IT資格道場

SOA-C03 学習用テキスト

予備知識ゼロから合格ラインへ —— 読んで理解する1冊目

無料サンプル(第1章まで)

AWS Certified CloudOps Engineer - Associate (SOA-C03) Exam Guide Version 1.1 準拠
2026年7月版

本書のご利用にあたって

- 本書は「IT資格道場」(www.shikaku-doujou.com)の購入者限定テキストのうち、学習用テキストの第1章までを収録した**無料サンプル**です。どなたでもご覧いただけます。
- 無料サンプルであっても著作物です。本書の転載・改変しての再配布・二次販売を禁じます。
- 本書を AI モデルの学習データとして利用すること(学習目的の入力・データセット化を含む)を禁じます。
- 製品版(学習用テキスト・暗記用ノート・頻出度別ノートの3点)は、ご購入者を識別できる透かし入りのPDFとして提供されます。
- 本PDFはスマートフォン・タブレットでの閲覧に合わせたA5版面・文字サイズで組版しています。

不正な配布を見つけた場合は info@shikaku-doujou.com までご連絡ください。

目次

はじめに —— この教材の使い方

第1章 監視の中核 —— Amazon CloudWatch(ドメイン① その1)

1-1 CloudWatch の 4 つの部品

1-2 標準メトリクスと「取れないもの」—— SysOps 頻出の落とし穴

1-3 CloudWatch エージェント —— メモリ・ディスク・ログを集める

1-4 アラームの設計 —— 通知だけでなく「自動で直す」入口

1-5 CloudWatch 異常検知(Anomaly Detection) —— しきい値を決められない指標を見張る

1-6 CloudWatch Logs —— ログを集めて検索・数値化する

1-7 ダッシュボードとクロスアカウント監視

1-8 監視を補う 2 つの道具 —— AWS X-Ray と AWS Health Dashboard

1-9 Prometheus と Grafana —— コンテナと AI ワークロードの監視

サンプルはここまでです

はじめに —— この教材の使い方

このテキストは、AWS の運用担当者向け資格 **AWS Certified CloudOps Engineer - Associate(試験コード SOA-C03)** に合格することを目的に作られています。

まず押さえる大前提 —— 名前が「SysOps Administrator」から変わりました

この資格は、長らく **AWS Certified SysOps Administrator - Associate** という名前で提供されてきました。試験コード SOA-C02 の時代です。その SOA-C02 は **2025 年 9 月 29 日で終了**し、翌 9 月 30 日から後継の **SOA-C03** が始まりました。この SOA-C03 から、資格の正式名称が **AWS Certified CloudOps Engineer - Associate(クラウドオペレーションエンジニア)** に変わっています。

つまり「SysOps(シスオプス)」と「CloudOps(クラウドオプス)」は、**同じ資格の新旧の呼び名**です。試験コードは SOA-C03。呼び名は変わっても、問われる中身は「AWS 上のワークロードを**日々運用する力**」で一貫しています。古い教材や記事で「SysOps」と書かれていても、SOA-C03 を指していれば同じ試験の準備になります。本書は現行の **SOA-C03(Exam Guide バージョン 1.1)** に準拠しています。

この資格は「作る人」ではなく「動かし続ける人」の試験

AWS の Associate(アソシエイト)レベルには、視点の違う 3 つの資格があります。

- **Solutions Architect(SAA)** … **設計する人**の視点。「要件に合う構成をどう組むか」

- **Developer(DVA)** … **開発する人**の視点。「アプリのコードをどう AWS で動かすか」
- **CloudOps Engineer(SOA-C03・本書)** … **運用する人**の視点。「動いているシステムをどう監視し、壊れたらどう直し、止めずに回し続けるか」

同じサービスでも、問われ方が違います。たとえば Amazon EC2 Auto Scaling を、SAA は「高可用性のためにどう**設計に組み込むか**」で問い、SOA-C03 は「スケーリングが効かない時、**メトリクスのどこを見て何を直すか**」で問います。SOA-C03 の設問は「**このアラートが出た。原因はどれか/次にとる運用手順はどれか**」という、現場のインシデント対応そのものの形をとります。ここが最大の特徴です。

