S3; Windows File Server = SMB + AD; EFS = NFS Linux.

Tabela de memorização: não confunda

Quando é...	...e quando é
AWS Global Accelerator IPs fixos, tráfego TCP ou UDP, failover rápido entre Regiões e um único endereço de entrada para aplicações dinâmicas.	**Amazon CloudFront** Conteúdo HTTP e HTTPS com cache na borda, origem S3 privada com OAC, URLs assinadas e WAF.
Aurora Global Database Relacional ACID com uma Região de escrita, leitura local nas secundárias e RPO de segundos.	**DynamoDB global tables** NoSQL com gravação local em várias Regiões (ativo-ativo) e baixa latência global.
Kinesis Data Streams Processamento em tempo real com vários consumidores, replay e retenção para reprocessar.	**Amazon Data Firehose** Entrega gerenciada com buffer ao S3, Redshift ou OpenSearch, sem gerenciar shards.

Warm standby RTO de minutos sem pagar duas Regiões cheias: a cópia reduzida já roda e só escala.	**Pilot light** Custo mínimo com dados replicados, aceitando RTO maior porque a aplicação ainda precisa subir.
Amazon Cognito Clientes finais da aplicação, login social, SAML ou OIDC por tenant, MFA.	**AWS IAM Identity Center** Funcionários acessando contas e aplicações corporativas.
AWS PrivateLink Expor um serviço específico a muitos clientes, sem internet e com IPs sobrepostos.	**VPC peering ou Transit Gateway** Conectar redes inteiras sem sobreposição de IP quando se quer roteamento amplo.
FSx for Lustre HPC em Linux, vazão extrema, vinculado ao S3.	**FSx for Windows File Server** Cargas Windows com SMB, Active Directory e NTFS.

Números que caem

Item	Valor
Peso do domínio no SAP-C02	**29% da pontuação do conteúdo**
Tempo máximo de execução de uma função Lambda	**15 minutos**
RTO típico de backup e restauração	**horas**

Alertas de prova (pegadinhas)

⚠ **Alerta de prova:** Usar CloudFront para um jogo com tráfego UDP.
Como não cair: CloudFront é HTTP e HTTPS; UDP com IPs fixos e failover rápido é Global Accelerator. ALB também não suporta UDP nem IP elástico.

⚠ **Alerta de prova:** Criar réplica de leitura e mandar gravações para ela, ou usar RDS com réplicas entre Regiões para gravar localmente.
Como não cair: Réplicas de leitura não aceitam gravações; gravação local em várias Regiões é DynamoDB global tables.

⚠ **Alerta de prova:** Restaurar backups do RDS em outra Região como solução de RPO de segundos.
Como não cair: Restauração leva de minutos a horas e o RPO depende do último backup copiado; use Aurora Global Database.

⚠ **Alerta de prova:** Uma função Lambda que fica rodando até o analista aprovar.
Como não cair: Lambda tem tempo máximo de 15 minutos; use Step Functions com token de retorno.

⚠ **Alerta de prova:** Aumentar timeout ou memória das funções Lambda para liberar conexões do banco.
Como não cair: Isso não reduz o número de conexões nem limita execuções simultâneas; use RDS Proxy, ou SQS com concorrência reservada para controlar o fluxo.

⚠ **Alerta de prova:** Uma única fila SQS compartilhada para quatro sistemas que precisam do mesmo evento.
Como não cair: Cada mensagem da fila é consumida por um só sistema; use SNS com uma fila por sistema.

⚠ **Alerta de prova:** Peering ou Transit Gateway para expor uma API a clientes com IPs sobrepostos.
Como não cair: Ambos exigem faixas sem sobreposição; PrivateLink com NLB resolve e expõe só o serviço.

⚠ **Alerta de prova:** Tornar o bucket público para o CloudFront ler os objetos.
Como não cair: Bucket público permite acesso direto à origem; use OAC com bucket policy que só aceita a distribuição.

⚠ **Alerta de prova:** Usar IAM Identity Center ou usuários do IAM para clientes finais.
Como não cair: Identity Center é para força de trabalho; clientes são Cognito.

⚠ **Alerta de prova:** Exportar a chave do KMS ou decifrar sempre pela Região primária no DR.
Como não cair: Chaves simétricas do KMS não são exportáveis e depender da primária quebra a recuperação; use chave multi-Região com réplica na secundária.

⚠ **Alerta de prova:** Escolher ativo-ativo quando o enunciado pede RTO de minutos com custo controlado.
Como não cair: Ativo-ativo atende ao RTO mas tem o custo que a empresa quer evitar; warm standby é o meio-termo. Backup e restauração tem RTO de horas.

⚠ **Alerta de prova:** Consumidores de Kinesis lendo com GetRecords ou gravar o evento em cinco fluxos para cinco aplicações.
Como não cair: GetRecords compartilha o limite do fragmento e multiplicar fluxos multiplica custo; use enhanced fan-out.

⚠ **Alerta de prova:** Spot com um único tipo de instância numa única zona e estado em memória.
Como não cair: Aumenta falta de capacidade e perda de trabalho; diversifique com price-capacity-optimized e salve checkpoints.

⚠ **Alerta de prova:** Replicação global automática em cenário com exigência de soberania de dados (LGPD, GDPR).
Como não cair: Mantenha dados sensíveis na Região de origem, com recursos regionais, e replique só o que é permitido ou pseudonimizado.

Como cai na prova

📝 **Como cai:** Aplicação global com requisitos de RTO e RPO numéricos, baixa latência e menor esforço operacional.
Como resolver: Anote RTO, RPO, tipo de dado (relacional ou NoSQL) e quem grava onde. Relacional com uma Região de escrita leva a Aurora Global Database; escrita em várias Regiões leva a DynamoDB global tables. Depois escolha a entrada (Route 53 com health checks ou Global Accelerator) e elimine backup e restauração quando o RPO é de segundos.

📝 **Como cai:** Pipeline de eventos ou IoT em tempo quase real com retenção longa e consultas ad hoc.
Como resolver: Ingestão (IoT Core, Kinesis Data Streams ou MSK), processamento (Lambda ou Flink), estado recente no DynamoDB e histórico no S3 via Firehose em Parquet, consultado pelo Athena. Prefira serverless e gerenciado a clusters próprios.

📝 **Como cai:** Pergunta curta com um sintoma técnico preciso (conexões esgotadas, latência de leitura, upload lento, UDP).
Como resolver: Associe sintoma a recurso: RDS Proxy, DAX, Transfer Acceleration, Global Accelerator. As erradas aumentam tamanho, timeout ou capacidade sem atacar a causa.

📝 **Como cai:** SaaS multi-tenant com isolamento, auditoria e IdPs diferentes por cliente.
Como resolver: Cognito com SAML e OIDC por tenant, tenant_id na chave do DynamoDB ou row-level security, chaves KMS por tenant quando exigido, auditoria no CloudTrail e S3 com Object Lock, novos tenants por IaC.

📝 **Como cai:** Cenário com residência de dados por país e baixa latência global.
Como resolver: Stacks regionais com banco e chave KMS da própria Região, contas por país padronizadas pelo Control Tower, Route 53 com geolocalização e failover e nenhuma replicação automática de dados sensíveis entre jurisdições (só replicação seletiva de dados pseudonimizados).

Gatilhos do enunciado

Se o enunciado trouxer...	Pense em
Relacional com RPO de segundos entre Regiões	**Aurora Global Database com promoção da secundária**

Gravar localmente em várias Regiões com baixa latência	**DynamoDB global tables**
Leitura do DynamoDB em microssegundos	**DAX**
Lambda esgotando conexões do RDS	**RDS Proxy**
UDP global, IPs fixos, failover rápido	**AWS Global Accelerator**
Etapa de aprovação humana que leva dias	**Step Functions com waitForTaskToken**
Um evento para vários sistemas, cada um no seu ritmo	**SNS com uma fila SQS por sistema**
Pico com destino limitado sem perder pedidos	**Fila SQS e concorrência reservada na função consumidora**
Expor serviço a outras VPCs sem internet e com IPs sobrepostos	**PrivateLink com Network Load Balancer**
RTO de minutos sem pagar duas Regiões cheias	**Warm standby**
Vários consumidores do mesmo stream sem disputar leitura	**Enhanced fan-out do Kinesis Data Streams**
Login de clientes com rede social e MFA	**Amazon Cognito**
API vendida por plano a parceiros	**Planos de uso com chaves de API no API Gateway e regras baseadas em taxa do WAF**
Upload global lento para um bucket	**S3 Transfer Acceleration**

🎯 **Se você lembrar de só uma coisa deste capítulo...**

- Réplicas de leitura não aceitam gravações.
- Lambda não espera mais de 15 minutos: espera humana é Step Functions.
- CloudFront não atende UDP; Global Accelerator sim.
- Escolha a estratégia de DR mais barata que ainda cumpre o RTO e o RPO pedidos.
- Soberania de dados derruba qualquer alternativa com replicação global automática.

Essenciais para a prova (checklist)

- ☐ Sei ordenar as estratégias de DR por RTO e custo e escolher pelo enunciado.
- ☐ Sei quando usar Aurora Global Database e quando usar DynamoDB global tables.
- ☐ Sei diferenciar Global Accelerator de CloudFront e de roteamento do Route 53.
- ☐ Sei montar um pipeline Kinesis, Flink ou Lambda, DynamoDB e S3 com Firehose e Athena.
- ☐ Sei aplicar SNS com SQS em fanout e SQS com concorrência reservada.

- ☐ Sei resolver conexões esgotadas com RDS Proxy e latência de leitura com DAX.
- ☐ Sei expor serviço com PrivateLink quando os IPs se sobrepõem.
- ☐ Sei escolher Cognito para clientes e Identity Center para funcionários.
- ☐ Sei proteger backups com Vault Lock e cópia entre contas.
- ☐ Sei escolher entre FSx for Lustre, FSx for Windows File Server e EFS.

Questões de revisão do capítulo

R2.1 · Design de novas soluções

Uma empresa precisa projetar uma nova solução de análise de dados em tempo quase real na AWS. Os dados são enviados continuamente a partir de dispositivos IoT espalhados globalmente. Os requisitos são: baixa latência de ingestão, capacidade de reprocessar dados históricos, possibilidade de executar consultas ad hoc em grandes volumes de dados e custo otimizado para armazenamento de longo prazo. Qual arquitetura atende melhor a esses requisitos?

A. Usar Amazon Kinesis Data Streams para ingestão, processar com AWS Lambda, armazenar os dados processados em Amazon S3 e usar Amazon Athena para consultas ad hoc.
B. Usar Amazon RDS para ingestão direta dos dados IoT e Amazon ElastiCache para consultas ad hoc, armazenando os dados só no banco relacional.
C. Usar Amazon SQS para ingestão dos dados IoT, processar com instâncias Amazon EC2 e armazenar só em Amazon EBS anexado às instâncias.
D. Usar Application Load Balancer para receber os dados IoT, gravar diretamente em Amazon Redshift e executar consultas ad hoc no próprio Redshift.

R2.2 · Design de novas soluções

Um fluxo de concessão de crédito tem várias etapas automáticas e uma etapa em que um analista precisa aprovar o pedido, o que pode levar até dois dias. O fluxo deve ter novas tentativas automáticas e rastreamento de cada execução. Qual solução é mais adequada?

A. Uma fila Amazon SQS por etapa, com consumidores em Amazon EC2 que consultam o banco em busca de aprovações
B. Um cron no Amazon EventBridge que verifica a cada minuto se há pedidos aprovados em uma tabela de controle
C. Uma função AWS Lambda que chama as etapas em sequência e fica em execução até o analista aprovar
D. AWS Step Functions com um estado de tarefa que usa token de retorno e aguarda a aprovação do analista

R2.3 · Design de novas soluções

Uma empresa de análise de dados deseja projetar uma solução serverless para processar grandes volumes de arquivos CSV recebidos diariamente em um bucket do Amazon S3. Os dados precisam ser limpos, transformados e então carregados em um data warehouse na AWS, com alta disponibilidade e mínima manutenção operacional. O arquiteto de soluções quer garantir que a arquitetura seja escalável, resiliente a falhas em uma região e que permita consultas analíticas rápidas. Qual desenho de arquitetura atende melhor a esses requisitos?

