Amazon EKS オブザーバビリティを合理化するためのベストプラクティス

AWS 規範ガイダンス

AWS 規範ガイダンス: Amazon EKS オブザーバビリティを合理化するためのベストプラクティス

Table of Contents

Amazon EKS オブザーバビリティを合理化するためのベストプラクティス

Ishwar Chauthaiwale、Navean Suthar、Pratap Kumar Nanda、Amazon Web Services (AWS)

2026 年 3 月 (ドキュメント履歴)

Amazon Elastic Kubernetes Service (Amazon EKS) には、コンテナ化されたワークロードを効果的にモニタリングおよびトラブルシューティングするための包括的なオブザーバビリティソリューションが必要です。分散システムとマイクロサービスは Amazon EKS 環境で複雑なアーキテクチャであるため、信頼性の高いオペレーションを維持するには、適切なオブザーバビリティプラクティスを実装することが重要です。Amazon EKS 環境での効果的なオブザーバビリティにより、チームはアプリケーションのパフォーマンスに関する深い洞察を得て、問題を効率的にトラブルシューティングし、最適なクラスターの状態を維持できます。

課題は、組織の目標と業界標準に沿ったベストプラクティスに従いながら、Amazon EKS オブザーバビリティに使用できるツールと手法の膨大な エコシステムをナビゲートすることです。効果的なオブザーバビリティ戦略では、包括的なデータ収集とパフォーマンス上の考慮事項、費用対効果、スケーラビリティのバランスを取る必要があります。

このガイドは、組織が以下の領域で Amazon EKS オブザーバビリティを最適化するのに役立つように設計されています。

- 効率的なログ記録メカニズムの確立
- 堅牢なモニタリングソリューションの実装
- 複雑なアーキテクチャでの分散トレースの使用
- アラートとインシデント対応戦略の実装

これらのベストプラクティスを採用することで、組織は Amazon EKS 環境に関する深い洞察を得る能力を強化できるため、信頼性、パフォーマンス、運用効率が向上します。このオブザーバビリティへの合理化されたアプローチは、トラブルシューティングとメンテナンスに役立ち、Kubernetes ベースのアプリケーションとインフラストラクチャを継続的に改善するためのデータ駆動型の意思決定をサポートします。(Amazon EKS の詳細については、サービスドキュメントを参照してください)。

このガイドでは、Amazon EKS オブザーバビリティの各側面について詳しく説明し、小規模なアプリケーションから大規模で複雑なマイクロサービスアーキテクチャまで、Amazon EKS デプロイの特定のニーズに合わせて調整できるツールと戦略について説明します。

このガイドの内容

目的

このガイドは、お客様とお客様の組織が以下のビジネス目標を達成するのに役立ちます。

- 運用の可視性の向上 – 効果的なオブザーバビリティプラクティスを通じて、Amazon EKS クラスターとアプリケーションに関する包括的なインサイトを実現します。

  この目標は、Amazon EKS 環境全体で完全な可視性を維持することの重要性を強調しています。AWS X-Ray、Amazon CloudWatch Container Insights、 AWS Distro for OpenTelemetry などのツールは、システムの動作を理解し、問題をすばやく特定し、最適なパフォーマンスを維持するのに役立ちます。
- トラブルシューティング効率の向上 – 効果的なトレースとモニタリング戦略により、平均検出時間 (MTTD) と平均解決時間 (MTTR) を短縮します。

  この目標は、問題の迅速な特定と解決を可能にするオブザーバビリティプラクティスの実装に焦点を当てています。この目標を達成するには、分散トレース、効果的な口グ記録、包括的なメトリクス収集などの手法が重要です。
- プロアクティブパフォーマンス管理 – エンドユーザーに影響を与える前に、潜在的な問題を早期に検出できます。

  高いサービスの可用性とパフォーマンスを維持するには、プロアクティブモニタリングが不可欠です。この目標は、サービスの中断を防ぐために、適切なアラート、傾向分析、予測モニタリングを実装することの重要性に対処します。

- コスト効率の高いオブザーバビリティ – 包括的なシステムの可視性を維持しながら、オブザーバビリティコストを最適化します。

  コスト最適化には、効率的なサンプリング戦略、適切なデータ保持ポリシー、最適な計測アプローチの実装が含まれます。目標は、効果的なシステムモニタリングを確保しながら、オブザーバビリティのニーズとコスト上の考慮事項のバランスを取ることです。
- スケーラブルなモニタリングアーキテクチャ – オブザーバビリティソリューションが Amazon EKS 環境とシームレスにスケーリングされていることを確認します。

  この目標は、アプリケーションで拡張できるモニタリングソリューションの実装に焦点を当てています。単一のクラスターを実行しているか、マルチクラスター、マルチリージョンのデプロイを実行しているかにかかわらず、オブザーバビリティ戦略はそれに応じてスケーリングする必要があります。

Amazon EKS でのログ記録

ログ記録は、Amazon EKS で実行されるアプリケーションを管理および維持する上で重要な側面です。Amazon EKS 環境での効果的なログ記録プラクティスは、開発者、運用チーム、システム管理者が、コンテナ化されたアプリケーションとその基盤となるインフラストラクチャの動作、パフォーマンス、状態に関する貴重なインサイトを得るのに役立ちます。

Amazon EKS に堅牢なログ記録戦略を実装することは、いくつかの理由で不可欠です。

- トラブルシューティング: ログは問題をすばやく特定して診断するのに役立ちます。これにより、ダウンタイムが短縮され、システム全体の信頼性が向上します。
- コンプライアンス: 多くの業界では、監査と規制の目的で包括的なログ記録が必要です。
- セキュリティ: ログ分析は、潜在的なセキュリティの脅威や違反を検出して調査するのに役立ちます。
- パフォーマンスの最適化: ログはアプリケーションとシステムのパフォーマンスに関するインサイトを提供するため、ボトルネックを特定し、リソース使用率を最適化できます。
- モニタリングとアラート: ログデータを使用して、モニタリングシステムを設定し、特定のイベントまたは条件のアラートをトリガーできます。

このセクションの内容:

Amazon EKS でのログ記録のタイプ

Amazon EKS では、ログ記録には、Kubernetes クラスターのさまざまなコンポーネントによって生成されるさまざまなタイプのログデータのキャプチャ、保存、分析が含まれます。

- システムログ: 基盤となる Amazon Elastic Compute Cloud (Amazon EC2) インスタンスまたはAWS Fargateノードに関する情報
- Kubernetes コンポーネントログ: API サーバー、スケジューラ、コントローラーマネージャーなどのコア Kubernetes コンポーネントからのデータ
- コンテナランタイムログ: Docker や containerd などのコンテナランタイムからの情報

- アプリケーションログ: コンテナ化されたアプリケーションからの出力

Amazon EKS 環境のログを効果的に管理するには、通常 AWS のサービス、サードパーティーのツールとベストプラクティスを組み合わせて使用します。これには、Amazon CloudWatch、Fluent Bit、Elasticsearch、Kibana、その他のログ記録および分析ツールを使用してログデータを収集、保存、視覚化することが含まれます。

以下のセクションでは、Kubernetes クラスターに包括的なログ記録戦略を実装するためのベストプラクティス、ツール、手法など、Amazon EKS でのログ記録のさまざまな側面について説明しますAWS。

システムログ

Amazon EKS の基盤となる EC2 インスタンスまたは Fargate ノードのログ記録には、ノードタイプに応じて異なるアプローチが必要です。

Amazon EKS で EC2 インスタンスのログ記録を実装するには、次のツールを使用できます。

- CloudWatch エージェント: EC2 インスタンスに CloudWatch エージェントをインストールして設定します。`/var/log/messages` や などのシステムログを収集するように設定します`/var/log/secure`。ユーザーデータスクリプトまたは設定管理ツールを使用して、このプロセスを自動化できます。
- Fluent Bit: Fluent Bit を DaemonSet としてデプロイして、すべてのノードからログを収集します。ログを CloudWatch Logs またはその他の集中ロギングシステムに転送するように設定します。
- Container Insights: EKS クラスターの Container Insights を有効にして、EC2 インスタンスからメトリクスとログを自動的に収集します。
- カスタムスクリプト: カスタムスクリプトを開発して特定のログを収集し、任意のログ記録先に送信します。
- SSM エージェント: AWS Systems Manager エージェント (SSM エージェント) を使用してログを収集し、CloudWatch Logs に転送します。

Amazon EKS で Fargate ノードのログ記録を実装するには、次のツールを使用します。

- Fargate ログ記録: Fargate はコンテナから `stdout`と `stderr` ログを自動的に収集します。これらのログを CloudWatch Logs に送信するように Fargate プロファイルを設定します。

- Fargate の Fluent Bit: Fargate ログ記録専用の Fluent Bit イメージ AWS を提供します。Fargate ポッドにサイドカーコンテナとしてデプロイして、ログを収集して転送します。
- Container Insights for Fargate: Container Insights が Fargate ノードからメトリクスとログを収集できるようにします。

Kubernetes コンポーネントログ

Amazon EKS で API サーバー、スケジューラ、コントローラーマネージャーなどの Kubernetes コンポーネントからログを収集するには、アプリケーションのログ記録とは少し異なるアプローチが必要です。これらのコンポーネントは、 によって管理される Amazon EKS コントロールプレーンの一部として実行されます AWS。これらのログを収集してアクセスする方法は次のとおりです。