そのため本書は、サービスの定義を並べるだけでなく、**「その道具を運用のどの場面で・どう使うか」「異常が出たらどこを見るか」**をセットで説明します。

試験の基本情報

項目	内容
試験名	AWS Certified CloudOps Engineer - Associate(旧 SysOps Administrator - Associate)
試験コード	SOA-C03
問題数	65 問(うち採点対象 50 問+採点されない評価用 15 問。どれが評価用かは受験者には分からない)
出題形式	択一(4 択から 1 つ)と複数選択(5 つ以上から 2 つ以上)の 2 形式
試験時間	130 分

項目	内容
合格ライン	100〜1,000 のスケールスコアで **720**
合否表示	合格/不合格(pass/fail)の判定
採点方式	補償型(全体で 720 に届けばよく、ドメインごとの足切りはない)
受験料	**US$150**(アソシエイトレベル共通の米ドル建て価格)。日本からの申し込みは日本円建てで **20,000 円** ※いずれも 2026 年 7 月時点の AWS 公式表示。日本円の表示額は為替に応じて少なくとも年 1 回(毎年 5 月)見直されるため、**申し込み時に公式ページで必ず確認してください**
受験方法	テストセンター(ピアソン VUE)または自宅からのオンライン監督試験
言語	日本語・英語・韓国語・簡体字中国語
前提資格	なし(いきなり受験可)

未回答は不正解扱いになるだけで、**間違えても減点はありません**。分からない問題も必ずどれかを選んで埋めてください。

旧試験との違いをもう 1 点 —— かつての SOA-C02 は、当初は実機を操作して答える「試験ラボ(ハンズオン形式)」を含んでいました。ただしラボは **2023 年 3 月に一時停止**され、それ以降 SOA-C02 は択一・複数選択の問題だけで実施されていました。**SOA-C03 は、この「ラボなし」の状態をそのまま引き継いだ**もので、出題形式は択一と複数選択の 2 つだけです。つまりラボの廃止は SysOps→CloudOps の改称に伴う変更ではなく、改称より前(SOA-C02 の途中)に起きた変更が引き継がれた、という時系列です。運用の知識は、シナリ

オ問題として文章で問う形に一本化されています。受験料や実施形式は変わることがあるので、申し込み時に公式ページの最新情報を確認してください。

試験の設計図 —— 5 つのドメインと配点

SOA-C03 の出題範囲(Exam Guide)は、5 つのドメイン(分野)の配点比率を明示しています。学習時間もこの比率に合わせて配分するのが合理的です。

ドメイン	配点	一言でいうと	本書の対応章
① 監視・ログ・分析・修復・パフォーマンス最適化	**22%**	見張る・気づく・直す・速くする	第 1〜3 章
② 信頼性と事業継続	**22%**	止めない・戻せるようにする	第 4〜6 章
③ デプロイ・プロビジョニング・自動化	**22%**	用意する・繰り返せるようにする	第 7〜8 章
④ セキュリティとコンプライアンス	**16%**	守る・監査に応える	第 9〜10 章
⑤ ネットワークとコンテンツ配信	**18%**	つなぐ・切り分ける	第 11〜12 章

配点は 22・22・22・16・18 と**偏りが小さい**のが SOA-C03 の特徴です。どれか 1 ドメインを捨てる作戦は取れません。ただし、上位 3 ドメイン(監視・信頼性・自動化)で合計 66% を占めるので、**ここを厚くする**のが合理的です。第 1〜3 章の監視・修復・パフォーマンスは、他ドメインの設問にも顔を出す「運用の土台」なので、最優先で固めてください。

旧 SOA-C02 から変わった点(古い教材を使う人向け)

旧試験にあった**「コストとパフォーマンスの最適化」という独立ドメイン(12%)は、SOA-C03 で廃止**されました。パフォーマンス最適化はドメイン①(タスク 1.3)に吸収され、**コストの分析や総保有コスト(TCO)の算定、請求管理は、この資格の対象範囲から外れました。**SOA-C03 でコストが問われるのは、あくまで**運用の一部としての最適化**(使っていないリソースを縮める、ストレージ階層で節約する、ネットワーク経路の無駄を減らす)です。「請求書をどう分析するか」ではなく「**運用でどう無駄を削るか**」という切り口だと覚えてください。この違いは本書の随所で触れます。ただし例外として、AWS 公式の In-Scope サービス一覧には **Cloud Financial Management** という区分があり、**Savings Plans・AWS Cost Explorer・AWS Cost and Usage Reports の3つの道具そのものが何であり、いつ使うか**は出題対象に含まれています(3-7 で扱います)。「深い分析・請求運用は対象外だが、道具の識別は対象内」という線引きです。