A. Usar eventos do S3 acionam uma Lambda de validação que envia metadados ao SQS; jobs do Glue transformam e carregam no Redshift RA3; e a replicação entre regiões protege os dados brutos
B. Configurar eventos do Amazon S3 para acionar funções AWS Lambda que executam a transformação dos arquivos e gravam os resultados diretamente em tabelas do Amazon Redshift.
C. Usar uma instância Amazon EC2 com scripts em cron para monitorar o bucket S3, baixar os arquivos, transformá-los localmente e carregá-los em um banco Amazon RDS for PostgreSQL. Configurar snapshots automáticos multi-AZ do RDS para garantir alta disponibilidade e performance nas consultas analíticas.
D. Criar um cluster gerenciado do Amazon EMR com auto scaling para processar os arquivos CSV diretamente do Amazon S3 e gravar os resultados em tabelas do Amazon Aurora MySQL Serverless. Replicar o cluster EMR para outra região por meio de scripts personalizados e usar Aurora Read Replicas para consultas analíticas.

R2.4 · Design de novas soluções

Uma empresa de e-commerce pretende lançar uma nova aplicação crítica em múltiplas regiões da AWS para melhorar latência global e garantir alta disponibilidade. A aplicação é stateless, executada em instâncias Amazon EC2 atrás de Application Load Balancers (ALBs). Os dados devem ser gravados de forma consistente em um único banco de dados principal, mas os usuários de outras regiões devem ter leitura com baixa latência. Qual arquitetura atende melhor a esses requisitos com o menor esforço operacional?

A. Criar um cluster Amazon DynamoDB global table com replicação entre regiões, usando a mesma tabela.
B. Configurar bancos de dados Amazon RDS independentes em cada região e implementar replicação lógica customizada via scripts ou ferramentas de ETL para sincronizar as alterações entre as regiões.
C. Criar um cluster Amazon RDS Multi-AZ em uma única região e configurar VPC peering entre as VPCs regionais para que as aplicações remotas leiam e gravem diretamente nesse banco de dados central.
D. Usar o Aurora Global Database, com escrita na região primária e leitura local nos clusters secundários

Respostas comentadas

R2.1. A) Usar Amazon Kinesis Data Streams para ingestão, processar com AWS Lambda, armazenar os dados processados em Amazon S3 e usar Amazon Athena para consultas ad hoc.

Por quê: Esta alternativa utiliza Amazon Kinesis Data Streams para ingestão de dados em baixa latência, permitindo lidar com fluxos contínuos de dispositivos IoT de forma escalável. O processamento com AWS Lambda permite transformar ou enriquecer os dados sem necessidade de gerenciar servidores. O armazenamento em Amazon S3 oferece custo otimizado e alta durabilidade para retenção de longo prazo, além de possibilitar reprocessamento futuro (releitura dos dados armazenados).

✗ B) Incorreta. Armazenar todos os dados em um banco relacional (RDS) não é ideal para grandes volumes de dados de streaming, devido a limitações de escalabilidade, custo e desempenho para consultas analíticas em larga escala.

✗ C) Incorreta. Amazon SQS não é otimizado para cenários de streaming em alta taxa e baixa latência como Kinesis. Armazenar somente em EBS torna o reprocessamento e a retenção de longo prazo mais complexos e caros, além de limitar a...

✗ D) Incorreta. Gravar diretamente em Amazon Redshift pode funcionar para análises analíticas, porém não é a melhor escolha para ingestão em alta taxa a partir de dispositivos globais, além de ser mais caro para armazenamento de longo prazo em...

> **Dica de prova:** Para cenários de ingestão contínua e de alto volume (como IoT, logs, eventos), pense em serviços de streaming (Kinesis ou MSK). Para armazenamento analítico e de longo prazo, lembre-se de Amazon S3 como data lake e de ferramentas serverless como Athena para consultas ad hoc.

R2.2. D) AWS Step Functions com um estado de tarefa que usa token de retorno e aguarda a aprovação do analista

Por quê: Step Functions orquestra fluxos de longa duração com estados, novas tentativas e histórico. O padrão de callback com token integra etapas humanas sem manter recursos em execução.

✗ A) Incorreta. Filas e consumidores próprios exigem código de orquestração e controle de estado manual.

✗ B) Incorreta. Polling por cron não oferece orquestração, novas tentativas por etapa nem rastreamento da execução.

✗ C) Incorreta. Uma função Lambda tem tempo máximo de 15 minutos e não pode aguardar dois dias.

> **Dica de prova:** Etapa humana em fluxo de longa duração: Step Functions com waitForTaskToken.

R2.3. A) Usar eventos do S3 acionam uma Lambda de validação que envia metadados ao SQS; jobs do Glue transformam e carregam no Redshift RA3; e a replicação entre regiões protege os dados brutos

Por quê: Esta alternativa utiliza componentes majoritariamente serverless (S3, Lambda, SQS, Glue) para orquestrar o fluxo de dados com baixa manutenção operacional. O AWS Glue executa ETL gerenciado e escalável lendo diretamente do S3, aplicando as transformações necessárias e carregando os dados em um data warehouse no Amazon Redshift RA3, que oferece alta performance para consultas analíticas e armazenamento gerenciado.

✗ B) Incorreta. Embora use componentes serverless (S3, Lambda, Redshift), depende de scripts manuais em EC2 para replicação entre regiões, aumentando o esforço operacional e pontos de falha.

✗ C) Incorreta. Baseia-se em EC2 e RDS, aumentando o gerenciamento de servidores e banco relacional para workload analítico em larga escala. RDS não é otimizado como data warehouse e a abordagem não é verdadeiramente serverless.

✗ D) Incorreta. O Amazon EMR pode ser adequado para big data, porém não é uma solução totalmente serverless e exige gerenciamento de cluster (mesmo com auto scaling).

> **Dica de prova:** Para workloads analíticos em larga escala com mínimo gerenciamento, priorize serviços gerenciados e/ou serverless (S3, Glue, Lambda, Redshift RA3). Use S3 como data lake, Glue para ETL/ELT e Redshift para consultas analíticas.

R2.4. D) Usar o Aurora Global Database, com escrita na região primária e leitura local nos clusters secundários

Por quê: Esta alternativa é a mais adequada porque o Amazon Aurora Global Database foi projetado especificamente para cenários multi-região com baixa latência de leitura e um único ponto de escrita. Ele permite ter um cluster primário (onde ocorrem as gravações) e réplicas em outras regiões para leituras locais, reduzindo latência para usuários globais com esforço operacional mínimo, já que a replicação é gerenciada pelo serviço.

✗ A) Incorreta. DynamoDB global tables permite escrita em múltiplas regiões, o que não atende ao requisito de manter um único banco de dados principal para gravações.

✗ B) Incorreta. Manter bancos independentes e replicação lógica customizada aumenta muito a complexidade operacional, risco de inconsistências e esforço de manutenção, contrariando o objetivo de menor esforço operacional.

✗ C) Incorreta. Centraliza todas as leituras e gravações em um único RDS em uma região, o que gera latência alta para usuários distantes e potencial gargalo de rede, além de não explorar recursos nativos de replicação multi-região.

> **Dica de prova:** Para requisitos de banco relacional multi-região com uma região primária de escrita e réplicas de leitura globais, pense primeiro em Aurora Global Database. Para NoSQL com escrita multi-master em várias regiões, lembre-se de DynamoDB global tables.

03

CAPÍTULO 3

Otimização contínua

Neste domínio a arquitetura já está em produção e eu preciso melhorá-la de forma incremental: mais confiabilidade, menos custo, mais observabilidade, deploys mais seguros e governança contínua, sem big bang. A prova descreve um ambiente com problemas (custos altos, subutilização, falha de deploy global, achados de segurança demorados, Athena lento) e quer a melhoria com o menor esforço e o menor risco. As respostas certas combinam quatro peças recorrentes: ferramentas de recomendação (Compute Optimizer, Cost Explorer, Well-Architected Tool), automação e IaC com pipelines, deploy seguro (blue/green, canary, rollback por alarme) e observabilidade centralizada. Onde mais erro: escolher reescrita total ou troca de orquestrador sem evidência, depender só de Spot em produção e aceitar processos manuais (planilhas, scripts em bastion, revisões semanais) como melhoria contínua.

Neste capítulo

18 tópicos-chave · 14 pegadinhas · 5 macetes · 14 gatilhos · 4 questões comentadas

Tópicos-chave

AWS Well-Architected Tool

Formaliza revisões periódicas dos workloads pelos pilares, com lenses padrão e customizadas, e pode ficar em uma conta central com acesso delegado e relatórios consolidados pelas APIs. A melhoria é aplicada de forma incremental, começando pela não produção. Documentar sem revisar e corrigir não é melhoria contínua.

AWS Compute Optimizer

Gera recomendações de rightsizing com aprendizado de máquina para EC2, Auto Scaling groups, EKS, RDS e outros, e pode ser ativado para a organização a partir da conta de gerenciamento. Junto com as recomendações de rightsizing do Cost Explorer, substitui scripts que calculam uso à mão.

Visibilidade de custo multi-conta

CUR em um bucket central analisado no Athena e no QuickSight, Cost Categories e tags para classificar, Budgets com alertas por equipe e Cost Anomaly Detection para gastos fora do padrão. Planilhas externas e relatórios manuais são distratores.

Modelos de compra

Base estável em Savings Plans ou instâncias reservadas e picos em Spot ou sob demanda com Auto Scaling. Compute Savings Plans, com compromisso por hora, cobrem uso que muda de tipo e de Região e também Fargate e Lambda; instâncias reservadas padrão ficam presas a atributos específicos.

Mixed instances policy

No Auto Scaling group, combina uma base sob demanda coberta por reservas ou Savings Plans com Spot para o restante, diversificando tipos de instância e zonas. Depender só de Spot em produção crítica viola o requisito de resiliência.

Políticas de escala

Scheduled scaling para picos conhecidos e predictive scaling, que usa o histórico, para criar capacidade antes do pico; target tracking e step para variações imprevisíveis. Escale pela métrica que reflete a carga real: RequestCountPerTarget ou latência do ALB, profundidade da fila SQS. Warm pools mantêm instâncias pré-aquecidas para reduzir o tempo de substituição.

Instance Scheduler on AWS

Para ambientes de dev e teste usados só em horário comercial, parar recursos fora do horário e redimensionar pelo Compute Optimizer são as economias rápidas. Instâncias maiores, dedicadas ou misturar teste com produção aumentam custo ou risco.

Deploy seguro com CodeDeploy e CodePipeline

Blue/green ou canary com target groups separados no ALB, smoke tests antes do cutover e rollback automático por alarmes do CloudWatch. Em multi-Região, o CodePipeline libera em ondas por Região, evitando que um bug derrube todas ao mesmo tempo. Aprovação manual antes de um deploy global não substitui ondas e rollback automático.

AWS AppConfig e feature flags

Mudanças de configuração tratadas como feature flags com liberação gradual, alarmes sobre P95 e erros e rollback automático quando um alarme dispara. Permite experimentação controlada em produção sem novo deploy de código.

RDS Blue/Green Deployments e Multi-AZ

Multi-AZ é a forma padrão de alta disponibilidade do RDS dentro de uma Região, com failover automático; snapshots não evitam a indisponibilidade. RDS Blue/Green Deployments fazem mudanças de schema e atualizações sem indisponibilidade perceptível.

Observabilidade centralizada

Conta de observabilidade com CloudWatch cross-account observability para métricas, logs e traces, X-Ray ou ADOT para tracing distribuído (propagando trace ID nas mensagens SQS), anomaly detection nos alarmes e SLOs. Instrumentação padrão aplicada por StackSets ou CDK em todas as contas.