- コントロールプレーンのログ記録を有効にする: EKS クラスターのコントロールプレーンのログ記録は AWS マネジメントコンソール、、 AWS Command Line Interface (AWS CLI)、または AWS CloudFormationや Terraform などのInfrastructure as Code (IaC) ツールを使用して有効にできます。コントロールプレーンのログ記録を有効にすると、ログは Amazon CloudWatch Logs に送信されます。ログ`/aws/eks/<cluster-name>/cluster`グループの CloudWatch コンソールで表示できます。このロググループ内で、各コントロールプレーンコンポーネントには、次のように独自のログストリームがあります。

ストリーム名	説明
kube-apiserver	Kubernetes API サーバーログ
kube-scheduler	スケジューラ決定ログ
kube-controller-manager	コントローラーマネージャーログ
認証	IAM 認証ログ
監査	Kubernetes 監査ログ (明示的に有効にする必要があります)

  特定のコンポーネントのログを表示するには、クラスターロググループに移動し、ターゲットログストリーム名でフィルタリングします。
- CloudWatch Logs Insights を使用する: CloudWatch Logs Insights を使用して、ログに対して複雑なクエリを実行できます。

- ログを Amazon S3 にエクスポートする: 長期ストレージまたはさらなる分析のために、ログを Amazon Simple Storage Service (Amazon S3) にエクスポートできます。https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html
- サードパーティーのツールを使用: Fluent Bit などのツールを使用してこれらのログを収集し、Elasticsearch や Splunk などの他のログシステムに転送できます。
- 使用 AWS CloudTrail: AWS CloudTrailサービスは、EKS クラスターに対して行われた API コールに関する追加のインサイトを提供できます。

コンテナランタイムログ

Amazon EKS でのコンテナランタイムログのログ記録には、コンテナランタイムからのログのキャプチャと管理が含まれます。通常、これは Amazon EKS `containerd`用です。Amazon EKS でのコンテナランタイムログのログ記録にアプローチする方法は次のとおりです。

- Amazon EC2 ノードのログに直接アクセスします。セルフマネージド EC2 ノードの場合、次の場所からホストのコンテナランタイムログに直接アクセスできます。
  - `containerd` ログ: `/var/log/containers/`
  - Docker ログ (Docker ランタイムを使用している場合) 。 `/var/log/docker.log`
- ログ収集には DaemonSet を使用します。
- ログ収集エージェント (Fluent Bit など) を DaemonSet としてデプロイして、すべてのノードからログを収集します。
- コンテナランタイムログを収集するように CloudWatch エージェントを設定します。
- Container Insights を有効にして、コンテナランタイムメトリクスとログを収集します。
- Fargate を使用します。Fargate ノードの場合、コンテナランタイムログは自動的に収集され、CloudWatch Logs からアクセスできます。
- Fluent Bit や Logstash などのツールを使用して、カスタムログ記録ソリューションを実装します。CloudWatch アラームを設定するか、Prometheus などのツールを使用して、コンテナランタイムログの特定のパターンや問題をモニタリングします。Datadog、Splunk、Elastic Stack (ELK スタック) など、Kubernetes や Amazon EKS と適切に統合されるサードパーティーのログ記録ソリューションの使用を検討してください。ログ集約ツールを使用して複数のソースからログを収集し、一元化されたログ記録システムに転送します。

アプリケーションログ

Amazon EKS のアプリケーションログは、アプリケーションのメンテナンスとトラブルシューティングに不可欠です。Amazon EKS でアプリケーションログ記録を実装するには、以下のオプションから選択できます。

- ログを stdout/ に書き込むstderr: アプリケーションログを処理する最もシンプルで Kubernetes ネイティブな方法は、 stdoutと に書き込むことですstderr。Kubernetes はこれらのストリームを自動的にキャプチャします。
- ログ集約の実装: Fluent Bit などのログアグリゲータを使用して、すべてのポッドからログを収集します。
- ログルーティングを設定する: ログを目的の宛先 (CloudWatch Logs や Elasticsearch など) にルーティングするようにログアグリゲータを設定します。
- CloudWatch Container Insights を使用する: Container Insights を有効にして、包括的なログ記録とモニタリングを行います。

Amazon EKS でのログ記録のベストプラクティス

以下のベストプラクティスは、Amazon EKS 環境用の堅牢でスケーラブル、効率的なログ記録システムを作成し、Kubernetes クラスターのトラブルシューティング、モニタリング、全体的な管理を向上させるのに役立ちます。

- ログ収集の一元化: CloudWatch Logs、Elasticsearch、サードパーティーサービスなどの一元化されたログ記録ソリューションを使用して、すべてのコンポーネントのログを集約します。これにより、ログ分析のための単一のアクセスポイントが提供され、管理が簡素化されます。
- 構造化ログの実装: JSON などの構造化ログ形式を使用して、ログをより簡単に解析および検索できるようにします。タイムスタンプ、ログレベル、ソース識別子などの関連するメタデータを含めます。
- ログレベルを適切に使用する: アプリケーションに適切なログレベル (DEBUG、WARN、、 INFOなどERROR) を実装します。過剰なログ記録を避けるため、適切なレベルでログを記録するように本番環境を設定します。
- コンテナログ記録を有効にする: stdoutおよび にログ記録するようにコンテナを設定しますstderr。これにより、Kubernetes はこれらのログをキャプチャし、選択したログ記録ソリューションに転送できます。

- アプリケーションのログ記録を有効にする: ログファイルに書き込むstderr代わりに、stdoutおよび にログを書き込むようにアプリケーションを設定します。これは 12 要素のアプリ方法論に従い、クラウドネイティブのベストプラクティスと一致しています。
- ログ収集に Kubernetes DaemonSets を使用する: ログ収集エージェント (Fluent Bit など) を DaemonSets としてデプロイして、クラスター内のすべてのノードで実行されるようにします。
- 保持ポリシーの実装: 規制に準拠し、ストレージコストを管理するために、ログ保持ポリシーを定義して適用します。
- 安全なログデータ: 転送中および保管中のログを暗号化します。アクセスコントロールを実装して、ログを表示および管理できるユーザーを制限します。
- ログ取り込みのモニタリング: ログ取り込みの失敗または遅延のアラートを設定して、継続的なログ記録を確保します。
- Kubernetes 注釈とラベルを使用する: Kubernetes 注釈とラベルを使用してログにメタデータを追加し、検索可能性とフィルタリングを向上させます。
- 分散トレースの実装: AWS X-Rayや Jaeger などの分散トレースツールを使用して、マイクロサービス間でログを関連付けます。
- ログボリュームの最適化: 不要なコストやパフォーマンスの問題を回避するために、ログの内容を選択します。大量の低値ログにはサンプリングを使用します。
- ログ集約の実装: Logstash などのツールを使用して、複数のソースからログを集約してから、中央ログシステムに送信します。
- 可能な AWS のサービス 場合に使用する: CloudWatch Logs や Container Insights などのサービスは、他の とシームレスに統合できます AWS のサービス。
- ログ分析と視覚化を実装する: CloudWatch Logs Insights、Elasticsearch with Kibana、サードパーティーソリューションなどのツールを使用してログ分析と視覚化を行います。
- 自動ログ分析を実装する: 機械学習と AI を活用したツールを使用して、ログの異常とパターンを自動的に検出します。
- ログ記録戦略を文書化する: チームのログ記録アーキテクチャ、プラクティス、ツールを明確に文書化します。

Amazon EKS でのログ記録に関する重要な考慮事項

このセクションでは、Amazon EKS でログ記録を実装する際に留意すべき重要な考慮事項について説明します。

- パフォーマンスへの影響: 過剰なログ記録はアプリケーションのパフォーマンスに影響を与える可能性があります。生成されるログの量と頻度に注意してください。
- コスト管理: ログのストレージと処理には、特に大規模なコストがかかる可能性があります。ログ保持ポリシーを実装し、コストを削減するためにログ集約の使用を検討してください。
- セキュリティとコンプライアンス: ログにパスワードや個人データなどの機密情報が含まれていないことを確認します。転送中および保管中のログに暗号化を実装します。ログを処理するときは、一般データ保護規則 (GDPR) や医療保険の相互運用性と説明責任に関する法律 (HIPAA) などのコンプライアンス要件を検討してください。
- スケーラビリティ: ログ記録ソリューションがクラスターサイズとログボリュームに合わせてスケールできることを確認します。ログ送信にはバッファリングとバッチ処理の使用を検討してください。
- ログ保持: 適切なログ保持期間を定義して実装します。コンプライアンス要件とストレージコストのバランスを取ります。
- アクセスコントロール: ログアクセス用の適切な AWS Identity and Access Management (IAM) ロールとポリシーを実装します。ログ管理の最小特権の原則に従います。
- ログの整合性: さまざまなアプリケーションやサービスで一貫したログ形式を使用します。構造化ログ記録を使用すると、解析と分析が容易になります。
- 時刻同期: すべてのノードで時刻を同期して、ログに一貫したタイムスタンプを取得します。
- リソース割り当て: エージェントをログ記録するための適切なリソース (CPU やメモリなど) を割り当てます。ログ記録コンポーネントのリソース使用状況をモニタリングします。
- Fargate に関する考慮事項: Fargate には、EC2-basedノードとは異なる特定のログ記録メカニズムがあります。Fargate ログ記録の制限と機能について説明します。
- マルチテナントクラスター: マルチテナント環境では、テナント間でログが適切に分離されていることを確認します。
- ログ解析と分析: 効果的なログ分析に必要なツールとスキルを検討します。構造化データ抽出用のログ解析を実装します。
- ログ記録システムのモニタリング: ログ記録インフラストラクチャ自体のモニタリングを設定します。システム障害またはバックログのログ記録に関するアラートを生成します。
- ネットワークへの影響: ログ送信で使用されるネットワーク帯域幅に注意してください。ログデータに圧縮を使用することを検討してください。
- Kubernetes イベント: Kubernetes イベントを重要な情報のソースとして見落としないでください。