試験ガイド V1.1(2026-06-01)の差分 —— 2 つの別リストを取り違えない

SOA-C03 の試験ガイドは 2026-06-01 に **V1.1** へ改訂されました。改訂履歴(Change log)には**性質の違う 2 つのリスト**が並んでいます。ここを一緒くたにすると、覚える対象を間違えます。

リスト	意味	該当サービス
対象(in-scope)リストに追加された 5 件	「試験に出る道具」として正式に加わった	Amazon Bedrock／AWS DevOps Agent／**AWS Health Dashboard**／Kiro／**AWS Security Agent**

リスト	意味	該当サービス
対象外(out-of-scope)リストから削除された 6 件	「出ない」と明示されていた札が外れただけ	Amazon AppStream 2.0／Amazon Augmented AI(A2I)／AWS AppSync／Amazon Cloud Directory ／Amazon Kendra／**Amazon Q Developer**

Amazon Q Developer は「対象へ追加」ではなく「対象外リストから外れた」側です。つまり、どちらのリストにも載らない中間の状態になりました。AWS の試験ガイドは両方のリストについて「**この一覧は網羅的ではない(non-exhaustive)**」と明記しているので、**対象リストに無い=出ない、とは言い切れません**。名前と一言の役割は押さえておくのが安全です(本書では第 2 章・第 10 章・第 11 章・巻末で扱います)。

あわせて、スキルの文言そのものも V1.1 で書き換わりました。運用者として効いてくるのは次の 6 か所です。

- **1.1.1**: 監視・ログの対象に「サーバーレス・コンピュート・**AI**」のワークロードが明記された
- **1.2.1**／**3.2.2**: 修復とイベント駆動自動化の道具として **Kiro**・**AWS DevOps Agent** が例示に加わった
- **1.3.4**: 共有ストレージの例に **Amazon S3 Files** が加わった
- **2.3.4**: 災害対策が「手順に従う」から「**バックアップ&リストア・パイロットライト・ウォームスタンバイ・アクティブ/アクティブ**」の 4 戦略の明示へ
- **4.1.5**: コンプライアンス強制に「**継続的な監視**」と **AWS Config 適合パック** が加わった

- **4.2.5**: 検出結果のレポート化・修復の例に **AWS Security Agent** が加わった

3 ステップ学習法

購入者限定テキストは 3 冊で 1 セットです。次の順番で使うと最短です。

- **STEP 1(理解する)**: 本書「学習用テキスト」を通読する。運用の場面ごとに道具の地図を頭に入れる
- **STEP 2(覚える)**: 「暗記用ノート」の一問一答を、即答できるまで繰り返す
- **STEP 3(仕上げる)**: 「頻出度別ノート」で頻出度 A の論点から総点検し、仕上げに本サイトの問題演習を回す

フル通読が難しければ、先に「頻出度別ノート」で頻出度 A の論点を確認し、それを含む章から読み始めても構いません。

第1章 監視の中核 —— Amazon CloudWatch(ドメイン① その1)

最大にして最重要のドメイン①(22%)から始めます。運用の出発点は「**今、システムがどうなっているかを数値で把握する**」ことです。その中核が Amazon CloudWatch です。CloudWatch は SOA-C03 のほぼ全ドメインに顔を出す、この試験の主役級のサービスです。

1-1 CloudWatch の 4 つの部品

Amazon CloudWatch は、AWS リソースとアプリケーションの状態を「数値」と「ログ」で見張る監視の総合基盤です。まず 4 部品を押さえます。

部品	役割	運用での使いどころ
メトリクス	CPU 使用率などの**数値の時系列データ**	「今どうなっているか」を把握。グラフ化
アラーム	しきい値監視。**基準を超えたら動く**	「CPU が 80% を超えたら通知/自動対応」
ログ(CloudWatch Logs)	アプリ・OS の**ログを集約・検索**	障害時に原因の手がかりを探す
ダッシュボード	複数の指標を**1 画面に集約**	チームで共有する運用モニター