CloudWatch Synthetics e DevOps Guru

Canários do Synthetics simulam fluxos como login e checkout e detectam falha antes do cliente; alarmes de CPU não mostram se o fluxo funciona. DevOps Guru e Application Insights detectam anomalias automaticamente.

Governança contínua com Config e Systems Manager

Conformance packs e regras do Config aplicados pela organização, agregador central, Security Hub com padrões como CIS e FSBP e remediação pelo Systems Manager Automation. Parâmetros que evoluem (lista de tipos de instância aprovados) ficam no Parameter Store ou AppConfig.

Resposta automática a achados

Achados do GuardDuty ou do Security Hub disparam uma regra do EventBridge que aciona a automação de isolamento ou remediação. Relatórios diários e reuniões semanais mantêm a demora. IAM Access Analyzer com a organização como zona de confiança aponta recursos acessíveis de fora.

Otimização de data lake no S3 com Athena

Converter para Parquet ou ORC comprimido, particionar pelas colunas mais filtradas (data, domínio, tenant), compactar arquivos pequenos com jobs recorrentes e manter o Glue Data Catalog atualizado. Normalmente não se troca de serviço: otimiza-se o formato e o layout.

Ciclo de vida do S3 x Intelligent-Tiering

Padrão de acesso previsível vai para regras de ciclo de vida por prefixo e tag (Standard-IA, Glacier); acesso imprevisível vai para Intelligent-Tiering. Deep Archive tem recuperação de horas e One Zone-IA guarda dados em uma única zona.

Otimizações de banco e cache

Relatórios atrapalhando gravações vão para réplicas do Aurora pelo endpoint de leitura; leitura repetida vai para ElastiCache ou CloudFront na borda; conexões demais vão para RDS Proxy. No Redshift, Concurrency Scaling e filas separadas no WLM resolvem disputa de painéis com consultas ad hoc.

Endpoint de gateway do S3 e outros ajustes rápidos

Tráfego para o S3 passando por NAT gera custo; um endpoint de gateway nas tabelas de rotas privadas resolve. Fila de mensagens mortas com limite de tentativas trata mensagem que falha sempre, e partição quente no DynamoDB se resolve com chave de alta cardinalidade ou sufixos.

Macetes para memorizar

Fórmula da resposta certa

Medir (CloudWatch, X-Ray) + Recomendar (Compute Optimizer, WA Tool) + Automatizar (IaC, pipeline, SSM) + Liberar com segurança (blue/green, canary, rollback).

Custo de compute

Base = compromisso (SP ou RI); pico = Spot; nunca tudo Spot em produção.

Escala

Previsível? Agende ou preveja. Imprevisível? Rastreie a métrica certa (ALB ou fila, não só CPU).

Athena

Athena lento não pede outro serviço, pede outro layout: formato colunar comprimido, partição pelas colunas filtradas, arquivos pequenos compactados e catálogo do Glue atualizado.

Resiliência

Primeiro Multi-AZ, depois multi-Região, sempre com game day.

Tabela de memorização: não confunda

Quando é...	...e quando é
Scheduled ou predictive scaling Picos previsíveis e recorrentes (horário, campanha, Black Friday): cria capacidade antes do pico.	**Target tracking ou step scaling** Variações imprevisíveis, reagindo à métrica em tempo real.
Compute Savings Plans Uso estável que muda de tipo, de Região ou inclui Fargate e Lambda.	**Instâncias Reservadas padrão** Uso fixo em um tipo e uma Região, sem necessidade de flexibilidade.
S3 Intelligent-Tiering Padrão de acesso desconhecido ou imprevisível.	**Regras de ciclo de vida** Padrão conhecido, como mover após 30 dias para Standard-IA e depois para Glacier.
Réplicas de leitura do Aurora ou RDS Descarregar relatórios e consultas pesadas da instância de gravação.	**ElastiCache** Respostas repetidas e idempotentes com menor latência e melhor custo-benefício que escalar o banco.
Blue/green ou canary com rollback automático Reduzir risco de deploy, testar com uma fração dos usuários e reverter rápido por alarme.	**Rolling ou deploy manual com aprovação** Não atende quando o enunciado pede rollback rápido e evitar propagação global de falhas.
CloudWatch Synthetics Saber se o fluxo de negócio (login, checkout) funciona antes do cliente reclamar.	**Alarmes de CPU** Medir saturação de recurso; não indica se a jornada do usuário está funcionando.

Números que caem

Item	Valor
Peso do domínio no SAP-C02 (no guia oficial: Melhoria contínua das soluções existentes)	**25% da pontuação do conteúdo**

Alertas de prova (pegadinhas)

⚠ **Alerta de prova:** Migrar de ECS para EKS ou reescrever em microsserviços como melhoria contínua.
Como não cair: Troca de orquestrador ou reescrita sem evidência de gargalo é big bang de alto risco; prefira ajustes incrementais no que existe.

⚠ **Alerta de prova:** Usar só instâncias Spot em produção crítica para cortar custo.
Como não cair: Spot pode ser interrompido; mantenha base em reservas, Savings Plans ou sob demanda e use Spot para o excedente.

⚠ **Alerta de prova:** Desligar Config, CloudTrail ou GuardDuty para reduzir ruído ou custo.
Como não cair: Desativar detecção reduz visibilidade e governança, o contrário de melhoria contínua.

⚠ **Alerta de prova:** Aumentar o armazenamento ou trocar a família da instância de um RDS Single-AZ que caiu.
Como não cair: Nenhum dos dois elimina o ponto único de falha; habilite Multi-AZ.

⚠ **Alerta de prova:** Aumentar a capacidade de gravação do DynamoDB ou criar GSI com a mesma chave para resolver partição quente.
Como não cair: O problema é concentração em uma partição; mude para chave de alta cardinalidade ou write sharding.

⚠ **Alerta de prova:** Aumentar o tempo de visibilidade da fila para vários dias quando uma mensagem falha sempre.
Como não cair: Isso só atrasa as novas tentativas; configure fila de mensagens mortas com limite de tentativas.

⚠ **Alerta de prova:** Aumentar o número de gateways NAT para reduzir custo de tráfego ao S3.
Como não cair: Mais NAT aumenta o custo; use endpoint de gateway do S3.

⚠ **Alerta de prova:** Mover tudo para Glacier Deep Archive ou One Zone-IA para economizar.
Como não cair: Deep Archive tem recuperação de horas e One Zone-IA fica em uma única zona; para acesso imprevisível use Intelligent-Tiering.

⚠ **Alerta de prova:** Instâncias burstable para todas as cargas, independentemente do perfil.
Como não cair: Sem análise de perfil pode gerar throttling e custos inesperados; baseie-se nas recomendações do Compute Optimizer.

⚠ **Alerta de prova:** Script em Python numa instância bastion para aplicar tags, encerrar instâncias ou mudar infraestrutura.
Como não cair: Scripts ad hoc criam ponto único de falha, pouca auditoria e nenhuma visão central; use Config com remediação pelo Systems Manager e IaC em pipeline.

⚠ **Alerta de prova:** Migrar o data lake para EFS ou copiar tudo para Redshift sem partição para acelerar o Athena.
Como não cair: EFS não serve para analytics em massa e Redshift duplica custo; otimize formato colunar, partições e compactação no próprio S3.

⚠ **Alerta de prova:** Detectar tipos de instância não aprovados só com notificação, com a lista num documento interno.
Como não cair: Notificar sem corrigir e lista desvinculada geram divergência; conformance pack com remediação e lista no Parameter Store.

⚠ **Alerta de prova:** Usar Macie, Inspector ou Trusted Advisor para achar recursos acessíveis de fora da organização.
Como não cair: Macie acha dados sensíveis, Inspector procura vulnerabilidades de software e as verificações de custo do Trusted Advisor não identificam acesso externo; recursos acessíveis de fora da organização são IAM Access Analyzer com a organização como zona de confiança.

⚠ **Alerta de prova:** Centralizar a aplicação em uma única Região ou AZ para simplificar o DR.
Como não cair: Isso só desloca o ponto único de falha; resiliência vem em camadas: primeiro Multi-AZ, depois multi-Região, com game days testando o failover.

Como cai na prova

📝 **Como cai:** Ambiente maduro em produção, várias dores ao mesmo tempo (custo, observabilidade, manutenção de componente próprio) e pedido de melhoria incremental.
Como resolver: Escolha a alternativa que ataca todas as dores com serviços gerenciados e mantém a base: misto de reservas e Spot, escala por métrica real, logs nativos no CloudWatch eliminando o componente autogerenciado. Descarte troca de orquestrador e soluções que só resolvem custo.

📝 **Como cai:** Incidente causado por deploy e exigência de rollback rápido e visibilidade quase em tempo real.
Como resolver: CodeDeploy blue/green ou canary, CodePipeline em ondas, AppConfig para funcionalidades, rollback por alarmes e observabilidade central com X-Ray. Aprovação manual e troca de load balancer não tratam o problema.

📝 **Como cai:** Governança multi-conta com desvios recorrentes (tags, tipos de instância, buckets sem criptografia).
Como resolver: Config com conformance packs pela organização, agregador central, Security Hub e remediação pelo Systems Manager Automation, com parâmetros no Parameter Store. Elimine bloqueio total por SCP que trava as equipes e scripts por conta.

📝 **Como cai:** Questão de múltipla resposta sobre reduzir custo de ambientes ociosos ou de rightsizing.
Como resolver: Marque as duas ações gerenciadas que atacam a causa (Instance Scheduler e Compute Optimizer, ou Compute Optimizer e Cost Explorer). Descarte as que aumentam tamanho, usam dedicadas ou misturam ambientes.

📝 **Como cai:** Data lake ou banco com consultas lentas e caras.
Como resolver: Otimize sem trocar de serviço: formato colunar, partições alinhadas aos filtros, compactação, ciclo de vida; no relacional, réplicas, cache e RDS Proxy antes de migrar tecnologia.

Gatilhos do enunciado

Se o enunciado trouxer...	Pense em
RDS em uma única zona caiu	**Converter para Multi-AZ**
Patches desiguais em frota grande e várias contas	**Systems Manager Patch Manager com linhas de base e janelas de manutenção**
Subutilização de CPU e memória, rightsizing entre contas	**Compute Optimizer na organização e Cost Explorer**
Uso estável que muda de tipo e de Região	**Compute Savings Plans**
Picos previsíveis e recorrentes	**Scheduled ou predictive scaling**

Bug distribuído para todas as Regiões ao mesmo tempo	**CodePipeline em ondas por Região, CodeDeploy blue/green e rollback por alarme**
Mudança de configuração arriscada em produção	**AWS AppConfig com feature flags, liberação gradual e rollback automático**
Detectar falha antes do cliente	**Canários do CloudWatch Synthetics**
Resposta lenta a achados do GuardDuty	**Regra do EventBridge acionando automação de isolamento**
Custo de NAT com tráfego para o S3	**Endpoint de gateway do S3**
Athena lento e caro	**Parquet ou ORC, partições, compactar arquivos pequenos, Glue Data Catalog**
Relatórios atrapalhando as gravações	**Réplicas do Aurora e endpoint de leitura**
Mensagem que falha sempre	**Fila de mensagens mortas**
Falhas de entrega do SNS para consumidores HTTP em pico	**Fila SQS assinada como buffer e escala pela profundidade da fila**

🎯 **Se você lembrar de só uma coisa deste capítulo...**

- Melhoria contínua é incremental: big bang e reescrita total quase sempre estão erradas.
- Spot sozinho em produção crítica é distrator; combine com base comprometida.
- Escale pela métrica que representa a carga (ALB, fila), não só por CPU.
- Desligar controles de segurança nunca é a melhoria.
- Ferramenta nativa de recomendação vence script próprio de cálculo de uso.

Essenciais para a prova (checklist)

☐ Sei usar o Well-Architected Tool com revisões periódicas e lenses em várias contas.