- コントロールプレーンのログ記録: コントロールプレーンのログ記録を有効にする場合の影響とコストを理解します。
- デバッグ機能: ロギングソリューションでデバッグとトラブルシューティングを簡単に行えることを確認してください。
- 既存のツールとの統合: Amazon EKS ログ記録ソリューションが既存のモニタリングおよびアラートツールとどのように統合されるかを検討してください。
- テスト: 特にクラスターのアップグレード後に、ログ記録の設定を定期的にテストします。
- ドキュメント: ログ記録のアーキテクチャとプラクティスを明確に文書化します。
- ログ集約レイテンシー: ログ集約のレイテンシーと、それがリアルタイムモニタリングにどのように影響するかに注意してください。

Amazon EKS でのモニタリング

Amazon EKS でのモニタリングは、Kubernetes ワークロードのヘルス、パフォーマンス、セキュリティに関する重要な可視性を提供します。適切なモニタリングを行わないと、サービスの中断、セキュリティ違反、非効率的なリソース使用率が発生し、事業運営に影響を与え、コストが増加する可能性があります。効果的なモニタリングにより、問題をプロアクティブに特定して解決し、リソースの使用を最適化し、コンテナ化されたアプリケーション全体でコンプライアンス要件を維持できます。包括的なモニタリングソリューションを実装することで、高可用性を確保し、異常を早期に検出し、Amazon EKS インフラストラクチャのスケーリングと改善のためにデータ駆動型の意思決定を行うことができます。

このセクションでは、さまざまなモニタリングタイプ、利用可能なツール、Kubernetes 環境の堅牢なモニタリング戦略の構築に役立つベストプラクティスなど、Amazon EKS モニタリングのさまざまな側面について説明します。

このセクションの内容:

Amazon EKS のモニタリングのタイプ

Amazon EKS の効果的なオブザーバビリティには、インフラストラクチャ、アプリケーション、セキュリティのモニタリングアクティビティが含まれます。

インフラストラクチャのモニタリング

インフラストラクチャモニタリングは Amazon EKS オブザーバビリティの基本的なコンポーネントであり、Kubernetes クラスターの基本的な要素のヘルスとパフォーマンスに関する深いインサイトを提供します。その中核となるのは、コントロールプレーンコンポーネントとワーカーノードの両方のバイタルを追跡し、基盤となるプラットフォームが安定して効率的であることを確認することです。

- API サーバー、 etcd データベース、スケジューラなどの主要なコンポーネントを監督するため、コントロールプレーンのモニタリングは重要です。API サーバーのレイテンシーをモニタリングすることで、アプリケーションのデプロイやスケーリングオペレーションに影響を与える可能性のあるパフォーマンスのボトルネックをすばやく特定できます。Etcd パフォーマンスモニタリングは、クラスターの状態データベースが効率的に動作することを検証し、クラスター全体に影響を与える可能性のあるデータ整合性の問題を防ぎます。
- ノードレベルのモニタリングは、コンテナ化されたワークロードを実行するコンピューティングリソースに焦点を当てているため、同様に重要です。これには、すべてのワーカーノードでの CPU 使用率、メモリ消費量、ディスク I/O、ネットワークパフォーマンスの追跡が含まれます。これらのメトリクスを理解することで、リソースの枯渇を防ぎ、ノードスケーリングの決定を最適化し、適切なキャパシティプランニングを確保できます。
- ネットワークモニタリングは、ポッド、サービス、外部リソース間の信頼性の高い通信を維持する上で重要な役割を果たします。ネットワークスループット、レイテンシー、および接続状態をモニタリングすることで、接続の問題を早期に特定し、アプリケーションのスムーズな通信を確保できます。ストレージモニタリングは、ボリュームのパフォーマンス、容量使用率、I/O パターンを追跡することでネットワークモニタリングを補完し、データ関連のボトルネックを防ぐのに役立ちます。

インフラストラクチャモニタリングは、潜在的な問題の早期警告システムとして機能し、プロアクティブメンテナンスを可能にし、最適なリソース割り当てを確保します。堅牢なインフラストラクチャモニタリングを行わないと、予期しないダウンタイム、パフォーマンスの低下、リソースの非効率的な使用がビジネスオペレーションとコストに大きな影響を与えるリスクがあります。

アプリケーションのモニタリング

アプリケーションモニタリングは、Amazon EKS 環境で正常でパフォーマンスが高く、信頼性の高いコンテナ化されたアプリケーションを維持するために不可欠です。このレベルのモニタリングでは、クラスター内で実行される実際のワークロードに焦点を当て、アプリケーションがどのように動作し、実行し、他のサービスとやり取りするかに関する重要なインサイトを提供します。

アプリケーションモニタリングには、コンテナレベルのモニタリング、サービスレベルのモニタリング、分散トレースが含まれます。

- コンテナレベルでは、アプリケーションモニタリングは、コンテナのヘルスステータス、再起動数、リソース消費パターンなどの重要なメトリクスを追跡します。これらのメトリクスは、過剰なリソースを消費したり、頻繁に再起動したりしている可能性がある問題のあるコンテナを特定するのに役立ちます。これは、メモリリークや設定の問題などの根本的な問題を示している可能性があ

ります。コンテナライフサイクルイベントをモニタリングすることで、適切なアプリケーション動作を確保し、デプロイの問題を迅速にトラブルシューティングできます。

- サービスレベルのモニタリングは、応答時間、エラー率、リクエストスループットなどのアプリケーションのパフォーマンスと信頼性のメトリクスを可視化します。これらのメトリクスは、サービスレベル目標 (SLOs) を維持し、ポジティブなエンドユーザーエクスペリエンスを確保するために不可欠です。さまざまなサービスエンドポイント間でレイテンシーを追跡し、パフォーマンスのボトルネックを特定し、エラーパターンをモニタリングしてアプリケーションの信頼性を維持できます。
- 分散トレースは、特にマイクロサービスアーキテクチャにおけるアプリケーションモニタリングのもう 1 つの重要な側面です。トレースを実装することで、さまざまなサービスを通過するリクエストを追跡し、依存関係を理解し、パフォーマンスのボトルネックを特定できます。このend-to-end可視性は、サービスインタラクションを最適化し、複数のコンポーネントにまたがる複雑な問題のトラブルシューティングに役立ちます。

カスタムアプリケーションメトリクスは、ビジネス固有のインサイトを提供する上で重要な役割を果たします。これには、注文処理率、ユーザーログイン頻度、トランザクション成功率などのメトリクスが含まれる場合があります。これらのカスタムメトリクスをインフラストラクチャおよびコンテナメトリクスと関連付けて、インフラストラクチャのパフォーマンスがビジネスオペレーションにどのように影響するかをよりよく理解し、スケーリングと最適化のためのデータ駆動型の意思決定を行うことができます。

アプリケーションモニタリングの重要性は、アプリケーションのヘルスとパフォーマンスを包括的に把握できることにあります。このモニタリングにより、高いサービス品質を維持し、問題を迅速に解決し、ビジネス目標を達成するためにアプリケーションを継続的に最適化できます。

セキュリティモニタリング

Amazon EKS のセキュリティモニタリングは、組織が Kubernetes 環境の整合性、機密性、コンプライアンスを維持するのに役立つ重要なアクティビティです。この包括的なセキュリティアプローチは、継続的な監視、脅威検出、コンプライアンスモニタリングを組み合わせて、コンテナ化されたワークロードを潜在的なセキュリティリスクや不正アクセスから保護します。これには、認証と認可のモニタリング、ネットワークセキュリティのモニタリング、設定とコンプライアンスのモニタリングが含まれます。

- 認証と認可のモニタリングは、クラスターへのアクセスのすべての試行を追跡することで、防御の最前線を形成します。これには、API サーバーリクエストのモニタリング、ログイン試行の成功と失敗の追跡、ロールベースのアクセスコントロール (RBAC) の変更の監査が含まれます。誰がいつ

どのリソースにアクセスしたかの詳細な監査ログを維持することで、潜在的なセキュリティ違反、不正アクセスの試み、特権エスカレーションアクティビティをすばやく検出できます。これは、厳格なアクセスコントロールを維持することが重要なマルチテナント環境で特に重要です。

- ネットワークセキュリティモニタリングは、ポッドとサービス間の不正な通信を検出して防止することに重点を置いています。ネットワークポリシー違反や異常なトラフィックパターンをモニタリングすることで、コンテナエスケープの試みやクラスター内の横移動などの潜在的なセキュリティ脅威を特定できます。これには、内部クラスター通信と外部トラフィックパターンの両方を追跡して、コンテナが承認されたエンドポイントとのみ通信し、定義されたセキュリティポリシーに従うようにすることが含まれます。
- 設定とコンプライアンスのモニタリングは、セキュリティベースラインを維持し、規制要件を満たすために不可欠です。これには、コンテナイメージの脆弱性の継続的なスキャン、ランタイムセキュリティのモニタリング、セキュリティ体制に影響を与える可能性のある設定変更の追跡が含まれます。定期的なコンプライアンス監査により、業界標準と組織のセキュリティポリシーへの準拠が保証され、設定ドリフトの検出により、セキュリティリスクを引き起こす可能性のある不正な変更を防ぐことができます。