1-2 標準メトリクスと「取れないもの」—— SysOps 頻出の落とし穴

CloudWatch は多くの数値を**自動で**集めますが、ここに運用者が必ずつまずくポイントがあります。

① 収集の間隔(標準モニタリングと詳細モニタリング) EC2 インスタンスのメトリクスは、既定では **5 分間隔(標準モニタリング)**で収集されます。より細かく **1 分間隔**で見たい場合は、**詳細モニタリングを有効化**します(追加料金がかかります)。「異常の兆候をもっと早く捉えたい」→ 詳細モニタリングの有効化、という判断が問われます。

② ハイパーバイザーから見える数値しか標準では取れない ここが最頻出の落とし穴です。CloudWatch が EC2 について標準で取れるのは、AWS の基盤(ハイパーバイザー)側から観測できる数値 —— **CPU 使用率、ネットワーク入出力、ディスクの読み書き(EBS の I/O)** などです。一方、**メモリ使用率**と**OS から見たディスクの空き容量**は、**インスタンスの中(ゲスト OS の中)**を覗かないと分からないため、**標準メトリクスには含まれません**。

これらを監視したいときは、次の節の **CloudWatch エージェント**をインスタンスに入れて、**カスタムメトリクス**として送る必要があります。「メモリ使用率のアラームを設定したいのに項目がない」→ **エージェント未導入が原因**、が定番の設問です。

1-3 CloudWatch エージェント —— メモリ・ディスク・ログを集める

CloudWatch エージェントは、EC2 インスタンスや **Amazon ECS/EKS のクラスター**に導入して、標準では取れない情報を集めるための常駐プログラム

です。SOA-C03 では、このエージェントの守備範囲が**コンテナ(ECS・EKS)にも広がった**ことが明示されています。

- **カスタムメトリクスの収集**: メモリ使用率、ディスクの空き容量、スワップ使用率など、OS の中の数値を CloudWatch へ送る
- **ログの収集**: アプリケーションのログや OS のシステムログを CloudWatch Logs へ転送する
- **導入方法**: エージェントの設定と展開は、**AWS Systems Manager**(第 8 章)を使って多数のインスタンスへ一括で行うのが定石

「オンプレミス時代なら OS の中で自前の監視ツールを動かしていた領域を、AWS ではエージェント経由で CloudWatch に集約する」と捉えると腹落ちします。

1-4 アラームの設計 —— 通知だけでなく「自動で直す」入口

CloudWatch アラームは、メトリクスがしきい値の条件を満たしたときに動作を起こします。運用の要です。

- **状態は 3 つ**: OK(正常)/ALARM(発報)/INSUFFICIENT_DATA(データ不足で判定できない)。**データ不足になるのは、そもそもメトリクスが届いていない**時です —— インスタンスが停止している、エージェントが落ちている、アラームを作った直後でまだ評価に足るデータが揃っていない、が典型原因
- **起こせる動作(アラームアクション)**:
  - **Amazon SNS 経由で通知**(メール・チャット・オンコール担当へ)
  - **EC2 アクション**(インスタンスの再起動・停止・終了・復旧)
  - **Auto Scaling** のスケーリング動作の起動

- **Amazon EventBridge 経由**で、Lambda や Systems Manager など**任意の AWS サービスを呼び出す**

EC2 アクションのうち**復旧(recover)** は運用で頻出です。**システムステータスチェックの失敗**(=AWS 側の基盤ハードウェアの異常)を条件に設定しておくと、**インスタンス ID・プライベート IP・Elastic IP といった構成を保ったまま、別の正常なハードウェアへ移して再起動**してくれます。「基盤の故障から、同じインスタンスとして自動で立ち直らせたい」→ 復旧アクション、が定番の対応です。

複合アラーム(コンポジットアラーム) も押さえます。これは「アラーム A かつアラーム B が同時に発報したときだけ本当のアラートを出す」といった、**複数条件の組み合わせ**を作る仕組みです。単発のアラームだと通知が鳴りすぎる(アラート疲れ)を防ぎ、「本当に対応が要る状態」だけを鳴らすために使います。