☐ Sei ativar o Compute Optimizer na organização e combinar com Cost Explorer, CUR e Cost Anomaly Detection.

☐ Sei escolher entre Compute Savings Plans, instâncias reservadas e Spot numa mixed instances policy.

☐ Sei quando usar scheduled, predictive, target tracking e warm pools.

☐ Sei montar blue/green e canary com CodeDeploy, CodePipeline em ondas e AppConfig com rollback por alarme.

☐ Sei centralizar observabilidade com cross-account, X-Ray ou ADOT, Synthetics e anomaly detection.

☐ Sei automatizar remediação com Config, Security Hub, EventBridge e Systems Manager Automation.

☐ Sei otimizar Athena com Parquet, partições e compactação e escolher entre ciclo de vida e Intelligent-Tiering.

☐ Sei resolver custo de NAT com endpoint de gateway e mensagem com falha recorrente com DLQ.

Questões de revisão do capítulo

R3.1 · Otimização contínua

Uma empresa de e-commerce opera cargas de trabalho em produção na AWS há dois anos. O time de arquitetura deseja implementar um processo de melhoria contínua para otimizar custos, desempenho e confiabilidade sem interromper o ambiente atual. Qual abordagem o arquiteto de soluções deve recomendar como primeiro passo mais adequado?

A. Substituir de pronto os componentes legados por serviços gerenciados serverless para reduzir custos.
B. Realizar uma migração completa para uma nova conta AWS criada do zero, reprovisionando os recursos com as melhores práticas mais recentes.
C. Ativar automaticamente os recursos de alta disponibilidade e disaster recovery nas aplicações, independentemente de requisitos de negócio ou custo.
D. Fazer revisões recorrentes do Well-Architected Framework nos pilares relevantes, com o Well-Architected Tool, e aplicar as melhorias de forma incremental

R3.2 · Otimização contínua

Uma empresa de logística utiliza um conjunto de instâncias Amazon EC2 em um Auto Scaling group atrás de um Application Load Balancer (ALB) para processar requisições de rastreamento de entregas. Recentemente, a empresa percebeu que, durante horários de pico, a latência média de resposta aumenta significativamente, embora as instâncias ainda apresentem uso de CPU abaixo de 50%. O arquiteto de soluções recebeu a tarefa de melhorar continuamente essa solução, reduzindo a latência com o mínimo de mudanças disruptivas. Qual é a MELHOR abordagem para atingir esse objetivo?

A. Ativar o AWS Global Accelerator para o ALB e aumentar o tamanho das instâncias EC2 do Auto Scaling group para reduzir a latência de rede e de processamento.
B. Configurar o Auto Scaling group para escalar só com base na métrica de CPU média das instâncias EC2, diminuindo o threshold.
C. Ativar HTTP/2 no Application Load Balancer, ajustar o keep-alive nas instâncias de back-end e revisar o limite de conexões simultâneas no servidor de aplicação
D. Substituir o Application Load Balancer por um Network Load Balancer, pois o NLB sempre oferece menor latência, independentemente do tipo de tráfego.

R3.3 · Otimização contínua

Uma empresa de streaming de vídeos já opera em múltiplas regiões da AWS e utiliza Auto Scaling Groups e Elastic Load Balancing para sua aplicação principal. A equipe de arquitetura deseja melhorar continuamente a solução, reduzindo custos em períodos de baixa demanda sem comprometer a disponibilidade, e garantindo que mudanças de configuração sejam auditáveis e fáceis de reverter. Qual abordagem é MAIS adequada para atender a esses requisitos?

A. Habilitar instâncias spot nos Auto Scaling Groups, usar escalabilidade manual baseada em planilhas de previsão de demanda e aplicar mudanças de configuração diretamente em produção via console para maior agilidade.
B. Reduzir o número de regiões ativas para uma única região, usar só instâncias Reserved com capacidade fixa e aplicar mudanças de configuração trimestralmente para minimizar o risco de falhas.
C. Substituir Auto Scaling Groups por instâncias individuais de grande porte, manter só instâncias On-Demand para evitar complexidade de cobrança e registrar mudanças de configuração em documentos compartilhados.
D. Usar Auto Scaling por métricas, combinar instâncias reservadas e sob demanda e gerenciar a infraestrutura com CloudFormation versionado

R3.4 · Otimização contínua

Uma aplicação de relatórios financeiros roda em instâncias Amazon EC2 em uma única AZ, atrás de um Application Load Balancer. A aplicação é crítica e tem janelas de manutenção muito restritas. A equipe de arquitetura identificou que, nos últimos meses, houve aumento de latência em horários de pico e incidentes de indisponibilidade durante atualizações de versão. O objetivo é melhorar continuamente a confiabilidade e a escalabilidade da solução, minimizando impacto em usuários durante mudanças. Qual abordagem é MAIS apropriada para atender a esses requisitos?

A. Migrar de pronto toda a aplicação para AWS Lambda e API Gateway para eliminar servidores, sem alterar o processo atual de implantação e sem mudanças na topologia de rede.
B. Usar um ASG multi-AZ, implantação blue/green pelo CodeDeploy com o ALB e scaling baseado na latência medida no ALB
C. Remover o Application Load Balancer para reduzir complexidade, manter só uma instância EC2 de grande porte.
D. Aumentar o tipo das instâncias EC2 para um tamanho maior, manter uma única AZ para simplificar o ambiente e realizar atualizações de versão só em janelas noturnas com baixa utilização.

Respostas comentadas

R3.1. D) Fazer revisões recorrentes do Well-Architected Framework nos pilares relevantes, com o Well-Architected Tool, e aplicar as melhorias de forma incremental

Por quê: Esta alternativa é a mais adequada porque o AWS Well-Architected Framework foi projetado exatamente para apoiar a melhoria contínua de workloads existentes. Usar o AWS Well-Architected Tool permite revisar sistematicamente os pilares (como custo, desempenho, confiabilidade, segurança e eficiência operacional), identificar riscos e oportunidades de otimização e, principalmente, implementar mudanças de forma incremental, sem grandes interrupções no ambiente de produção.

✗ A) Incorreta. Trocar todos os componentes legados por serverless de uma vez não é uma melhoria incremental. Além disso, nem todas as cargas de trabalho se beneficiam de serverless; isso pode aumentar a complexidade e o risco de regressões.

✗ B) Incorreta. Migrar tudo para uma nova conta do zero é arriscado, caro e não caracteriza um processo de melhoria contínua. É uma abordagem de big bang, com alto potencial de impacto na produção e sem garantia de ganhos imediatos.

✗ C) Incorreta. Ativar alta disponibilidade e DR em tudo, sem considerar requisitos de negócio ou custo, pode gerar desperdício financeiro significativo.

> **Dica de prova:** Para questões de melhoria contínua em ambientes já em produção, procure palavras-chave como 'Well-Architected Framework', 'reviews periódicas', 'incremental', 'iterativo' e 'minimizar impacto em produção'. Desconfie de alternativas que impliquem reescrever tudo de uma vez ou grandes migrações sem faseamento.

R3.2. C) Ativar HTTP/2 no Application Load Balancer, ajustar o keep-alive nas instâncias de back-end e revisar o limite de conexões simultâneas no servidor de aplicação

Por quê: Esta alternativa é a melhor abordagem porque foca em otimizações de camada de aplicação e transporte, que são compatíveis com a arquitetura existente e pouco disruptivas. Habilitar HTTP/2 no ALB permite multiplexação de requisições em uma única conexão, compressão de cabeçalhos e melhor utilização das conexões persistentes, reduzindo a latência percebida pelo cliente principalmente em cenários com muitas requisições simultâneas.

✗ A) Incorreta. Aumenta a complexidade e o custo ao introduzir o Global Accelerator e instâncias maiores sem evidências de que o gargalo seja rede de longa distância ou capacidade de CPU.

✗ B) Incorreta. Manter a escala baseada apenas em CPU não resolve gargalos ligados a conexões, filas internas ou limitação de threads.

✗ D) Incorreta. O Network Load Balancer não é necessariamente mais adequado para tráfego HTTP complexo. Ele trabalha na camada 4 e não oferece recursos nativos para HTTP/2, roteamento baseado em host/path ou recursos avançados de aplicação.

> **Dica de prova:** Quando a CPU está baixa mas a latência é alta em picos, desconfie de gargalos de conexões, filas internas de aplicação, limite de threads ou de otimização ineficiente de protocolo (HTTP/1.1 vs HTTP/2).

R3.3. D) Usar Auto Scaling por métricas, combinar instâncias reservadas e sob demanda e gerenciar a infraestrutura com CloudFormation versionado

Por quê: Esta alternativa é a mais adequada porque combina boas práticas de otimização de custos e governança. Políticas de Auto Scaling baseadas em métricas (por exemplo, utilização de CPU ou métricas customizadas no Amazon CloudWatch) permitem ajustar a capacidade automaticamente conforme a demanda, reduzindo custos em horários de baixa utilização. A combinação de instâncias Reserved e On-Demand equilibra economia de longo prazo com flexibilidade.

✗ A) Incorreta. O uso de instâncias spot pode reduzir custos, mas escalabilidade manual baseada em planilhas não é contínua nem confiável para responder à variação de carga.

✗ B) Incorreta. Reduzir para uma única região aumenta o risco de indisponibilidade e viola práticas recomendadas de alta disponibilidade. Capacidade fixa com instâncias Reserved impede ajuste dinâmico à demanda.

✗ C) Incorreta. Substituir Auto Scaling por instâncias individuais de grande porte reduz elasticidade e resiliência. Manter apenas instâncias On-Demand ignora oportunidades claras de economia com Reserved.

> **Dica de prova:** Para melhoria contínua em ambientes já em produção, pense em três pilares: (1) elasticidade automática com Auto Scaling e métricas bem definidas; (2) otimização de custos com o mix certo de modelos de aquisição (On-Demand, Reserved, eventualmente Spot);

R3.4. B) Usar um ASG multi-AZ, implantação blue/green pelo CodeDeploy com o ALB e scaling baseado na latência medida no ALB

Por quê: Esta alternativa combina boas práticas de alta disponibilidade, escalabilidade e implantação segura. O Auto Scaling Group multi-AZ aumenta a resiliência e reduz a indisponibilidade em caso de falha de zona. O uso de blue/green com AWS CodeDeploy e Application Load Balancer permite implantações com impacto mínimo, possibilitando rollback rápido em caso de problemas. Políticas de scaling baseadas em latência do ALB alinham a capacidade às necessidades reais de carga, reduzindo a latência em horários de pico.

✗ A) Incorreta. Migrar imediatamente para serverless sem planejamento detalhado e sem rever o processo de implantação não é uma melhoria contínua, e sim uma mudança drástica com risco elevado.

✗ C) Incorreta. Remover o ALB e usar uma única instância de grande porte reduz a resiliência e a escalabilidade. Snapshots diários ajudam na recuperação de desastres, mas não oferecem alta disponibilidade nem zero-downtime deployment.

✗ D) Incorreta. Apenas aumentar o tamanho das instâncias é uma escala vertical e não resolve problemas de alta disponibilidade nem de falha de AZ.

> **Dica de prova:** Para melhoria contínua de soluções existentes, priorize abordagens incrementais que aumentem disponibilidade, escalabilidade e capacidade de mudança segura (blue/green, canary). Procure sempre combinar multi-AZ, Auto Scaling e estratégias de deployment orientadas a rollback rápido.