Amazon EKS のセキュリティモニタリングは、規制要件への準拠を確保しながら、最新のセキュリティ脅威から保護するために必要な可視性と制御を提供します。包括的なセキュリティモニタリングを実装することで、組織は強力なセキュリティ体制を維持し、セキュリティインシデントに迅速に対応し、さまざまな規制標準への準拠を実証することができます。

Amazon EKS のモニタリングツール

このセクションでは、Amazon EKS モニタリングツールの 3 つのカテゴリについて説明します。AWS モニタリングサービス、オープンソースまたは独自のソリューション、および特殊なツールです。

AWS サービス

- Amazon CloudWatch: 包括的なモニタリングとログ記録サービス

  CloudWatch は AWS モニタリングソリューションのバックボーンを形成し、Amazon EKS 環境に広範な機能を提供します。詳細なコンテナおよびクラスターメトリクスの Container Insights を提供するため、パフォーマンス、リソース使用率、アプリケーションのヘルスをモニタリングできます。このサービスはログの集約と分析に優れており、コンテナとノード間の一元的なログ記録をサポートしています。CloudWatch は と自然に統合されます AWS のサービス。自動アラーム設定を

提供し、カスタムメトリクスとダッシュボードをサポートしているため、Amazon EKS モニタリングに不可欠なツールです。

- AWS X-Ray: 高度な分散トレースプラットフォーム

  X-Ray は、高度な分散トレース機能を提供することでオブザーバビリティを向上させます。サービスマップの視覚化により、アプリケーションアーキテクチャと依存関係を明確に把握でき、詳細なリクエスト追跡により、サービス全体のパフォーマンスのボトルネックを特定できます。X-Ray は複雑なマイクロサービスアーキテクチャを通じてリクエストを追跡できるため、特に複数の にまたがる分散システムでは、トラブルシューティングや最適化に非常に役立ちます AWS のサービス。

- AWS Distro for OpenTelemetry: 統合オブザーバビリティフレームワーク

  Distro for OpenTelemetry は、クロスプラットフォームをサポートする統合データ収集機能を提供するため、ハイブリッド環境に最適です。このサービスは他の と統合され AWS のサービス、カスタム計測をサポートし、業界標準との互換性を維持しながら包括的なモニタリングソリューションを柔軟に実装できます。

- Amazon Managed Grafana: エンタープライズグレードの視覚化

  Amazon Managed Grafana は、データの可視化と分析のためのフルマネージドサービスを提供します。他の や AWS のサービス組み込みのセキュリティ機能、エンタープライズグレードのスケーラビリティとシームレスに統合できます。このサービスは、クロスアカウントデータソースへのアクセスや との統合などの高度な機能を提供しながら、ダッシュボードの作成と管理を簡素化します AWS IAM アイデンティティセンター。

- Amazon Managed Service for Prometheus: 高可用性、セキュア、マネージドモニタリング

  Amazon Managed Service for Prometheus は、完全マネージド型の Prometheus 互換モニタリングサービスです。自動スケーリング、高可用性、安全なメトリクスの取り込みとクエリを提供します。このサービスは Amazon EKS とシームレスに統合され、Prometheus サーバーを管理するための運用上のオーバーヘッドを排除します。

オープンソースまたは独自のソリューション

前のセクションで説明した AWS ツールは、シームレスな統合サービスとマネージドサービスを提供します。このセクションに記載されているオープンソースツールは、柔軟性と広範なカスタマイズオプションを提供すること AWS のサービス で補完します。各ツールの機能とユースケースを理解することは、特定の要件を満たすモニタリング戦略を設計するのに役立ちます。

- Prometheus: メトリクス収集ツールキット

  Prometheus は、Kubernetes 環境でメトリクスを収集するためのオープンソースソリューションです。時系列データベースと PromQL クエリ言語により、高度なメトリクス分析が可能になります。プラットフォームのサービス検出機能は動的 Kubernetes 環境に自動的に適応し、アラート管理システムは重大な問題を常に把握します。Prometheus には広範な統合オプションが用意されており、包括的なメトリクスモニタリングのための汎用的な選択肢となります。

- Grafana: 高度な視覚化エンジン

  Grafana は、視覚化機能を通じて、複雑なモニタリングデータを実用的なインサイトに変換します。プラットフォームは、複数のソースからのデータを組み合わせたカスタマイズされたダッシュボードを作成し、インフラストラクチャとアプリケーションメトリクスの統合ビューを提供します。さまざまなデータソースとアラート管理機能をサポートしているため、包括的なモニタリングが可能です。Grafana は、リアルタイムデータと履歴データの両方を視覚化するのに役立つため、傾向を特定し、情報に基づいた意思決定を行うことができます。

- Fluent Bit: 統合ログ記録レイヤー

  このログ記録ソリューションは、Kubernetes 環境のログ収集と管理を提供します。ネイティブの Kubernetes 統合により、コンテナとノードからのシームレスなログ収集が保証され、複数の出力先がサポートされるため、ログのストレージと分析に柔軟性がもたらされます。ログ解析やフィルタリングなどの高度な機能により、特定の要件に基づいてログを処理およびルーティングできます。Fluent Bit の軽量性により、コンテナ化された環境に特に適しています。

- Datadog: フルスタックオブザーバビリティ

  Datadog は、ネイティブ Kubernetes サポートを備えた包括的なモニタリング機能を提供します。インフラストラクチャモニタリング、アプリケーションパフォーマンスモニタリング (APM)、ログ管理、リアルタイム分析を提供します。プラットフォームの自動サービス検出と Amazon EKS モニタリング用の広範な統合カタログ、およびその機械学習機能を使用して、異常を検出し、潜在的な問題を予測できます。

- New Relic: アプリケーションパフォーマンスのモニタリング

  New Relic は、アプリケーションのパフォーマンスとインフラストラクチャの状態を可視化します。Kubernetes 統合により、詳細なコンテナインサイト、分散トレース、カスタムダッシュボードが提供されます。プラットフォームは、アプリケーションのパフォーマンスをインフラストラクチャメトリクスに関連付けるため、問題をすばやく特定して解決できます。

- Elastic Stack (ELK スタック): ログ分析と検索

ELK スタックは、Elasticsearch、Logstash、Kibana を組み合わせて、ログ管理と分析機能を提供します。高度な検索機能、視覚化ツール、機械学習機能を提供します。スタックを使用して、Amazon EKS 環境からの大量のログデータを処理できます。

特殊なツール

特定のモニタリング要件、運用規模、組織設定に基づいて、以下のツールを混在させて一致させることができます。重要なのは、管理性とコスト効率を維持しながら、包括的な可視性を提供するモニタリングスタックを作成することです。

- kube-state-metrics (KSM): Kubernetes 状態モニタリング

  このアドオンサービスは Kubernetes API サーバーをリッスンし、オブジェクトの状態に関するメトリクスを生成します。デプロイ、ポッド、その他の Kubernetes リソースの正常性に関するインサイトを提供します。
- Kubernetes メトリクスサーバー: リソースメトリクス

  このメトリクスサーバーは、kubelets からリソースメトリクスを収集し、Kubernetes メトリクス API を通じて公開します。水平ポッドの自動スケーリングと基本的な CPU およびメモリメトリクスを提供します。
- Kubecost: Kubernetes コストモニタリング

  Kubecost などのツールは、EKS クラスターの詳細なコスト分析と最適化の推奨事項を提供します。これらは、さまざまな名前空間、デプロイ、サービスにわたるクラウド支出の理解と最適化に役立ちます。

Amazon EKS モニタリングソリューションの高可用性の実装

Amazon EKS モニタリングの堅牢な高可用性 (HA) 戦略は、Kubernetes 環境を継続的に可視化するために不可欠です。このセクションでは、モニタリングインフラストラクチャのさまざまな側面に HA を実装するための包括的なアプローチについて説明します。

アーキテクチャの冗長性とスケーラビリティ

高可用性モニタリングシステムの構築は、適切なアーキテクチャ設計から始まります。モニタリングコンポーネントは、ゾーンの障害から保護するために、複数の AWS アベイラビリティーゾーンに分散する必要があります。これには、Prometheus サーバー、ログコレクター、アラートマネー

ジャーなどの重要なモニタリングコンポーネントの水平スケーリングの実装が含まれます。Amazon Managed Service for Prometheus や Amazon Managed Grafana などのマネージド AWS サービスを使用すると、高可用性を確保しながら運用オーバーヘッドを削減できます。自動フェイルオーバーメカニズムを設定して、コンポーネントの障害発生時のサービス継続性を維持し、ヘルスチェックと自動復旧手順を実施します。

回復力のあるデータストレージ戦略

データストレージの耐障害性は、モニタリングシステムの信頼性を維持するために不可欠です。分散ストレージソリューションを実装することで、個々のストレージノードに障害が発生しても、メトリクスデータとログにアクセスし続けることができます。これには、複数のアベイラビリティーゾーン間で適切なデータレプリケーションを設定し、冗長性のために異なるストレージバックエンドを使用することが含まれます。履歴データの定期的なバックアップ手順を確立し、さまざまな障害シナリオの復旧プロセスを文書化します。Prometheus などの時系列データベースの場合、リモートストレージソリューションを実装すると、ストレージの懸念をデータ収集から分離し、システム全体の信頼性を向上させることができます。

冗長アラート管理

アラート管理では、HA セットアップに特別な注意が必要です。冗長アラートマネージャーをデプロイすると、システム障害発生時でも重要な通知が意図した受信者に到達します。E メール、SMS、Slack、PagerDuty などの複数の通知チャネルを設定して、代替通信パスを提供します。アラート重複排除メカニズムを使用して部分的なシステム障害時のアラートストームを防ぎ、フォールバック通知方法を使用して重要なアラートを見逃さないようにします。アラート相関を実装すると、フェイルオーバーシナリオ中にコンテキストを維持し、冗長システムからの通知の重複を防ぐことができます。