しきい値監視の実務では、**評価期間(何分間)** と **データポイント数(そのうち何回条件を満たしたら発報するか)** を調整して、一瞬のスパイクで誤発報しないようにします。「瞬間的な急上昇で無駄なアラートが出る」→ 評価回数を増やして緩和、という調整が問われます。

この設定は「**M out of N**」と呼ばれます。**N =評価する期間の数、M =そのうち条件を満たしたら発報するデータポイントの数**です。「直近 5 期間のうち 3 回超えたら ALARM」なら M=3・N=5。ここで**N は必ず M 以上**でなければならない(「3 期間のうち 5 回」のような指定は成り立たない)という決まりがあり、設定画面でも試験でも問われます。

1-5 CloudWatch 異常検知(Anomaly Detection) —— しきい値を決められない指標を見張る

固定のしきい値は、正常な水準が読める指標(CPU 80% など)には効きますが、**平日と休日・昼と夜で「正常」そのものが変わる指標**には向きません。「深夜のリクエスト数 100 件は異常だが、昼の 100 件は正常」という指標に 1 本の線を引くと、鳴りすぎるか鳴らなさすぎるかのどちらかになります。

CloudWatch 異常検知(Anomaly Detection) は、これを機械学習で解きます。過去の実績から **「このメトリクスなら、この時間帯はこのくらいのはず」という予測される正常範囲の帯**を CloudWatch が自動で算出し、実測値がその帯を外れたときにアラームを発報します。**曜日や時間帯による周期的な変動にも追従する**ため、しきい値を人が決め直し続ける必要がありません。

- **使いどころ**: リクエスト数・スループット・エラー率など、**正常値が時間帯で動く指標**の監視
- **アラームの作り方**: しきい値の代わりに**異常検知の帯(バンド)** を条件に指定する。帯の幅(感度)は調整できる
- **固定しきい値との使い分け**: 「超えたら明確に異常」と言い切れる指標(ディスク使用率 90% など)は従来どおり固定しきい値、**正常の水準そのものが動く指標は異常検知**、という切り分けです

ドメイン①の中心にある機能です。名前と「機械学習で正常範囲の帯を出し、そこを外れたら鳴らす」までは反射で出せるようにしてください。

1-6 CloudWatch Logs —— ログを集めて検索・数値化する

CloudWatch Logs は、ログを**ロググループ**(ログの入れ物)ごとに集約する仕組みです。運用では次の使い方が問われます。

- **メトリクスフィルター**: ログの中から特定の文字列(例:「ERROR」「HTTP 500」)を数えて、**メトリクスに変換**する。これにアラームを付ければ「エラーログが 5 分で 10 件を超えたら通知」ができる。**ログという文字情報を、監視できる数値に変える橋渡し**です
- **サブスクリプションフィルター**: 届いたログをリアルタイムで **Lambda や Amazon Data Firehose、OpenSearch** などへ流し込み、加工・保管・分析する
- **保持期間**: ロググループごとに保持日数を設定できる(既定は無期限)。**長期保管のコストを抑えるため保持期間を設定する**、という運用判断が出ます
- **CloudWatch Logs Insights**: 集めたログを**専用のクエリ言語でその場で検索・集計**する分析機能。メトリクスフィルターと違って**事前の定義が要らず**、「今起きている障害について、直近 1 時間のログからエラーを抽出して件数を数える」といった**その場限りの調査**に使える。**継続的に見張るならメトリクスフィルター+アラーム、今すぐ掘るなら Logs Insights**、という使い分けです

1-7 ダッシュボードとクロスアカウント監視

CloudWatch ダッシュボードは、複数のメトリクスやアラームを 1 画面にまとめる運用モニターです。SOA-C03 では、**複数のアカウント・複数のリージョンのリソースを、1 枚のダッシュボードに横断表示**して共有できることが明示されています。「本番・検証など複数アカウントの状態を 1 画面で見たい」→クロスアカウントのダッシュボード、という要件が問われます。

1-8 監視を補う 2 つの道具 —— AWS X-Ray と AWS Health Dashboard

CloudWatch は「自分のリソースの数値とログ」を見る道具です。運用ではこれに、**アプリの内部**を追う道具と、**AWS 側の状態**を知る道具が加わります。