04

CAPÍTULO 4

Soluções de design de complexidade organizacional

Aqui eu preciso desenhar a fundação multi-conta de uma empresa grande: como as OUs espelham unidades de negócio, ambientes, países e regulações, onde ficam as SCPs, quais contas são centrais (segurança, Log Archive, rede, serviços compartilhados) e como contas novas já nascem governadas pelo Control Tower. O domínio também cobra rede corporativa (Transit Gateway, compartilhamento de VPC pelo RAM, Direct Connect resiliente, DNS híbrido), identidade federada com o IAM Identity Center, auditoria centralizada (trilha de organização, agregador do Config) e rateio de custos. Os enunciados são longos, com várias BUs e requisitos de autonomia local junto com controle central, e a resposta certa quase sempre combina Organizations + OUs + SCPs + Control Tower + contas dedicadas. Onde eu mais erro: tratar SCP como algo que concede permissão, escolher peering em malha em vez de Transit Gateway e aceitar soluções por conta (scripts, trilhas locais, usuários do IAM) quando o enunciado pede padronização em escala.

Neste capítulo
18 tópicos-chave · 14 pegadinhas · 6 macetes · 14 gatilhos · 4 questões comentadas

Tópicos-chave

AWS Organizations e OUs

A hierarquia de OUs é o espelho da estrutura da empresa: por ambiente (Prod, Non-Prod, Sandbox), por unidade de negócio, por país ou por domínio regulatório. Contas são a fronteira de isolamento; OUs são a fronteira de governança onde se anexam políticas. A organização única dá faturamento consolidado na conta de gerenciamento e governança central; organizações separadas por BU perdem essas duas coisas.

Service Control Policies (SCPs)

SCPs nunca concedem permissões: definem o teto máximo do que as políticas do IAM podem permitir nas contas abaixo. O padrão cobrado é um baseline mínimo e estável na raiz, herdado por todos, e SCPs mais restritivas nas OUs filhas (produção mais fechada, sandbox mais permissiva). O IAM local continua detalhando permissões dentro desses limites.

SCP com aws:RequestedRegion

Para restringir Regiões em toda a organização, aplica-se uma SCP que nega ações quando aws:RequestedRegion não está na lista aprovada, com exceção para os serviços globais. É controle preventivo central, que vale também para contas futuras da OU.

AWS Control Tower e Account Factory

O Control Tower monta a landing zone com guardrails (controles) obrigatórios e recomendados, contas de log e auditoria e CloudTrail e Config centralizados. O Account Factory, integrado ao Service Catalog, cria contas padronizadas e pode ser delegado às unidades em autoatendimento, com baseline já aplicado. O Landing Zone Accelerator com CloudFormation complementa rede, segurança e componentes comuns.

Contas centrais dedicadas

A arquitetura de referência separa contas de segurança (auditoria), Log Archive, rede e serviços compartilhados, normalmente em uma OU própria. A conta de gerenciamento não deve ser usada para operações diárias e não pode ser movida para dentro de uma OU.

Administrador delegado

Serviços como Config, Security Hub, GuardDuty e CloudTrail podem ter uma conta de segurança registrada como administrador delegado. Assim a gestão central sai da conta de gerenciamento e a equipe de segurança tem visão consolidada sem usuários na conta de gerenciamento. Também se dá a cada unidade de negócio administração delegada para gerir contas só nas próprias OUs.

Trilha de organização do CloudTrail

Criada na conta de gerenciamento, cobre todas as contas, inclusive as futuras, e entrega os logs em um bucket da conta de Log Archive ou auditoria. Combine com SCPs que negam parar ou excluir trilhas, bucket policy restritiva (aws:PrincipalOrgID) e S3 Object Lock no modo Compliance quando o requisito é imutabilidade.

Agregador e conformance packs do AWS Config

O agregador do Config na conta de auditoria, abrangendo a organização, mostra a conformidade de todas as contas em um só lugar. Conformance packs e regras organizacionais padronizam os controles por OU; o Config registra configuração de recursos, não chamadas de API (isso é CloudTrail).

AWS IAM Identity Center

É o acesso corporativo a muitas contas: federado ao IdP (SAML, com sincronização por SCIM, por exemplo com o Entra ID), com permission sets atribuídos por grupos do IdP às contas. Elimina usuários do IAM por conta, chaves de longo prazo e usuários compartilhados, e segue o ciclo de vida do diretório.

Permissions boundary

Para delegar a criação de perfis com segurança, permita iam:CreateRole somente quando um limite de permissões for anexado e exija esse boundary nos perfis criados. Isso impede que desenvolvedores escalem privilégios criando perfis mais poderosos que eles mesmos.

Políticas de tag e Cost Categories

Políticas de tag do Organizations padronizam chave e valores permitidos entre todas as contas. Cost Categories com regras por conta e por tag de alocação fazem o rateio por unidade de negócio, inclusive dentro de contas compartilhadas; somado a Budgets por equipe, fecha o chargeback.

Transit Gateway hub-and-spoke

Um Transit Gateway por Região em uma conta de rede, compartilhado pelo RAM, com as VPCs das contas como spokes, substitui a malha de peerings. Tabelas de rotas por ambiente ou domínio segregam o tráfego, VPCs de inspeção e de egress centralizam firewalls e saída para a internet (com appliance mode para os firewalls), e o peering entre Transit Gateways liga Regiões diferentes.

Compartilhamento de VPC pelo RAM

A conta de rede é dona da VPC e compartilha sub-redes pelo AWS RAM com as contas de aplicação. A equipe de rede mantém controle central de roteamento, DNS e inspeção, e as equipes de aplicação ficam isoladas em suas contas sem administrar a rede.

Direct Connect resiliente e VPN de backup

Máxima resiliência exige conexões em dispositivos e locais diferentes, com um Direct Connect gateway para alcançar várias Regiões. O backup barato é uma VPN Site-to-Site com roteamento dinâmico BGP, que faz o failover automático; a VPN precisa estar ativa e pronta, não desligada.

Endpoints do Route 53 Resolver

DNS híbrido nos dois sentidos usa endpoint de entrada (o data center resolve nomes da AWS) e endpoint de saída com regras de encaminhamento (a VPC resolve nomes on-premises). As regras do Resolver são compartilhadas pelo RAM com as outras contas.

KMS centralizado na conta de segurança

Chaves (inclusive multi-Region) ficam na conta de segurança, com key policies que liberam uso às contas da organização por condição aws:PrincipalOrgID ou só às contas de uma BU, rotação e políticas mantidas centralmente e uso auditado no CloudTrail central. SCPs podem exigir essa chave em PutObject.

CloudFormation StackSets

A partir da conta de gerenciamento ou de uma conta de segurança, StackSets com permissões de serviço e implantação automática aplicam stacks padrão (CloudTrail, Config, regras, rede) a todas as contas de uma OU, inclusive as novas. Times locais ajustam parâmetros dentro de limites ou usam módulos do registry nos próprios templates.

Lake Formation entre contas

Em empresas com muitos domínios, o padrão é data mesh: contas por domínio ou país mantêm buckets e catálogos, e o Lake Formation compartilha tabelas entre contas com permissões finas, consultadas pelo Athena ou Spectrum sem copiar dados.

Macetes para memorizar

SCP

SCP é teto, IAM é chão: a SCP diz até onde pode ir, o IAM diz o que de fato pode.

Hierarquia de SCPs

Raiz mínima e estável, filhos mais fechados: produção aperta, sandbox afrouxa.

CloudTrail x Config

CloudTrail = quem fez o quê (API). Config = como está configurado (recurso).

Contas centrais

Kit da landing zone: Gerenciamento (organização e faturamento consolidado, nunca operação diária), Segurança, Log Archive, Rede, Compartilhados.

Rede multi-conta

Muitas VPCs = TGW no hub; sub-redes para os outros = RAM; IP sobreposto = PrivateLink.

Direct Connect

Resiliência de verdade = dispositivo diferente E local diferente; backup barato = VPN + BGP.

Tabela de memorização: não confunda

Quando é...	...e quando é
SCP Limite máximo para contas ou OUs inteiras (negar Regiões, impedir parar o CloudTrail, bloquear recurso sem tag), aplicado centralmente e herdado por contas futuras.	**Política do IAM / permissions boundary** Conceder e detalhar permissões dentro de uma conta, ou limitar perfis criados por desenvolvedores (boundary obrigatório no iam:CreateRole).
AWS Control Tower com Account Factory Criação de contas em escala, padronizada, com guardrails, logs centralizados e autoatendimento delegado às BUs.	**Organizations com runbooks ou scripts por conta** Nunca é a melhor resposta quando o enunciado fala em padronizar e automatizar: runbooks manuais são lentos e sujeitos a erro.
Transit Gateway Dezenas de VPCs e contas, roteamento transitivo, inspeção central, conexão com o data center e segmentação por tabelas de rotas.	**VPC peering** Poucas VPCs ligadas ponto a ponto; peering entre cada par escala mal e não é transitivo.
Trilha de organização do CloudTrail Registrar todas as chamadas de API de todas as contas, inclusive futuras, para auditoria.	**AWS Config com agregador** Avaliar e consolidar a conformidade da configuração dos recursos com regras e conformance packs.
Políticas de tag do Organizations Padronizar chave e valores permitidos das tags entre todas as contas.	**SCP com condição de tag** Impedir de forma preventiva a criação de recursos sem as tags obrigatórias.

Compartilhamento de sub-redes pelo RAM Rede central controlada pela equipe de rede com aplicações isoladas em contas separadas.	**VPC independente em cada conta** Equipe de aplicação administrando a própria rede, o que tira o controle central e é o que o enunciado quer evitar.
IAM Identity Center com IdP externo Acesso da força de trabalho a muitas contas com SSO, grupos e permission sets.	**Usuários do IAM por conta** Casos isolados; por conta gera credenciais espalhadas, remoção manual e falta de rastreabilidade.

Números que caem

Item	Valor
Peso do domínio no SAP-C02	**26% da pontuação do conteúdo**

Alertas de prova (pegadinhas)

⚠ **Alerta de prova:** Achar que uma SCP na raiz dá permissão às contas.
Como não cair: SCP só limita; sem política do IAM permitindo, nada é concedido. Pense em teto máximo, não em concessão.

⚠ **Alerta de prova:** Bloquear ec2:RunInstances por SCP em todas as contas e centralizar todo provisionamento na equipe de plataforma.
Como não cair: Isso cria gargalo e fere o requisito de impacto mínimo nas equipes; prefira guardrails e autoatendimento (Service Catalog, Account Factory) com detecção e remediação.

⚠ **Alerta de prova:** Criar uma trilha do CloudTrail em cada conta com script que configura as contas novas.
Como não cair: Trilhas por conta dependem de script e podem ser alteradas pelos administradores locais; a trilha de organização cobre contas futuras automaticamente.

⚠ **Alerta de prova:** Usar o AWS Config para registrar todas as chamadas de API.
Como não cair: O Config registra configurações de recursos; chamadas de API são CloudTrail.

⚠ **Alerta de prova:** Mover a conta de gerenciamento para a OU de segurança ou criar usuários nela para a equipe de segurança.
Como não cair: A conta de gerenciamento não pode ser movida para uma OU e não deve ser usada no dia a dia; registre a conta de segurança como administrador delegado.

⚠ **Alerta de prova:** Criar uma organização separada para cada unidade de negócio.
Como não cair: Organizações separadas perdem faturamento consolidado e governança central; use uma organização com OUs por BU e administração delegada.

⚠ **Alerta de prova:** Adicionar uma segunda conexão Direct Connect no mesmo dispositivo ou no mesmo local para ter redundância.
Como não cair: Mesmo dispositivo e mesmo local continuam sendo ponto único de falha; use dispositivos e locais diferentes ou VPN com BGP como backup.

⚠ **Alerta de prova:** Manter a VPN de backup desligada enquanto o Direct Connect está ativo, ou trocar rotas estáticas à mão na falha.
Como não cair: A VPN precisa estar pronta para assumir e o failover deve ser automático por BGP.

⚠ **Alerta de prova:** Resolver DNS híbrido com arquivos hosts, servidores DNS próprios em EC2 ou registros públicos para nomes internos.
Como não cair: Arquivos hosts não escalam, DNS próprio aumenta administração e registros públicos expõem a infraestrutura; use endpoints de entrada e saída do Route 53 Resolver.