ロードバランシングとサービス検出

安定したモニタリングサービスを維持するには、適切な負荷分散が不可欠です。 AWS Application Load Balancer は受信モニタリングトラフィックを複数のエンドポイントに分散し、ヘルスチェックはトラフィックが正常なインスタンスにのみルーティングされるようにします。サービス検出メカニズムは、モニタリングコンポーネントが新しいノードやサービスの追加など、環境の変化に自動的に適応するのに役立ちます。DaemonSets を使用してモニタリングエージェントをすべてのノードに一貫してデプロイし、クラスターのスケールに応じて包括的なカバレッジを確保します。

HA に関するその他の考慮事項

ネットワークの耐障害性:

- 冗長ネットワークパスを実装します。
- アベイラビリティーゾーン全体で適切なサブネット設計を設定します。
- バックアップルートAWS Direct Connectで を使用します。
- 適切なセキュリティグループとネットワークアクセスコントロールリスト (ネットワーク ACLs) を設定します。

モニターのモニタリング:

- セカンダリモニタリングシステムをデプロイします。
- クロスリージョンモニタリングを実装します。
- 応答しないシステムのアラートを設定します。
- フェイルオーバー手順を定期的にテストします。

キャパシティプランニング:

- リソースの使用状況の傾向をモニタリングします。
- 予測スケーリングを実装します。
- パフォーマンスを定期的にテストします。

データ管理:

- データ保持ポリシーを実装します。
- メトリクス集約を設定します。
- データライフサイクル管理の計画を立てます。
- ストレージを定期的に最適化します。

復旧手順:

- 復旧プロセスを文書化します。
- ディザスタリカバリを定期的にテストします。
- 可能な場合は、自動復旧を実装します。
- 明確なエスカレーションパスを特定して実装します。

これらの高可用性プラクティスを実装することで、Amazon EKS モニタリングインフラストラクチャの信頼性と回復力を維持し、さまざまな障害シナリオでも Kubernetes 環境を継続的に可視化できます。これらの HA 設定の定期的なテストと更新により、環境の進化に合わせて有効のままになります。

Amazon EKS でのモニタリングのベストプラクティス

戦略的実装アプローチ

Amazon EKS モニタリング戦略を成功させるには、十分に計画された段階的な実装アプローチから始めます。

- まず、ビジネスオペレーションとアプリケーションの信頼性に直接影響する重要なメトリクスを特定してモニタリングします。この基盤には、重要なインフラストラクチャメトリクス、主要なアプリケーションパフォーマンス指標、重要なセキュリティメトリクスを含める必要があります。運用上のニーズと学んだ教訓に基づいてモニタリングカバレッジを段階的に拡大し、各追加が有意義な価値をもたらすことを確認します。
- Terraform や などの infrastructure as code (IaC) ツールを使用して、一貫性と再現性を確保 CloudFormation することで、自動デプロイプロセスを実装します。
- 信頼性と精度を維持するために、モニタリングシステムをテストおよび検証します。
- 変化するビジネスニーズに合わせて、モニタリングパラメータを継続的に改善します。

効果的なデータ管理

効率的で費用対効果の高いモニタリングソリューションを維持するには、適切なデータ管理が不可欠です。

- 履歴分析のニーズとストレージコストのバランスを取る明確なデータ保持ポリシーを実装します。
- さまざまなメトリクスタイプに適切なサンプリングレートを設定します。重要なメトリクスでは頻度が高くなり、重要度の低いメトリクスでは頻度が低くなります。
- メトリクス集約を使用して、特に長期的な傾向分析のために、有意義なインサイトを維持しながらデータ量を削減します。
- 一元的なログ記録システム (CloudWatch Logs など) に体系的なログ保持とアーカイブ手順を実装して、ストレージコストを管理し、重要なデータへのアクセスを引き続き維持します。

> **Note**
>
> コンテナレベルのログローテーションは、Amazon EKS バージョン 1.21 以降の kubelet によって自動的に処理されます。

- アクセス速度とコスト効率の両方を最適化するために、ログストレージにhot-warm-coldアーキテクチャを実装することを検討してください。

アラートの設定と管理

アラート設定では、アラートの疲労を引き起こすことなく有効性を維持するために慎重に検討する必要があります。

- サービスレベル目標 (SLOs) と過去のパフォーマンスパターンに基づいて、明確で実用的なしきい値を定義します。
- 即時対応が必要な重大な問題と緊急性の低い問題を明確に区別する階層型アラート重要度システムを実装します。
- アラートが、迅速な問題解決に役立つ十分なコンテキストと実用的な情報を提供していることを確認します。
- さまざまなアラート重要度の所有権と応答時間を定義して、明確なエスカレーション手順を確立します。
- アラート設定を定期的に見直して絞り込み、関連性と有効性を維持します。

リソースの最適化

費用対効果の高い運用を維持するには、リソース使用率の継続的なモニタリングが不可欠です。

- ノード、ポッド、永続ボリュームなど、すべてのクラスターコンポーネントに包括的なリソースモニタリングを実装します。
- 実際の使用パターンとパフォーマンス要件に基づいて自動スケーリングを設定し、パフォーマンスを維持しながら効率的なリソース使用率を確保します。
- コスト配分タグを使用して、さまざまなチーム、アプリケーション、環境のリソース消費量を追跡します。
- リソース効率メトリクスを定期的に分析して、最適化の機会を特定し、改善を実装します。
- クラウド支出を追跡して最適化するためのコスト管理ツールの実装を検討してください。

セキュリティ

セキュリティ上の考慮事項は、モニタリング戦略に不可欠です。

- すべてのモニタリングコンポーネントの最小特権アクセス原則を実装して、ユーザーとサービスに必要なアクセス許可のみを持つようにします。
- 包括的な監査ログ記録を有効にして、モニタリングシステムへのすべてのアクセスと変更を追跡します。
- 潜在的な脆弱性を特定するために、モニタリング設定とアクセスパターンを定期的にセキュリティレビューします。
- 転送中と保管時の両方の機密データモニタリングデータの暗号化を実装します。
- セキュリティモニタリングを既存のセキュリティ情報およびイベント管理 (SIEM) システムと統合して、包括的なセキュリティ可視性を実現します。

Amazon EKS での高度なモニタリングに関する考慮事項

パフォーマンスの最適化:

- メトリクス収集間隔を最適化します。
- 効率的なクエリパターンを設定します。
- メトリクスの事前集約を実装します。
- 適切なストレージソリューションを使用します。

コンプライアンスとガバナンス:

- 監査証跡を維持します。
- コンプライアンスモニタリングを実装します。
- 定期的なコンプライアンスレポートを提供します。
- モニタリング手順を文書化します。

ディザスタリカバリ:

- モニタリング設定を定期的にバックアップします。
- 復旧手順を文書化します。

- 復旧プロセスをテストします。

継続的な改善:

- レビューセッションを定期的にモニタリングします。
- パフォーマンスサイクルを最適化します。
- インシデントに基づいてモニタリングを更新します。
- ユーザーフィードバックを組み込みます。

これらのベストプラクティスは、Amazon EKS 環境の効果的なモニタリングソリューションを実装および維持するためのフレームワークを提供します。これらのプラクティスを定期的に見直して更新し、組織のニーズと業界標準に合わせるようにします。モニタリングは 1 回限りのセットアップではなく、定期的な注意と改良が必要な継続的なプロセスです。

Amazon EKS でのトレース

トレースは、Amazon EKS でのアプリケーションオブザーバビリティの重要なコンポーネントです。トレースは、EKS クラスターにデプロイされたさまざまなマイクロサービスを通過するリクエストのパスを収集、処理、視覚化することで、リクエストフローとサービスインタラクションを詳細に可視化します。この機能は、Amazon EKS 環境でシステムの動作を理解し、ボトルネックを特定し、問題を効果的にトラブルシューティングするのに役立ちます。効果的なトレースにより、リクエストフローend-to-end可視化できるため、分散システムのデバッグの複雑さがなくなります。これにより、サービス境界を越えてトランザクションを追跡し、Amazon EKS ワークロード内のパフォーマンスの問題や障害を特定できます。

Amazon EKS の全体的なトレース実装により、システム動作を理解し、パフォーマンスを最適化し、コンテナ化されたアプリケーションの信頼性を維持できます。最終的に、トレースの機能により、Amazon EKS 環境における運用の可視性とシステムの保守性が向上します。

AWS X-Ray は、アプリケーションに関するデータのトレースに重要な役割を果たします。トレースには、以下を含むサービスインタラクションのさまざまな側面のモニタリングが含まれます。

- リクエストパスと依存関係は、分散システムの動作に関する重要なインサイトを提供します。さまざまなマイクロサービスやコンポーネントを通過するリクエストの完全なジャーニーを追跡します。サービスの依存関係のマッピングは、通信パターンを理解し、アプリケーションアーキテクチャの重要なパスを特定するのに役立ちます。実装の詳細については、X-Ray ドキュメントの AWS X-Ray 「サービストレースマップの使用」を参照してください。
- サービスのレイテンシーとボトルネックは、最適なシステムパフォーマンスを維持するために不可欠なメトリクスです。サービス間の応答時間を測定および分析することで、パフォーマンスの問題を効果的に特定できます。このデータにより、リクエストチェーンで遅延を引き起こしている特定のサービスまたはオペレーションを特定し、ターゲットを絞った最適化作業を実現できます。レイテンシー分析の詳細については、X-Ray ドキュメントの「分析コンソールの操作」を参照してください。
- エラー伝達パターンは、システムの信頼性と耐障害性を理解するのに役立ちます。サービス間でエラーパスを追跡することで、障害がシステムをどのようにカスケードするかを理解することで、アプリケーションをより適切に設計できます。この可視性により、エラーの根本原因と依存サービスへの影響を特定し、より回復力のあるシステムにつながります。実装の詳細については、X-Ray ドキュメントの「トレース」を参照してください。
- サービス全体のリソース使用率は、システム効率とコスト最適化に関するインサイトを提供します。トレースデータと相関する CPU、メモリ、ネットワーク使用状況パターンをモニタリングし