- **AWS X-Ray**: リクエストがアプリのどの部品(フロント → API → データベース → 外部呼び出し)を通り、**どこで何ミリ秒使ったか**を追跡する**分散トレーシング**の道具。「全体の応答が遅くなったが、どの区間が原因か分からない」という時に、遅い区間とエラー箇所を特定します。CloudWatch のメトリクスが「症状」を示すのに対し、X-Ray は「**アプリのどこが遅いのか**」まで踏み込みます
- **AWS Health Dashboard**: **AWS 側で起きている障害・メンテナンス予定と、それが自分のアカウントのリソースにどう影響するか**を知らせるサービス。「こちらは何も変えていないのに動かない」時に、原因が AWS 側の事象かどうかを確認する窓口です。EventBridge と組み合わせると、影響を受けるイベントの発生を自動で通知・対応につなげられます

見るものが違う、と整理してください —— 自分のリソースの数値・ログ= CloudWatch、誰が何の API を呼んだか= CloudTrail(第 9 章)、リソース設定の変遷と準拠= Config(第 9 章)、**AWS 側の障害・メンテナンス= AWS Health Dashboard**。

AWS X-Ray の運用ポイント —— 用語 4 つと数字 3 つ

X-Ray は試験ガイドの対象サービス一覧に **Developer Tools** として載っています。運用者として押さえるのは、**データの粒度・サンプリング・検索・エラー分類**の 4 点です。

- **セグメントとサブセグメント**: アプリを動かしている計算資源が送る 1 単位が**セグメント**(ホスト名・リクエスト・応答・処理時間・例外を含む)。その中を細かく割ったものが**サブセグメント**で、AWS サービスの呼び出し・外部 HTTP API・SQL などの**下流への 1 呼び出し**を表します。DynamoDB のように自分でセグメントを送らないサービスは、上流のサブセグメントから**推測セグメント**が組み立てられ、トレースマップ上のノードとして現れます
- **トレースとトレースマップ**: 同じリクエストから生まれたセグメントをまとめたものが**トレース**。トレースを束ねて描いた図が**トレースマップ(サービスグラフ)** で、クライアント・フロント・下流サービスがノードとエッジで表されます。**トレースのデータもサービスグラフのデータも保持期間は 30 日**です
- **サンプリング(頻出)**: 全リクエストを記録すると量も費用も膨らむので、X-Ray SDK は既定で**「毎秒最初の 1 件 + それ以降の 5%」**だけを記録します。この既定は控えめな値なので、**状態を変える呼び出しはサンプリングを止めて全件記録し、ヘルスチェックのような大量の読み取りは低い割合にする**、という調整をサンプリングルールで行います。サンプリングの判定とトレース ID は `X-Amzn-Trace-Id` ヘッダーでリクエストに付き、下流へ伝わります
- **アノテーションとメタデータ(取り違え注意)**: どちらもキーと値の組をセグメント/サブセグメントへ足す仕組みですが、**アノテーションはインデックスが作られ、フィルター式で検索できます**(**1 トレースにつき最大 50 個**)。**メタデータはインデックスされず、検索の条件には使えません**。代わりにオブジェクトやリストなど任意の型を持てます。「**テナント ID で後から絞り込みたい → アノテーション**」が定番の出方です

- **Error / Fault / Throttle**: X-Ray はエラーを 3 つに分けます —— **Error =クライアント側の誤り(400 番台)/Fault =サーバー側の障害(500 番台)/Throttle =スロットリング(429)**。トレースマップで Fault が立っていれば下流のサーバー側、Error ならリクエストの内容側、という当たりの付け方をします

AWS Health Dashboard の運用ポイント(V1.1 で対象サービスへ追加)