⚠ **Alerta de prova:** Ligar VPCs de duas Regiões com peering entre cada par ou passando pelo data center via VPN.
Como não cair: Peering em malha escala mal e passar pelo data center adiciona latência e ponto único de falha; use peering entre os Transit Gateways.

⚠ **Alerta de prova:** Exigir tags com uma política do IAM em cada conta ou com uma planilha enviada às equipes.
Como não cair: Políticas por conta são difíceis de manter e não padronizam valores; planilha não aplica nada. Use políticas de tag do Organizations e SCP com condição de tag.

⚠ **Alerta de prova:** Dividir o custo total igualmente entre as BUs ou agrupar só por conta.
Como não cair: Divisão igual não reflete consumo e agrupar por conta não rateia contas compartilhadas; use Cost Categories por conta e por tag.

⚠ **Alerta de prova:** Dar AdministratorAccess aos desenvolvedores ou criar perfis com a conta raiz para agilizar.
Como não cair: Ambos permitem escalar privilégios; delegue com permissions boundary obrigatório.

⚠ **Alerta de prova:** Dar às equipes de aplicação acesso de administrador na conta de rede ou colocar todas as aplicações nela.
Como não cair: Quebra separação de responsabilidades e isolamento; compartilhe sub-redes pelo RAM e mantenha as aplicações em contas próprias.

Como cai na prova

📝 **Como cai:** Empresa com muitas BUs, países e regulações pede governança central, autonomia local e contas novas padronizadas, com quatro alternativas longas parecidas.
Como resolver: Procure a que tem os quatro elementos: Organizations com OUs alinhadas à estrutura, SCPs na raiz e nas OUs, Control Tower com Account Factory e contas dedicadas de segurança e logs. Descarte as que usam organizações separadas, scripts por conta ou centralizam tudo numa equipe.

📝 **Como cai:** Requisito de auditoria e imutabilidade de logs de todas as contas.
Como resolver: Trilha de organização na conta de gerenciamento, bucket na conta de auditoria com Object Lock Compliance e KMS, SCP que impede mexer no CloudTrail e agregador do Config. Elimine cópias semanais e trilhas por conta.

📝 **Como cai:** Rede com dezenas de VPCs em várias contas, inspeção central e conexão com o data center.
Como resolver: Hub-and-spoke: Transit Gateway em conta de rede, compartilhado pelo RAM, tabelas de rotas por ambiente, VPC de inspeção ou egress e Direct Connect gateway. Elimine peering em malha e VPN por VPC.

📝 **Como cai:** Pergunta curta pedindo o recurso exato (restringir Regiões, delegar perfis, padronizar tags, ratear custos).
Como resolver: Mapeie direto: aws:RequestedRegion em SCP, permissions boundary, políticas de tag, Cost Categories. As erradas costumam ser planilha, política por conta ou ação manual.

📝 **Como cai:** Fusão ou aquisição: integrar a conta da empresa comprada.
Como resolver: Convide a conta para a organização em uma OU própria com as SCPs corporativas, crie contas novas por ambiente e migre os workloads por fases com identidade e logs centralizados.

Gatilhos do enunciado

Se o enunciado trouxer...	Pense em
Várias unidades de negócio, autonomia local e controles mínimos centrais	**Uma organização, SCPs de baseline na raiz, OUs por BU com SCPs adicionais e administração delegada**

Contas novas devem nascer padronizadas e com guardrails	**AWS Control Tower com Account Factory**
Auditar todas as contas, inclusive as futuras	**Trilha de organização do CloudTrail entregando em bucket da conta de Log Archive**
Logs imutáveis por exigência regulatória	**S3 Object Lock no modo Compliance e SCP negando parar ou excluir trilhas**
Conformidade de toda a organização em um lugar	**Agregador do AWS Config na conta de auditoria**
Restringir Regiões em toda a organização	**SCP com aws:RequestedRegion, com exceção para serviços globais**
Acesso corporativo a muitas contas pelo diretório da empresa	**IAM Identity Center com IdP externo (SAML/SCIM) e permission sets**
Desenvolvedores podem criar perfis sem escalar privilégios	**Permissions boundary obrigatório no iam:CreateRole**
Malha de peerings crescendo e inspeção central	**Transit Gateway na conta de rede compartilhado pelo RAM, VPC de inspeção, appliance mode**
Rede central e aplicações em contas separadas	**Compartilhamento de sub-redes pelo AWS RAM**
Resolver nomes nos dois sentidos entre data center e VPCs	**Endpoints de entrada e saída do Route 53 Resolver com regras compartilhadas pelo RAM**
Backup barato do Direct Connect	**VPN Site-to-Site com BGP**
Ratear custos por unidade de negócio	**Cost Categories com regras por conta e tag, mais tag policies**
Segurança central fora da conta de gerenciamento	**Conta de segurança como administrador delegado do Security Hub, Config, GuardDuty**

🎯 Se você lembrar de só uma coisa deste capítulo...

- SCP nunca concede permissão; ela só limita o máximo, e a conta de gerenciamento não pode ser movida para uma OU.
- Trilha de organização cobre contas futuras; trilhas por conta e scripts são distratores.
- Padronizar contas novas em escala é Control Tower com Account Factory, não runbook.
- Mesmo local ou mesmo dispositivo do Direct Connect não é redundância.

Essenciais para a prova (checklist)

- ☐ Sei desenhar OUs por ambiente, BU, país ou regulação e explicar onde vai cada SCP.

- ☐ Sei escrever a ideia de uma SCP que restringe Regiões com aws:RequestedRegion.
- ☐ Sei o papel do Control Tower, do Account Factory e dos guardrails obrigatórios e recomendados.
- ☐ Sei montar a trilha de organização com bucket imutável na conta de auditoria.
- ☐ Sei diferenciar CloudTrail, Config com agregador e Security Hub.
- ☐ Sei quando usar Transit Gateway, peering de Transit Gateway e compartilhamento de VPC pelo RAM.
- ☐ Sei configurar Direct Connect resiliente e VPN com BGP como backup.
- ☐ Sei o que fazem os endpoints de entrada e saída do Route 53 Resolver.
- ☐ Sei delegar criação de perfis com permissions boundary e acesso com IAM Identity Center.
- ☐ Sei ratear custos com tag policies, Cost Categories e Budgets.

Questões de revisão do capítulo

R4.1 · Soluções de design de complexidade organizacional

Uma empresa tem um AWS Transit Gateway em sa-east-1 e outro em eu-west-1, cada um conectando várias VPCs. As VPCs das duas Regiões precisam se comunicar pela rede da AWS, sem criar conexões ponto a ponto entre elas. Qual solução atende a esse requisito?

A. Expor as aplicações na internet e restringir o acesso por IP
B. Criar um peering entre os dois Transit Gateways e ajustar as rotas
C. Criar peering de VPC entre cada par de VPCs das duas Regiões
D. Criar uma VPN entre cada VPC e o data center, que roteia o tráfego

R4.2 · Soluções de design de complexidade organizacional

Uma empresa usa AWS Organizations com dezenas de contas. Por exigência regulatória, nenhum recurso pode ser criado fora das Regiões sa-east-1 e us-east-1, mas serviços globais como IAM e Route 53 precisam continuar funcionando. Qual solução atende ao requisito com o menor esforço operacional?

A. Usar regras do AWS Config nas Regiões habilitadas para detectar recursos fora das duas Regiões e excluí-los por automação diária
B. Aplicar uma SCP que negue ações quando aws:RequestedRegion não for uma das duas Regiões, com exceção para os serviços globais
C. Desativar as Regiões não aprovadas nas configurações de cada conta e revisar manualmente a lista a cada novo serviço lançado
D. Criar em cada conta uma política do IAM que negue ações fora das duas Regiões e anexá-la aos usuários e perfis existentes na conta

R4.3 · Soluções de design de complexidade organizacional

Uma empresa global possui múltiplas contas AWS organizadas por unidades de negócio (finanças, marketing, engenharia) em uma organização do AWS Organizations. Cada unidade de negócio precisa autonomia para criar e gerenciar suas próprias VPCs, mas a empresa exige que: (1) todo tráfego entre VPCs passe por uma camada central de inspeção e segurança; (2) o roteamento seja padronizado e facilmente auditável; (3) a solução seja escalável para dezenas de VPCs em múltiplas contas. Qual abordagem de arquitetura atende melhor a esses requisitos?

A. Configurar peering VPC ponto a ponto entre as VPCs das contas, usando tabelas de rotas personalizadas para forçar a passagem por uma VPC de segurança central.
B. Habilitar AWS PrivateLink entre as VPCs das contas para que o tráfego privado seja roteado automaticamente por endpoints de serviço centralizados de segurança.
C. Criar uma conta de rede com um Transit Gateway compartilhado pelo RAM, anexar a ele as VPCs das demais contas e inspecionar o tráfego com appliances nessa conta central
D. Criar uma VPC compartilhada (Shared VPC) única em uma conta central e mover as workloads das outras contas para sub-redes dessa VPC.

R4.4 · Soluções de design de complexidade organizacional

Os funcionários de uma empresa se autenticam no Microsoft Entra ID. Eles precisam acessar dezenas de contas AWS com permissões por função, e os acessos devem ser removidos automaticamente quando alguém sai da empresa. Qual solução atende a esses requisitos?

A. Usuários do IAM criados em cada conta, com senhas sincronizadas a partir do diretório atual
B. Um usuário do IAM compartilhado por área, com a senha guardada no AWS Secrets Manager
C. Chaves de acesso de longo prazo distribuídas aos funcionários e trocadas a cada seis meses
D. AWS IAM Identity Center com o Entra ID como provedor de identidade e sincronização por SCIM

Respostas comentadas

R4.1. B) Criar um peering entre os dois Transit Gateways e ajustar as rotas

Por quê: Transit Gateways podem ser conectados entre Regiões por peering, formando uma rede global com roteamento centralizado.

✗ A) Incorreta. Expor na internet reduz a segurança.
✗ C) Incorreta. Peering de VPC entre cada par escala mal e é exatamente o que se quer evitar.
✗ D) Incorreta. Passar pelo data center adiciona latência e ponto único de falha.

> **Dica de prova:** Ligar redes de Transit Gateway em Regiões diferentes: peering de Transit Gateway.

R4.2. B) Aplicar uma SCP que negue ações quando aws:RequestedRegion não for uma das duas Regiões, com exceção para os serviços globais

Por quê: SCPs definem o máximo de permissões das contas de uma organização. A condição aws:RequestedRegion é o padrão para restringir Regiões, com exceções para serviços globais que operam a partir de us-east-1.

✗ A) Incorreta. O AWS Config detecta depois da criação. O requisito é impedir, e não corrigir depois.

✗ C) Incorreta. Controlar Regiões conta por conta, sem guardrail central, é manual e sujeito a falhas.

✗ D) Incorreta. Políticas do IAM por conta exigem manutenção contínua e não cobrem perfis criados depois, nem o usuário raiz.

> **Dica de prova:** Restringir Regiões em toda a organização: SCP com aws:RequestedRegion.

R4.3. C) Criar uma conta de rede com um Transit Gateway compartilhado pelo RAM, anexar a ele as VPCs das demais contas e inspecionar o tráfego com appliances nessa conta central

Por quê: Esta alternativa utiliza um padrão amplamente recomendado para ambientes multi-conta e multi-VPC: uma conta de rede central com um AWS Transit Gateway (TGW) compartilhado via AWS RAM. As VPCs de cada unidade de negócio são anexadas ao TGW, permitindo autonomia de criação e gestão por conta, enquanto o tráfego inter-VPC pode ser roteado obrigatoriamente por uma camada de segurança central (por exemplo, VPC de inspeção com firewalls, IDS/IPS ou appliances de terceiros).

✗ A) Incorreta. O peering VPC ponto a ponto escala mal em ambientes com muitas VPCs (crescimento quadrático de conexões). Além disso, peering não permite transitivity nativa, o que complica forçar todo tráfego a passar por uma VPC de segurança.