て、リソースの需要を把握できます。このデータは、リソース消費の傾向を分析して、EKS クラスター全体のサービスパフォーマンスとコストを最適化するのに役立ちます。モニタリングのセットアップについては、「Amazon EKS ドキュメント」の「クラスターのパフォーマンスをモニタリングし、ログを表示する」を参照してください。

- エンドユーザートランザクションフローは、ユーザーエクスペリエンスを理解して改善するために不可欠です。フロントエンドからバックエンドサービスへの完全なユーザーインタラクションを追跡することで、最適なアプリケーションパフォーマンスを確保できます。重要なユーザージャーニーのend-to-endの応答時間を測定および最適化できるため、顧客満足度に直接影響します。エンドユーザーモニタリングを実装するには、プログラミング言語に AWS X-Ray SDK を使用します。
- API ゲートウェイインタラクションは、アプリケーションのパフォーマンスとセキュリティの最前線を形成します。API エントリポイントでリクエストパターンとパフォーマンスをモニタリングして、最適なサービス配信を確保できます。この可視性により、認証、認可、リクエストフローへの影響のレート制限を追跡し、セキュリティ要件とパフォーマンス要件の両方を維持できます。X-Raydocumentation を使用した Amazon API Gateway での API トレースについて説明します。

Amazon EKS での効果的なトレースは、スパンとトレースの収集にとどまりません。これには、オブザーバビリティのニーズとシステムパフォーマンスのバランスを取る、適切に構造化された戦略が必要です。この戦略では、以下に焦点を当てる必要があります。

- 適切なサンプリングレートの実装: トラフィックパターンとビジネスの優先順位に基づいてサンプリングルールを設定し、重要なトランザクションの可視性を維持しながらコストを最適化します。詳細については、X-Ray ドキュメントの「サンプリングルールの設定」を参照してください。
- トレースする重要なパスとサービスの定義: 最適なパフォーマンスモニタリングを確保するために詳細なトレースを必要とする重要なサービスとユーザージャーニーを特定して優先順位を付けます。詳細については、Amazon EKS ドキュメントの「ADOT Operator を使用してメトリクスとトレースデータを送信する」を参照してください。
- 適切なデータ保持ポリシーの確立: データライフサイクル管理ルールを設定して、オブザーバビリティのニーズとストレージコストおよびコンプライアンス要件のバランスを取ります。CloudWatch 保持ポリシーを表示するには、CloudWatch Logs ドキュメント」の「ロググループとログストリームの使用」を参照してください。
- 効果的な視覚化および分析ツールの設定: Analytics AWS X-Ray コンソールや Amazon Managed Grafana などの視覚化ツールをデプロイして設定し、トレースデータを効果的に分析します。詳細については、X-Ray ドキュメントの「分析コンソールの操作」を参照してください。

このセクションの内容:

Amazon EKS のトレースツール

Amazon EKS は、分散トレースを実装するためのいくつかの AWS およびサードパーティーのオプションをサポートしています。

AWS のサービス

- AWS X-Ray: 高度な分散トレースプラットフォーム

  X-Ray は、end-to-end AWS のサービス のトレース機能を提供するフルマネージド型です。Amazon EKS で実行されるアプリケーションの詳細なサービスマップと分析を自動的に計測 AWS のサービス して提供します。X-Ray は AWS のサービス、Amazon CloudWatch を含む他のと統合されており、トレースと AWS のサービス 呼び出しの自動相関を提供します。

- AWS Distro for OpenTelemetry: Unified Observability Framework

  Distro for OpenTelemetry は、クラウドネイティブアプリケーション向けの OpenTelemetry の安全で本番環境に対応した、 AWSでサポートされているディストリビューションです。ネイティブ AWS のサービス 統合を維持しながらベンダーに依存しない計測機能を提供するため、ハイブリッドクラウド環境に最適です。Distro for OpenTelemetry は、複数のオブザーバビリティバックエンドをサポートし、 AWS モニタリングサービスとのシームレスな統合を提供します。

オープンソースソリューション

- OpenTelemetry: オープンソースのオブザーバビリティフレームワーク

  OpenTelemetry は、複数のプログラミング言語をサポートする包括的な計測ライブラリを備えた標準化されたオブザーバビリティフレームワークを提供します。柔軟なバックエンドオプションとベンダーに依存しないアプローチにより、さまざまな環境間で整合性を必要とするワークロードに最適です。フレームワークの広範なエコシステムにより、さまざまなモニタリングソリューションとの幅広い互換性が保証されます。

- Jaeger: オープンソースの分散トレースプラットフォーム

Jaeger は、リアルタイムの分散コンテキスト伝達による包括的なトレース機能を提供します。詳細なサービス依存関係の視覚化を通じて、根本原因の分析とパフォーマンスの最適化を提供します。Jaeger のアーキテクチャは、高いスケーラビリティを実現するように設計されており、さまざまなストレージバックエンドをサポートしているため、大規模な Amazon EKS デプロイに適しています。EKS 用 ViewJaeger のセットアップ

- Grafana Tempo: 分散トレース

  Tempo は、大規模なトレースストレージと Prometheus メトリクスとのシームレスな統合を提供する Grafana Labs ソリューションです。費用対効果の高いトレース保持モデルと Grafana とのネイティブ統合により、視覚化にすでに Grafana を使用している組織に適しています。Tempo のアーキテクチャは、Amazon EKS などのクラウドネイティブ環境専用に設計されています。

Amazon EKS でのトレースのベストプラクティス

このセクションでは、Amazon EKS の Kubernetes ベースのアプリケーションのオブザーバビリティとトラブルシューティングを強化する効果的なトレースシステムを作成するためのベストプラクティスと手法の包括的なリストを提供します。

- 戦略的サンプリング: アプリケーションのトラフィックパターンと使用しているサービスの重要性に基づいて、さまざまなサンプリングレートを設定します。重要パスのサンプリングレートを高めながら、重要ではない大量のルートのサンプリングを減らしてコストを最適化します。ガイダンスについては、 AWS X-Ray ドキュメントの「サンプリングルールの設定」を参照してください。
- 計測設定: X-Ray SDK や AWS Distro for OpenTelemetry コレクターなどの自動計測ツールを使用して、手動計測の労力を最小限に抑えます。トレースの相関性を向上させるために、サービス間で一貫した命名規則とコンテキスト伝達を維持します。詳細については、 Distro for OpenTelemetry コレクターのドキュメントを参照してください。
- データ管理: 適切な保持期間と圧縮戦略を実装して、ストレージコストとオブザーバビリティのニーズのバランスを取ります。機密性の高いトレースデータを保護するために、明確なデータプライバシーコントロールとバックアップ手順を確立します。詳細については、CloudWatch Logs ドキュメントの「CloudWatch Logs での口グデータ保持の変更」を参照してください。 CloudWatch
- パフォーマンスの最適化: トレースオーバーヘッドをモニタリングおよび最適化して、アプリケーションのパフォーマンスへの影響を最小限に抑えます。効率的なバッファリングと非同期処理を使用して、レイテンシーへの影響を軽減します。詳細については、X-Ray ドキュメントの AWS X-Ray 「デーモンの設定」を参照してください。

- セキュリティコントロール: IAM ロールとポリシーを使用して、適切なアクセスコントロールとデータ保護対策を実装します。定期的なセキュリティ監査とコンプライアンスレビューは、トレースデータの安全性を確保するのに役立ちます。詳細については、X-Ray ドキュメントの「 のセキュリティ AWS X-Ray」を参照してください。
- モニタリングとアラート: トレースコレクションの状態に関する包括的なモニタリングを設定し、コレクションの問題に関するアラートを設定します。サンプリングレートとシステムパフォーマンスメトリクスを追跡して、最適なオペレーションを確保します。詳細については、CloudWatch ドキュメントの「Container Insights」を参照してください。
- 高可用性: 複数のアベイラビリティーゾーンに冗長コレクターをデプロイし、適切なフェイルオーバーメカニズムを設定します。高可用性セットアップの定期的なテストにより、信頼性の高いトレース収集が保証されます。詳細については、Amazon Managed Service for Prometheus AWS ドキュメントの Distro for OpenTelemetry をコレクターとして使用するを参照してください。

これらのベストプラクティスに従うことで、Amazon EKS 環境用の堅牢で効率的で効果的なトレースシステムを作成できます。これにより、Kubernetes ベースのアプリケーションの包括的なオブザーバビリティ、効率的なトラブルシューティング、最適なパフォーマンスを確保できます。

Amazon EKS でのアラート

アラートは、Amazon EKS で実行されるアプリケーションを管理および維持する上で重要なコンポーネントです。これは、サービスの可用性やユーザーエクスペリエンスに影響を与える可能性のある重大な問題にエスカレートする前に、潜在的な問題、異常、パフォーマンスの低下をオペレーターや開発者に通知する早期警告システムとして機能します。アラートには、Kubernetes クラスターのさまざまな側面のモニタリングが含まれます。