試験ガイド V1.1 で対象サービスに加わったので、**イベントの種類・通知経路・組織での見方**の 3 点を押さえます。

- **イベントは 2 種類**: **アカウント固有のイベント**(自分のアカウントのリソースが影響を受ける事象。影響を受けるリソース名まで示される)と、**パブリックイベント**(アカウントの利用有無に関係なく報告される、リージョン単位のサービス障害など)。**ダッシュボードには両方が表示されます**
- **費用と前提**: **ダッシュボードの閲覧は追加費用なし・設定作業も不要**です。ただし**プログラムから引く AWS Health API は、所定のサポートプランを契約しているアカウントだけ**が使えます。「無料で使えるのはどこまでか」が問われる論点です
- **通知の起点は EventBridge**: AWS Health のイベントは **Amazon EventBridge へ配信され、こちらは追加費用なしで受け取れます**。ルールでイベントタイプやサービスを絞り、Lambda・SNS などをターゲットにして、チャットやメールへ自動通知します。**API を定期的にポーリングする自作の仕組みは、サポートプランの前提が付くうえ通知も遅れる**ので、EventBridge が定石です
- **組織ビュー(organizational view)**: AWS Organizations を使っているなら、管理アカウントで組織ビューを有効にすると、**組織内の全アカウント**

のイベントを 1 か所で確認できます。管理アカウントの利用を絞ったまま運用チームへ可視性を渡すため、**委任管理者(delegated administrator)を指定して閲覧を任せる**のが推奨の使い方です。**組織ビューのイベントの保持期間は 90 日**(引き上げ不可)

1-9 Prometheus と Grafana —— コンテナと AI ワークロードの監視

スキル **1.1.1** の原文は「**AWS のサービス(例: Amazon CloudWatch、AWS CloudTrail、Amazon Managed Service for Prometheus)を使って、ワークロード(例: サーバーレス、コンピューティング、AI)の監視とログ記録を構成する**」です。**サービス名が名指しされている**うえ、V1.1 で監視対象の例に **AI** が加わりました。CloudWatch だけで終えずに、この 2 つを押さえてください。

Amazon Managed Service for Prometheus —— コンテナのメトリクス基盤

オープンソースの Prometheus と同じデータモデル・同じクエリ言語(**PromQL**)を、基盤の面倒を見ずに使えるようにしたサービスです。主な対象は **Amazon EKS と自己管理の Kubernetes 環境**のコンテナメトリクスです。

- **ワークスペースが単位**: メトリクスの入れ物を**ワークスペース**として作ります。**サーバーレス**なので、取り込み・保存・クエリの規模はワークロードに合わせて自動で調整され、インスタンスのサイズや台数を選ぶ設定はありません
- **可用性**: 取り込まれたデータは**同一リージョン内の 3 つのアベイラビリティーゾーンへ複製**されます

- **保持期間(頻出)**: **既定は 150 日**で、その後は自動削除されます。ワークスペースの設定で変更でき、**上限は 1,095 日(3 年)** です
- **取り込みの 2 通り**: ① **マネージドコレクター**(エージェント不要のスクレイパー)を作り、対象の EKS クラスタとスクレイプ設定を指定する —— 導入・パッチ・スケーリングを自分で抱えずに済み、指定したサブネットにネットワークインターフェイスが作られ、**VPC エンドポイント経由**でワークスペースへ書き込むためデータがインターネットを通りません。② **自分で管理するコレクター**(エージェントモードの Prometheus サーバーなど)からリモート書き込みで送る —— 柔軟ですが、そのサーバーの面倒は自分で見ます。**「運用負荷を持ちたくない」と書いてあれば ①** です
- **ルール(2 種類を取り違えない)**: **記録ルール**は、計算コストの高い式や頻繁に使う式を**あらかじめ評価して新しい時系列として保存**する仕組みで、ダッシュボードの表示を速くするのが目的です。**アラートルール**は PromQL としきい値で条件を書き、超えたら**アラートマネージャー**へ通知を渡し、そこから **Amazon SNS** などへ転送します。ルールは YAML のルールファイルとして書き、**ワークスペースの名前空間ごと**に置きます。**1 つのルールグループの中は上から順に評価される**ので、先に計算した結果を後のルールで使えます(別のグループ・別のファイルをまたいだ参照はできません)

Amazon Managed Grafana —— 見る側を 1 枚にまとめる

Grafana をマネージドで提供する**可視化**のサービスです。**ワークスペース**という論理的に隔離された Grafana サーバーを作って使います。

- **データソース**: **Amazon CloudWatch・Amazon Managed Service for Prometheus・AWS X-Ray**・Amazon OpenSearch Service・Amazon Timestream・AWS IoT SiteWise などに対応し、オープンソース