✗ B) Incorreta. O AWS PrivateLink é ideal para exposição de serviços específicos de forma privada, não para roteamento geral entre VPCs.

✗ D) Incorreta. Uma única VPC compartilhada reduz a autonomia das unidades de negócio, concentra blast radius em uma única VPC e não resolve bem o requisito de inspeção centralizada de tráfego entre workloads em larga escala.

> **Dica de prova:** Para cenários multi-conta e multi-VPC com requisitos de roteamento padronizado, inspeção centralizada e escalabilidade, pense em 'hub-and-spoke' com AWS Transit Gateway em uma conta central de rede. Lembre-se: peering VPC é simples, mas não escala bem; PrivateLink é para expor serviços, não para roteamento geral;

R4.4. D) AWS IAM Identity Center com o Entra ID como provedor de identidade e sincronização por SCIM

Por quê: Para acesso da força de trabalho em várias contas, o padrão é o IAM Identity Center federado ao provedor de identidade corporativo, com provisionamento automático por SCIM.

✗ A) Incorreta. Usuários do IAM por conta geram credenciais espalhadas e remoção manual.

✗ B) Incorreta. Usuários compartilhados impedem a rastreabilidade individual.

✗ C) Incorreta. Chaves de longo prazo aumentam o risco e não seguem o ciclo de vida do diretório.

> **Dica de prova:** Acesso corporativo a muitas contas: IAM Identity Center com IdP externo.

DIA DA PROVA

Estratégia de prova

O SAP-C02 é um exame de cenários longos de empresas complexas: muitas contas, várias Regiões, legado on-premises e requisitos simultâneos de custo, disponibilidade, segurança e prazo. Quase todas as alternativas são tecnicamente possíveis; a certa é a que atende a todas as restrições do enunciado com o menor esforço operacional, usando serviços gerenciados, automação e mudanças incrementais. Os distratores repetem padrões: processos manuais (planilhas, scripts em bastion, revisões semanais), big bang (reescrever tudo, trocar de orquestrador sem evidência), desligar controles e soluções que resolvem só um dos requisitos. Há questões de múltipla escolha e de múltipla resposta, e pelo peso Design de novas soluções (29%) e Complexidade organizacional (26%) somam mais da metade do conteúdo.

Como responder

✓ Controle o ritmo: são 75 questões em 180 minutos com enunciados longos; num cenário difícil, elimine o que viola as restrições, fique com a melhor restante e siga em vez de travar.

✓ Leia primeiro a pergunta final e destaque o qualificador decisivo: menor esforço operacional, menor custo, RTO e RPO, sem downtime, mínima refatoração, prazo agressivo.

✓ Liste as restrições do enunciado (por exemplo: multi-Região, auditoria, residência de dados, impacto mínimo nas equipes) e elimine toda alternativa que viole uma delas, mesmo que resolva as outras.

✓ Quando as alternativas são longas e parecidas, compare só o trecho que muda entre elas; geralmente a diferença é um serviço errado ou uma etapa manual.

✓ Elimine de cara alternativas com planilha, script próprio, cópia manual, desligar CloudTrail, Config ou GuardDuty, ou reescrever tudo de uma vez.

✓ Em múltipla resposta, selecione exatamente a quantidade pedida; você só ganha crédito marcando todas as corretas.

✓ Não deixe nada em branco: questões não respondidas contam como erradas e não há penalidade por chutar.

✓ Não se abale com uma questão estranha: 10 das 75 não valem nota e não são identificadas, e a pontuação é compensatória, então você precisa passar no exame geral, não em cada domínio.

Erros que derrubam candidatos

✗ Escolher a alternativa que resolve o problema técnico mas ignora o qualificador de menor esforço operacional ou menor custo.

✗ Confundir serviços vizinhos: MGN com DMS, Global Accelerator com CloudFront, Kinesis Data Streams com Firehose, Identity Center com Cognito, CloudTrail com Config.

✗ Cair em big bang: reescrita total, troca de orquestrador ou migração com virada única quando o enunciado pede incremental.

✗ Achar que SCP concede permissão ou aceitar soluções por conta quando o cenário é a organização inteira.

✗ Escolher DR mais caro (ativo-ativo) ou mais lento (backup e restauração) do que o RTO e o RPO pedem.

✗ Ignorar residência de dados e marcar alternativa com replicação global automática.

✗ Gastar tempo demais nos primeiros cenários longos e chegar às últimas questões sem tempo para ler os enunciados com calma.

Na semana da prova

- ☐ Confirme no agendamento que a prova é a SAP-C02 (75 questões, 180 minutos); o guia SAP-C03 tem outros domínios, então estude pelos quatro domínios do SAP-C02.
- ☐ Refaça as provas simuladas cronometradas e revise cada erro pelo porquê das alternativas erradas, anotando o par de serviços que confundiu.
- ☐ Revise os pares que mais derrubam: MGN x DMS, Snowball x DataSync, Aurora Global x DynamoDB global tables, Global Accelerator x CloudFront, warm standby x pilot light, SCP x IAM.
- ☐ Releia os gatilhos e macetes de cada capítulo no dia anterior, sem tentar aprender serviço novo.
- ☐ No dia, administre o tempo para chegar ao fim das 75 questões e confira se nenhuma ficou em branco, já que não há penalidade por chutar.

REVISÃO FINAL

Folha de cola

Leia na véspera e na manhã da prova: os gatilhos e as pegadinhas mais importantes de cada domínio.

1. Aceleração da migração e da modernização de workloads

‣ Lift-and-shift de muitos servidores com mínima mudança → **AWS Application Migration Service (MGN) com replicação contínua**

‣ Centenas de VMs VMware, prazo agressivo, manter vSphere → **VMware Cloud on AWS com HCX**

‣ Descobrir dependências e acompanhar a migração → **Application Discovery Service e Migration Hub**

‣ Recomendar estratégia por aplicação → **Migration Hub Strategy Recommendations**

‣ Padronizar runbooks repetitivos por onda → **Migration Hub Orchestrator**

⚠ O DMS migra bancos de dados; servidores com sistema operacional e aplicações são MGN.

⚠ O MGN replica servidores inteiros, o DMS migra bancos e rsync não tem verificação nem monitoramento gerenciados; cópia recorrente de NFS para EFS ou S3 é DataSync com tarefas agendadas.

⚠ Nenhum dos dois supera o limite do link local; com prazo curto e muitos TB use Snowball Edge.

2. Design de novas soluções

‣ Relacional com RPO de segundos entre Regiões → **Aurora Global Database com promoção da secundária**

‣ Gravar localmente em várias Regiões com baixa latência → **DynamoDB global tables**

‣ Leitura do DynamoDB em microssegundos → **DAX**

‣ Lambda esgotando conexões do RDS → **RDS Proxy**

‣ UDP global, IPs fixos, failover rápido → **AWS Global Accelerator**

⚠ CloudFront é HTTP e HTTPS; UDP com IPs fixos e failover rápido é Global Accelerator. ALB também não suporta UDP nem IP elástico.

⚠ Réplicas de leitura não aceitam gravações; gravação local em várias Regiões é DynamoDB global tables.

⚠ Restauração leva de minutos a horas e o RPO depende do último backup copiado; use Aurora Global Database.

3. Otimização contínua

‣ RDS em uma única zona caiu → **Converter para Multi-AZ**

‣ Patches desiguais em frota grande e várias contas → **Systems Manager Patch Manager com linhas de base e janelas de manutenção**

‣ Subutilização de CPU e memória, rightsizing entre contas → **Compute Optimizer na organização e Cost Explorer**

‣ Uso estável que muda de tipo e de Região → **Compute Savings Plans**

‣ Picos previsíveis e recorrentes → **Scheduled ou predictive scaling**

⚠ Troca de orquestrador ou reescrita sem evidência de gargalo é big bang de alto risco; prefira ajustes incrementais no que existe.

⚠ Spot pode ser interrompido; mantenha base em reservas, Savings Plans ou sob demanda e use Spot para o excedente.

⚠ Desativar detecção reduz visibilidade e governança, o contrário de melhoria contínua.

4. Soluções de design de complexidade organizacional

‣ Várias unidades de negócio, autonomia local e controles mínimos centrais → **Uma organização, SCPs de baseline na raiz, OUs por BU com SCPs adicionais e administração delegada**

‣ Contas novas devem nascer padronizadas e com guardrails → **AWS Control Tower com Account Factory**

‣ Auditar todas as contas, inclusive as futuras → **Trilha de organização do CloudTrail entregando em bucket da conta de Log Archive**

‣ Logs imutáveis por exigência regulatória → **S3 Object Lock no modo Compliance e SCP negando parar ou excluir trilhas**

‣ Conformidade de toda a organização em um lugar → **Agregador do AWS Config na conta de auditoria**

⚠ SCP só limita; sem política do IAM permitindo, nada é concedido. Pense em teto máximo, não em concessão.

⚠ Isso cria gargalo e fere o requisito de impacto mínimo nas equipes; prefira guardrails e autoatendimento (Service Catalog, Account Factory) com detecção e remediação.

⚠ Trilhas por conta dependem de script e podem ser alteradas pelos administradores locais; a trilha de organização cobre contas futuras automaticamente.

GABARITO

Respostas do diagnóstico

D1. D) Retire, desligando-as após confirmar que não são necessárias → revise o capítulo 1

Por quê: As estratégias de migração, conhecidas como 7 Rs, incluem retire para o que não é mais necessário, reduzindo custo e escopo da migração.

✗ A) Incorreta. Levar para contêineres também gera custo sem benefício.

✗ B) Incorreta. Migrar algo sem uso gera custo sem valor.

✗ C) Incorreta. Reescrever uma aplicação sem uso desperdiça esforço.

> **Dica de prova:** Aplicação sem uso: retire.

D2. C) Fazer lift-and-shift do monólito com o AWS MGN e do Oracle com o DMS e colocar uma camada de APIs na frente do monólito para extrair microsserviços aos poucos → revise o capítulo 1

Por quê: Esta alternativa equilibra velocidade de migração e caminho de modernização. O uso do AWS Application Migration Service (MGN) permite um lift-and-shift rápido do monólito para Amazon EC2, reduzindo mudanças de código e risco, alinhado ao requisito de 'mínima refatoração' em poucos meses. Em paralelo, o AWS Database Migration Service (DMS) possibilita migrar o banco Oracle para a AWS com downtime reduzido.

✗ A) Incorreta. O replatform imediato para ECS e RDS exige mais trabalho de adaptação de infraestrutura, empacotamento do monólito em contêiner e ajustes de dependências.

✗ B) Incorreta. Tenta fazer modernização completa (refatorar para microsserviços e migrar Oracle para Aurora PostgreSQL) antes de migrar para a nuvem, o que não atende ao prazo de poucos meses.

✗ D) Incorreta. Manter o banco Oracle on-premises em longo prazo aumenta a latência e a dependência do data center, contrariando o objetivo de migração do core para a AWS.

> **Dica de prova:** Em exames de migração e modernização, procure soluções em duas fases: (1) mover rápido com o mínimo de mudanças (lift-and-shift com MGN/DMS) e (2) criar mecanismos que facilitem a refatoração incremental, como camadas de API, estrangulamento de monólito (strangler pattern) e uso de serviços gerenciados em novos...

D3. D) Usar o AWS MGN para o lift-and-shift e, depois de estabilizar, modernizar aos poucos com RDS, ECS ou Lambda → revise o capítulo 1

Por quê: Esta alternativa é a mais adequada porque o uso do AWS Application Migration Service (AWS MGN) permite uma migração rápida no modelo lift-and-shift, com mínima alteração no código dos aplicativos. Após a migração e estabilização na AWS, a empresa pode modernizar gradualmente os workloads, migrando partes para serviços gerenciados como Amazon RDS, Amazon ECS, AWS Lambda ou outros, seguindo uma estratégia de modernização incremental (por exemplo, strangler pattern).