- インフラストラクチャのヘルス
- アプリケーションのパフォーマンス
- コンテナのメトリクス
- カスタムビジネスメトリクス

Amazon EKS での効果的なアラートは、単なる通知の設定にとどまりません。これには、タイムリーな情報の必要性とアラート疲労の可能性のバランスを取るwell-thought-outされた戦略が必要です。この戦略では、次のことを行う必要があります。

- 意味のあるしきい値と条件を定義します。
- 重要度と影響に基づいてアラートに優先順位を付けます。
- 適切なルーティングとエスカレーションの手順を実装します。
- インシデント管理およびコミュニケーションツールと統合します。

このセクションの内容:

- Amazon EKS のアラートツール
- Amazon EKS でのアラートのベストプラクティス

Amazon EKS のアラートツール

Amazon EKS は、アラートを実装するためのいくつかの AWS およびサードパーティーのオプションをサポートしています。Amazon EKS アラート用のツールを選択するときは、統合機能、スケーラビリティ、使いやすさ、コスト、モニタリング要件とアラート要件に合った特定の機能などの要因を考慮してください。多くの組織は、これらのツールを組み合わせて、Amazon EKS 環境の包括的なモニタリングおよびアラートソリューションを作成します。

- Amazon CloudWatch: モニタリングとオブザーバビリティ AWS のサービス 用

  CloudWatch は、EKS クラスターのメトリクス、ログ、アラームを提供し、他の と適切に統合します AWS のサービス。
- Prometheus: Kubernetes 用のオープンソースのモニタリングおよびアラートツール

  Prometheus は、アラート条件を定義するための強力なクエリ言語 (PromQL) を提供します。
- アラートマネージャー: アラートを処理するための Prometheus へのコンパニオン

  アラートマネージャーは、アラートの重複排除、グループ化、ルーティングを行います。E メール、Slack、PagerDuty など、さまざまな通知チャネルをサポートしています。
- Grafana: モニタリングとオブザーバビリティのためのオープンソースプラットフォーム

  Grafana は視覚化とアラート機能を提供します。Prometheus や CloudWatch など、さまざまなデータソースと統合できます。
- Elastic Stack (ELK スタック): Elasticsearch、Logstash、Kibana の組み合わせ

  このツールは、ログの集約、分析、アラートに役立ちます。Elastic のオブザーバビリティ機能を使用して拡張できます。
- サードパーティーソリューション

  Datadog、New Relic、Sysdig、Dynatrace、Zabbix、Nagios、Splunk、IBM Instana、AppDynamics など、市場には多くのツールがあります。

Amazon EKS でのアラートのベストプラクティス

このセクションでは、Amazon EKS の Kubernetes ベースのアプリケーションの信頼性とパフォーマンスを強化する堅牢なアラートシステムを作成するためのベストプラクティスについて説明します。

明確なアラートしきい値を定義します。

- 履歴データとビジネス要件に基づいて意味のあるしきい値を設定します。
- 必要に応じて動的しきい値を使用して、さまざまなワークロードを考慮します。

アラートの優先順位付けを実装します。

- 重要度 (重大、高、中、低など) でアラートを分類します。
- アラートの優先順位をビジネスへの影響に合わせます。

アラートの疲労を避ける:

- 冗長アラートや低値アラートを排除してノイズを減らします。
- アラートを関連付けて、関連する問題をグループ化します。

マルチステージアラートを使用する:

- クリティカルレベルに達する前に警告しきい値を実装します。
- アラート重要度ごとに異なる通知チャネルを使用します。

適切なアラートルーティングを実装します。

- アラートが適切なチームまたは個人に送信されていることを確認します。
- オンコールスケジュールとローテーションを終日、毎日適用します。

Kubernetes ネイティブメトリクスを活用する:

- コア Kubernetes コンポーネント (ノード、ポッド、サービス) をモニタリングします。
- 追加の Kubernetes オブジェクトメトリクスには kube-state-metrics (KSM) を使用します。

インフラストラクチャとアプリケーションの両方をモニタリングします。

- クラスターの状態、ノードのステータス、リソース使用率に関するアラートを設定します。
- エラー率やレイテンシーなどのアプリケーション固有のアラートを実装します。

Prometheus とアラートマネージャーを使用する:

- メトリクス収集には Prometheus を使用し、アラート条件を定義するには PromQL を使用します。
- アラートのルーティングと重複排除には、アラートマネージャーを使用します。

Amazon CloudWatch との統合:

- Amazon EKS 固有のメトリクスには CloudWatch Container Insights を使用します。
- 重要な AWS リソースメトリクスの CloudWatch アラームを設定します。

コンテキスト豊富なアラートを実装します。

- クラスター名、名前空間、ポッドの詳細などの関連情報をアラートメッセージに含めます。
- アラート内の関連するダッシュボードまたはランブックへのリンクを提供します。

異常検出を使用します。

- 複雑なパターンに対して機械学習ベースの異常検出を実装します。
- CloudWatch 異常検出やサードパーティーツールなどのサービスを使用します。

アラート抑制とサイレンシングを実装します。

- 既知の問題の一時的な抑制を許可します。
- メンテナンスウィンドウを実装して、計画されたダウンタイム中のノイズを減らします。

アラートパフォーマンスをモニタリングします。

- アラートの頻度、解決時間、誤検出率などのメトリクスを追跡します。
- これらのメトリクスに基づいて、アラートルールを定期的に見直して絞り込みます。

エスカレーション手順を実装します。

- 未解決のアラートの明確なエスカレーションパスを定義します。
- 自動エスカレーションには PagerDuty や Opsgenie などのツールを使用します。

アラートシステムを定期的にテストします。

- アラートパイプラインの定期的なテストを実施します。
- ディザスタリカバリドリルにアラートテストを含めます。

アラートの一貫性を保つには、 テンプレートを使用します。

- 一般的なシナリオ用に標準化されたアラートテンプレートを作成します。
- すべてのアラートで一貫したフォーマットと情報を確保します。

レート制限を実装します。

- 頻繁にトリガーされるアラートにレート制限を実装することで、アラートストームを防止します。

カスタムメトリクスを使用する:

- アプリケーション固有のモニタリング用のカスタムメトリクスを実装します。
- これらのメトリクスに基づく自動スケーリングには、Kubernetes カスタムメトリクス API を使用します。

ログ記録統合を実装します。

- アラートを関連するログと関連付けて、トラブルシューティングを高速化します。
- Grafana Loki や ELK スタックなどのツールをアラートシステムと組み合わせて使用します。

コストアラートを検討します。

- リソースの使用量やコストが予期せず急増した場合のアラートを設定します。
- AWS Budgets またはサードパーティーのコスト管理ツールを使用します。

分散トレースを使用する:

- Jaeger や などの分散トレースツールを統合しますAWS X-Ray。
- 異常なトレースパターンまたはレイテンシーのアラートを設定します。

ドキュメントアラートランブック:

- アラートタイプごとに明確で実用的なランブックを作成します。
- トラブルシューティングステップとエスカレーション手順をランブックに含めます。

これらのベストプラクティスに従うことで、Amazon EKS 環境に堅牢で効率的で効果的なアラートシステムを作成できます。これにより、Kubernetes ベースのアプリケーションの高可用性、迅速な問題解決、最適なパフォーマンスを確保できます。

次のステップ

このガイドでは、Amazon EKS 環境で堅牢なオブザーバビリティを実装するための包括的なフレームワークを提供し、メトリクス収集、ログ記録インフラストラクチャ、分散トレース、コスト最適化に焦点を当てました。これらのコアコンポーネントを理解して適用することで、アプリケーションとインフラストラクチャの動作に関する深いインサイトを提供する、非常にオブザーバビリティが高く、保守可能で費用対効果の高いコンテナ環境を構築できます。Amazon CloudWatch Container Insights や AWS のサービス などの と Prometheus や OpenTelemetry などのAWS X-Rayオープンソースソリューションの統合により、コンテナ化されたアプリケーションをモニタリングおよびトラブルシューティングするための強力な基盤が作成されます。

実装の成功は、コアメトリクス収集から始まり、徐々に包括的なログ記録と分散トレース機能に拡張する段階的なアプローチに依存しています。まず、現在のモニタリング能力を評価し、ギャップを特定し、運用要件とチームの専門知識に合った適切なツールの組み合わせを選択することをお勧めします。この体系的なアプローチにより、オブザーバビリティスタックの各コンポーネントが適切に実装および統合され、チームはこれらのツールを効果的に使用するために必要なスキルとプロセスを開発します。

Amazon EKS オブザーバビリティの長期的な持続可能性は、コスト、リソース、プロセスの定期的な最適化によって異なります。データ保持ポリシー、サンプリングレート、リソース割り当てなどのオブザーバビリティインフラストラクチャを継続的に見直して調整し、包括的なモニタリングと運用効率の適切なバランスを維持する必要があります。この改善への反復的なアプローチと継続的なチームトレーニングとドキュメントの更新を組み合わせることで、組織はビジネスの成長をサポートし、進化するアプリケーションアーキテクチャに適応しながら、効果的なオブザーバビリティを維持できます。

リソース

AWS ドキュメント

- Amazon EKS ベストプラクティスガイド
- Amazon CloudWatch Container Insights
- Amazon Managed Service for Prometheus
- Amazon マネージド Grafana
- AWS Distro for OpenTelemetry および AWS X-Ray
- Amazon OpenSearch Service