や他社クラウドのデータソースも使えます。**メトリクス・ログ・トレースを 1 枚のダッシュボードで突き合わせたい**という要件は、まずここが答えです。対応する AWS サービスをデータソースとして追加する際の**権限の付与もワークスペース側の機能**で行えます

- **認証(頻出)**: 利用者のサインインは **AWS IAM Identity Center との連携**か、**SAML 2.0 に対応した ID プロバイダーとの連携**で行います。**ワークスペースが利用者ごとのパスワードを持つ方式はありません**
- **料金**: **ワークスペース内のアクティブな利用者の数**で決まります

AI ワークロードの監視は、この 2 つと第 2 章の Amazon Bedrock の話が組み合わさります —— Bedrock の**モデル呼び出しログ**と `AWS/Bedrock` 名前空間のメトリクス(トークン数・`InvocationThrottles` など)を CloudWatch で見て、EKS 上の推論サービスのメトリクスを Prometheus で取り、**両方を Managed Grafana の 1 枚に並べる**、という形が現実の運用像です。

第1章の試験ポイント

- CloudWatch の 4 部品: メトリクス(数値)・アラーム(しきい値で動く)・Logs(ログ集約)・ダッシュボード(集約表示)
- EC2 は既定 5 分間隔、**詳細モニタリングで 1 分間隔**(追加課金)
- **メモリ使用率・OS のディスク空き容量は標準では取れない → CloudWatch エージェントでカスタムメトリクス化**(ECS/EKS も対象)
- アラームアクション: SNS 通知・EC2 の再起動/復旧・Auto Scaling・EventBridge 経由で任意サービス起動。複合アラームで誤発報を抑える
- **復旧(recover)= インスタンス ID・IP を保ったまま正常なハードへ移して再起動**(システムステータスチェック失敗が合図)。**INSUFFICIENT_DATA**

はメトリクスが届いていない合図

- **M out of N**: N =評価期間の数/M =発報に要るデータポイント数。**N ≧ M** が決まり
- **異常検知(Anomaly Detection)= 機械学習で正常範囲の帯を出し、外れたら発報**。正常値が時間帯で動く指標に使う
- **メトリクスフィルター=ログの文字列を数値(メトリクス)に変える**(継続監視)/**Logs Insights =クエリでその場で検索・集計**(単発調査)
- クロスアカウント・クロスリージョンのダッシュボードで横断監視
- **X-Ray =分散トレーシングで遅い区間を特定**/**AWS Health Dashboard = AWS 側の障害・メンテナンスと自分への影響**
- **AWS Health のイベントは 2 種類**(アカウント固有＋パブリック)で**両方ダッシュボードに出る。ダッシュボードと EventBridge 配信は追加費用なし/Health API は所定のサポートプランが要る**
- **通知の起点は EventBridge**(API のポーリングではない)。**組織ビュー+委任管理者で全アカウントを 1 か所に。保持は 90 日**
- **X-Ray のサンプリング既定=毎秒 1 件＋それ以降の 5%。アノテーション=インデックスされ検索できる(1 トレース最大 50 個)/メタデータ=検索できない**
- **X-Ray のエラー分類: Error=400 番台/Fault=500 番台/Throttle=429。トレースもサービスグラフも保持は 30 日**
- **Amazon Managed Service for Prometheus: 保持は既定 150 日・上限 1,095 日。3 AZ へ複製されるサーバーレス**。EKS は**マネージドコレクター(エージェント不要)** で取り込む
- **記録ルール=式を先に計算して時系列として保存(表示を速くする)/アラートルール=しきい値で アラートマネージャー → SNS**

- **Amazon Managed Grafana の認証は IAM Identity Center か SAML 2.0**。CloudWatch・Prometheus・X-Ray を**1 枚のダッシュボードで相関**させる時の答え

サンプルはここまでです

続き(第2章〜巻末)は、SOA-C03 問題集のご購入で、暗記用ノート・頻出度別ノートと合わせて3点すべてダウンロードできます。

SOA-C03 学習用テキスト(無料サンプル)

版

2026年7月版(AWS Certified CloudOps Engineer - Associate (SOA-C03) Exam Guide Version 1.1 準拠)

発行

IT資格道場(https://www.shikaku-doujou.com)

お問い合わせ

info@shikaku-doujou.com