✗ A) Incorreta. Exigir a reescrita imediata de todos os aplicativos em microsserviços aumenta muito o risco, o esforço e o tempo de migração. Isso contraria o requisito de 'mínima refatoração inicial' e atrasa a ida para a nuvem.

✗ B) Incorreta. Migrar apenas os bancos de dados e manter os aplicativos on-premises acessando remotamente introduz alta latência, complexidade operacional e não acelera a migração dos aplicativos em si.

✗ C) Incorreta. AWS Snowball é focado em migração de dados em massa, não em migração de servidores/aplicações. Manter aplicativos legados em instâncias spot sem planejamento de modernização não garante estabilidade nem uma estratégia clara de...

> **Dica de prova:** Para acelerar migrações com baixa refatoração, lembre de associar: AWS MGN (lift-and-shift de servidores) e Database Migration Service (migração de bancos). Depois de estabilizar na nuvem, pense em modernização incremental com serviços gerenciados e padrões como strangler pattern, contêineres e serverless.

D4. A) Usar chaves multi-Região do AWS KMS, com réplica da chave na Região secundária → revise o capítulo 2

Por quê: Chaves multi-Região do AWS KMS permitem criptografar em uma Região e decifrar em outra, sem novas chamadas entre Regiões, o que apoia cenários de recuperação de desastres.

✗ B) Incorreta. Guardar a chave junto aos dados anula a proteção.

✗ C) Incorreta. Depender da Região primária para decifrar contraria o requisito de recuperação.

✗ D) Incorreta. Chaves simétricas geradas no KMS não são exportáveis, e a cópia manual seria insegura.

> **Dica de prova:** Decifrar em outra Região sem depender da primária: chave multi-Região do KMS.

D5. D) Usar o Aurora Global Database com escrita na região primária e leitura na secundária, roteamento por latência no Route 53 para ALBs nas duas regiões e um runbook automatizado de promoção da secundária → revise o capítulo 2

Por quê: Esta alternativa utiliza Amazon Aurora Global Database, que foi projetado especificamente para cenários multi-Região com baixa latência de replicação (tipicamente < 1 segundo) e RPO de poucos segundos, atendendo confortavelmente o RPO de 5 minutos. O RTO de 15 minutos é viável, pois a promoção da Região secundária a primária é um processo suportado nativamente pelo Aurora Global Database e pode ser automatizado via AWS Lambda + Step Functions ou outros mecanismos de orquestração.

✗ A) Incorreta. Depende de uma única Região com Multi-AZ, o que não cobre a falha total de Região. Os backups automáticos de Aurora são adequados para recuperação de desastres, porém com RPO e RTO típicos bem maiores que 5/15 minutos, além de...

✗ B) Incorreta. Mistura replicação artesanal (scripts customizados entre Aurora e MySQL em EC2) com diferentes tipos de load balancer e failover manual de DNS, o que eleva o risco operacional e o tempo de recuperação.

✗ C) Incorreta. Embora forneça multi-Região com RTO potencialmente aceitável, a replicação baseada em binlog configurada manualmente aumenta muito o esforço operacional (monitoramento, tuning e correção de falhas de replicação).

> **Dica de prova:** Para requisitos de DR multi-Região com RPO e RTO agressivos em bancos relacionais, priorize serviços e recursos nativos da AWS, como Aurora Global Database ou cross-Region read replicas, em vez de soluções artesanais com replicação manual.

D6. D) Amazon Aurora PostgreSQL com Aurora Global Database e uma Região secundária pronta para promoção → revise o capítulo 2

Por quê: Aurora Global Database é indicado para recuperação de desastres entre Regiões com RPO em segundos e RTO em torno de um minuto, com replicação gerenciada no nível de armazenamento.

✗ A) Incorreta. O Multi-AZ protege contra falhas de zona, não de Região, e snapshots por hora dão RPO de até uma hora.

✗ B) Incorreta. Restaurar backups leva de minutos a horas e o RPO depende do último backup copiado.

✗ C) Incorreta. Replicação em EC2 exige muita administração e não garante o RTO pedido.

> **Dica de prova:** Relacional com RPO de segundos entre Regiões: Aurora Global Database.

D7. B) Ativar o Config e os logs no CloudWatch, criar alarmes para métricas críticas e testar os ajustes em um ambiente idêntico antes da produção → revise o capítulo 3

Por quê: Esta alternativa combina monitoramento contínuo (AWS Config, CloudWatch Logs e alarmes CloudWatch) com um mecanismo seguro de evolução de arquitetura (CloudFormation StackSets para ambientes de teste e produção). Isso permite detectar falhas e desvios de configuração rapidamente, além de testar melhorias de forma controlada em um ambiente espelhado antes de aplicá-las em produção, alinhando-se às práticas de melhoria contínua e mudança segura.

✗ A) Incorreta. Depende de inspeção manual pós-falha, não oferece monitoramento proativo nem um processo estruturado para testar e promover melhorias. A análise é reativa, não contínua.

✗ C) Incorreta. Elimina automação e reduz a frequência de revisões. Um processo manual e trimestral é lento e sujeito a erros humanos, contrariando o princípio de observabilidade contínua e evolução iterativa.

✗ D) Incorreta. Implica uma grande mudança arquitetural de alto risco sem abordar claramente monitoramento, governança e processo de experimentação controlada.

> **Dica de prova:** Para questões de melhoria contínua em soluções existentes, procure combinações de: (1) monitoramento e observabilidade (CloudWatch, X-Ray, Config), (2) automação de mudanças (CloudFormation, CDK, CodePipeline), e (3) ambientes separados para teste/validação antes da produção.

D8. B) Regra do EventBridge que aciona a automação de isolamento → revise o capítulo 3

Por quê: Automatizar a resposta a achados de segurança com EventBridge reduz o tempo de contenção e melhora a postura de segurança.

✗ A) Incorreta. Desativar a detecção aumenta o risco.

✗ C) Incorreta. Um relatório diário mantém a demora na resposta.

✗ D) Incorreta. Revisões semanais são lentas demais para incidentes ativos.

> **Dica de prova:** Resposta lenta a achados: EventBridge com remediação automática.

D9. C) Criar um endpoint de gateway do S3 nas tabelas de rotas privadas → revise o capítulo 3

Por quê: Endpoints de gateway para Amazon S3 e DynamoDB são uma das formas mais simples de reduzir custos de NAT em tráfego para esses serviços.

✗ A) Incorreta. Expor instâncias em sub-redes públicas aumenta o risco de segurança.

✗ B) Incorreta. Trocar de armazenamento é uma mudança grande e sem relação com o custo de NAT.

✗ D) Incorreta. Mais gateways NAT aumentam o custo total.

> **Dica de prova:** Custo de NAT com tráfego para o S3: endpoint de gateway.

D10. D) Criar um trail organizacional com os logs entregues a um bucket da conta de segurança, com bucket policy restritiva, e SCPs que impedem as contas-membro de alterar o CloudTrail → revise o capítulo 4

Por quê: Esta alternativa é a mais adequada porque usa recursos nativos de governança organizacional da AWS. O AWS CloudTrail organizacional, criado a partir da conta de gerenciamento (ou delegada), garante que todas as contas-membro tenham trilhas ativas, e que não possam ser desativadas individualmente. Centralizar os logs em um bucket S3 de uma conta dedicada de segurança, com políticas de bucket que negam exclusão/alteração de logs, reforça a integridade dos registros.

✗ A) Incorreta. Consolidar faturamento não resolve o problema de governança e segurança. Um único CloudTrail na conta de gerenciamento, sem uso de CloudTrail organizacional e SCPs, não garante que todas as contas estejam cobertas nem impede que...

✗ B) Incorreta. AWS Config complementa, mas não substitui o CloudTrail. Config registra mudanças de configuração de recursos, enquanto CloudTrail registra chamadas de API e atividade de conta.

✗ C) Incorreta. Exige gerenciamento separado de CloudTrail em cada conta, aumentando muito o esforço operacional. IAM policies em cada conta não são suficientes para impedir alterações por administradores com privilégios elevados;

> **Dica de prova:** Para requisitos organizacionais de auditoria e conformidade, pense em três camadas: (1) AWS Organizations + SCPs para controle centralizado, (2) serviços organizacionais como CloudTrail organizacional e AWS Config Aggregator, e (3) conta dedicada de segurança para centralizar logs e monitoramento.

D11. D) Usar políticas de controle de serviço (SCPs) em nível de organização para negar a criação de recursos que não contenham as tags obrigatórias, combinando condições de tags nas políticas. → revise o capítulo 4

Por quê: Esta alternativa é a mais adequada porque utiliza AWS Organizations com Service Control Policies (SCPs) para impor, de forma centralizada, a presença de tags obrigatórias na criação de recursos. As SCPs permitem definir condições baseadas em tags (por exemplo, 'aws:RequestTag/Owner', 'aws:RequestTag/CostCenter') e negar ações de criação de recursos que não incluam essas tags. Essa abordagem: - Garante conformidade preventiva (bloqueia a criação sem tags, em vez de apenas detectar depois).

✗ A) Incorreta. Baseia-se apenas em processo e documentação, sem qualquer controle técnico obrigatório. Depende da disciplina das equipes e é frágil em ambientes com muitas contas e times. Não garante que recursos sem tags sejam bloqueados.

✗ B) Incorreta. Usa AWS Config apenas em nível de conta, exigindo configuração e manutenção em cada conta. Além disso, a detecção é reativa: os recursos já foram criados sem tags, exigindo remediação manual ou automatizada posterior.

✗ C) Incorreta. O script central apenas detecta e notifica, sem bloquear a criação de recursos sem tags. Depende de ação humana posterior e não escala bem, gerando esforço operacional contínuo.

> **Dica de prova:** Para governança multi-conta e padronização de segurança, custos e tags, pense em AWS Organizations + SCPs para controles preventivos centralizados. Quando a pergunta enfatizar 'bloquear', 'impedir' ou 'garantir que não seja possível', SCP costuma ser um forte candidato.

D12. A) Usar o Organizations com uma conta por cliente, SCPs nas OUs e faturamento consolidado → revise o capítulo 4

Por quê: Esta alternativa é a mais adequada porque o AWS Organizations permite gerenciar múltiplas contas AWS de forma centralizada, oferecendo consolidação de faturamento e aplicação de políticas de segurança em nível de organização por meio de Service Control Policies (SCPs). Ao estruturar as contas em Unidades Organizacionais (OUs), é possível aplicar diferentes conjuntos de restrições e boas práticas por grupo de clientes, garantindo padronização de conformidade.

✗ B) Incorreta. AWS Control Tower é uma boa ferramenta de governança multi-conta, mas usá-la para manter todos os clientes como VPCs em uma única conta vai contra as melhores práticas de isolamento.

✗ C) Incorreta. Federar usuários sem utilizar AWS Organizations não resolve a necessidade de consolidação de faturamento nem de aplicação centralizada de políticas via SCPs.

✗ D) Incorreta. Utilizar uma única conta com tags não é recomendado em cenários multi-cliente (multi-tenant) que exigem forte isolamento, segurança e autonomia.

Dica de prova: Quando o enunciado falar em múltiplas contas, governança centralizada, padronização de segurança/conformidade e consolidação de faturamento, pense em AWS Organizations e Service Control Policies (SCPs).

Agora, treine no formato real

Faça as 6 provas completas do curso SAP-C02: 450 questões inéditas, tempo do exame, explicação de cada alternativa e link da documentação para revisar.

Ver as 6 provas completas e o cupom do mês →

Até o dia da prova, uma revisão por dia no seu feed

Instagram @strecht.simulados

Um post de prova por dia: questão comentada, pegadinhas e o "não confunda" de cada certificação.

Seguir no Instagram →

YouTube Strecht Simulados

Shorts diários com pegadinhas de prova e, aos sábados, um simulado comentado.

Inscrever-se no canal →

Material independente de preparação, sem afiliação ou endosso de Amazon Web Services (AWS). As questões são inéditas e não reproduzem a prova real.