AWS ブログ投稿

- Amazon EKS が Kubernetes コントロールプレーンのオブザーバビリティを強化
- Amazon Managed Service for Prometheus マネージドスクレイパーによる Amazon EKS でのメトリクス収集の自動化
- CloudWatch Container Insights を使用して Amazon EKS クラスターのモニタリングを自動化する
- Amazon EKS のマネージドモニタリングソリューションによるオブザーバビリティの強化

その他のリソース

- OpenTelemetry ドキュメント
- Prometheus ドキュメント
- Fluent Bit ドキュメント
- Kubernetes ドキュメントのモニタリング、ログ記録、デバッグ

ドキュメント履歴

以下の表は、本ガイドの重要な変更点について説明したものです。今後の更新に関する通知を受け取る場合は、RSS フィードをサブスクライブできます。

変更	説明	日付
更新	Amazon EKS の「ログ記録」の章を更新しました。	2026 年 3 月 17 日
初版発行	—	2025 年 4 月 10 日

AWS 規範ガイダンスの用語集

以下は、 AWS 規範ガイダンスによって提供される戦略、ガイド、パターンで一般的に使用される用語です。エントリを提案するには、用語集の最後のフィードバックの提供リンクを使用します。

数字

7 Rs

アプリケーションをクラウドに移行するための 7 つの一般的な移行戦略。これらの戦略は、ガートナーが 2011 年に特定した 5 Rs に基づいて構築され、以下で構成されています。

- リファクタリング/アーキテクチャの再設計 — クラウドネイティブ特徴を最大限に活用して、俊敏性、パフォーマンス、スケーラビリティを向上させ、アプリケーションを移動させ、アーキテクチャを変更します。これには、通常、オペレーティングシステムとデータベースの移植が含まれます。例: オンプレミスの Oracle データベースを Amazon Aurora PostgreSQL 互換エディションに移行する。
- リプラットフォーム (リフトアンドリシェイプ) — アプリケーションをクラウドに移行し、クラウド機能を活用するための最適化レベルを導入します。例: お客様のオンプレミスの Oracle データベースを AWS クラウドの Oracle 用の Amazon Relational Database Service (Amazon RDS) に移行する。
- 再購入 (ドロップアンドショップ) — 通常、従来のライセンスから SaaS モデルに移行して、別の製品に切り替えます。例: 顧客関係管理 (CRM) システムを Salesforce.com に移行する。
- リホスト (リフトアンドシフト) — クラウド機能を活用するための変更を加えずに、アプリケーションをクラウドに移行します。例: お客様のオンプレミスの Oracle データベースを AWS クラウドの EC2 インスタンス上の Oracle に移行する。
- 再配置 (ハイパーバイザーレベルのリフトアンドシフト) – 新しいハードウェアを購入したり、アプリケーションを書き換えたり、既存の運用を変更したりすることなく、インフラストラクチャをクラウドに移行できます。オンプレミスプラットフォームから同じプラットフォームのクラウドサービスにサーバーを移行します。例： Microsoft Hyper-Vアプリケーションを に移行します AWS。
- 保持 (再アクセス) — アプリケーションをお客様のソース環境で保持します。これには、主要なリファクタリングを必要とするアプリケーションや、お客様がその作業を後日まで延期したいアプリケーション、およびそれらを移行するためのビジネス上の正当性がないため、お客様が保持するレガシーアプリケーションなどがあります。

- 廃止 — お客様のソース環境で不要になったアプリケーションを停止または削除します。

A

A2A (Agent-to-Agent)

タスクの委任と状態転送をサポートするagent-to-agentコラボレーション用のステートフルプロトコル。

ABAC

「属性ベースのアクセス制御」をご覧ください。

抽象化されたサービス

「マネージドユーザー」をご覧ください。

ACID

「原子性、一貫性、分離性、耐久性 (ACID)」をご覧ください。

アクティブ/アクティブ移行

(双方向レプリケーションツールまたは二重書き込み操作を使用して) ソースデータベースとターゲットデータベースを同期させ、移行中に両方のデータベースが接続アプリケーションからのトランザクションを処理するデータベース移行方法。この方法では、1 回限りのカットオーバーの必要がなく、管理された小規模なバッチで移行できます。アクティブ/パッシブ移行よりも柔軟な方法ですが、さらに多くの作業が必要となります。

アクティブ/パッシブ移行

ソースデータベースとターゲットデータベースを同期させながら、データがターゲットデータベースにレプリケートされている間、接続しているアプリケーションからのトランザクションをソースデータベースのみで処理するデータベース移行方法。移行中、ターゲットデータベースはトランザクションを受け付けません。

[エージェント]

目標を達成するためのツールを使用して、自律的に推論、計画、アクションを実行できる AI システム。

エージェントオペレーション

AI エージェントを本番環境で大規模に構築、テスト、デプロイ、実行するための運用プラクティス。

集計関数

複数行に処理を行い、グループ全体を対象に単一の戻り値を計算する SQL 関数。集計関数の例としては、SUM や MAX などがあります。

AI

「人工知能」をご覧ください。

AIOps

「AI オペレーション」をご覧ください。

匿名化

データセット内の個人情報を完全に削除するプロセス。匿名化は個人のプライバシー保護に役立ちます。匿名化されたデータは、もはや個人データとは見なされません。

アンチパターン

繰り返し起こる問題に対して頻繁に用いられる解決策で、その解決策が逆効果であったり、効果がなかったり、代替案よりも効果が低かったりするもの。

アプリケーション制御

マルウェアからシステムを保護するために、承認されたアプリケーションのみを使用できるようにするセキュリティアプローチ。

アプリケーションポートフォリオ

アプリケーションの構築と維持にかかるコスト、およびそのビジネス価値を含む、組織が使用する各アプリケーションに関する詳細情報の集まり。この情報は、ポートフォリオの検出と分析プロセス の重要な要素であり、移行、モダナイズ、最適化するアプリケーションを特定し、優先順位を付けるのに役立ちます。

人工知能 (AI)

コンピューティングテクノロジーを使用し、学習、問題の解決、パターンの認識など、通常は人間に関連づけられる認知機能の実行に特化したコンピュータサイエンスの分野。詳細については、「人工知能 (AI) とは何ですか?」をご覧ください。

AI オペレーション (AIOps)

機械学習技術を使用して運用上の問題を解決し、運用上のインシデントと人の介入を減らし、サービス品質を向上させるプロセス。 AWS 移行戦略での AIOps の使用方法については、オペレーション統合ガイド を参照してください。

非対称暗号化

暗号化用のパブリックキーと復号用のプライベートキーから成る 1 組のキーを使用した、暗号化のアルゴリズム。パブリックキーは復号には使用されないため共有しても問題ありませんが、プライベートキーの利用は厳しく制限する必要があります。

原子性、一貫性、分離性、耐久性 (ACID)

エラー、停電、その他の問題が発生した場合でも、データベースのデータ有効性と運用上の信頼性を保証する一連のソフトウェアプロパティ。

属性ベースのアクセス制御 (ABAC)

部署、役職、チーム名など、ユーザーの属性に基づいてアクセス許可をきめ細かく設定する方法。詳細については、 AWS Identity and Access Management (IAM) ドキュメントの「 の ABAC AWS」を参照してください。

信頼できるデータソース

最も信頼性のある情報源とされるデータのプライマリーバージョンを保存する場所。匿名化、編集、仮名化など、データを処理または変更する目的で、信頼できるデータソースから他の場所にデータをコピーすることができます。

アベイラビリティーゾーン (AZ)

他のアベイラビリティーゾーンの障害から AWS リージョン 隔離され、同じリージョン内の他のアベイラビリティーゾーンへの低コストで低レイテンシーのネットワーク接続を提供する 内の別の場所。

AWS クラウド導入フレームワーク (AWS CAF)

組織がクラウドへの移行を成功させるための効率的で効果的な計画を立て AWS るための、 のガイドラインとベストプラクティスのフレームワークです。 AWS CAF は、ビジネス、人材、ガバナンス、プラットフォーム、セキュリティ、運用という 6 つの重点分野にガイダンスを整理しています。ビジネス、人材、ガバナンスの観点では、ビジネススキルとプロセスに重点を置き、プラットフォーム、セキュリティ、オペレーションの視点は技術的なスキルとプロセスに焦点を当てています。例えば、人材の観点では、人事 (HR)、人材派遣機能、および人材管理を扱うステークホルダーを対象としています。この観点から、 AWS CAF は、クラウド導入を成功させるための組織の準備に役立つ人材開発、トレーニング、コミュニケーションに関するガイダンスを提供します。詳細については、AWS CAF ウェブサイトと AWS CAF のホワイトペーパー を参照してください。

AWS ワークロード認定フレームワーク (AWS WQF)

データベース移行ワークロードを評価し、移行戦略を推奨し、作業見積もりを提供するツール。AWS WQF は AWS Schema Conversion Tool (AWS SCT) に含まれています。データベーススキーマとコードオブジェクト、アプリケーションコード、依存関係、およびパフォーマンス特性を分析し、評価レポートを提供します。

B

不正なボット

個人や組織に混乱や損害を与えることを目的としたボット。

BCP

「ビジネス継続性計画 (BCP)」をご覧ください。

動作グラフ

リソースの動作とインタラクションを経時的に示した、一元的なインタラクティブビュー。Amazon Detective の動作グラフを使用すると、失敗したログオンの試行、不審な API 呼び出し、その他同様のアクションを調べることができます。詳細については、Detective ドキュメントの「動作グラフのデータ」を参照してください。

ビッグエンディアンシステム

最上位バイトを最初に格納するシステム。「エンディアン性」もご覧ください。

二項分類

バイナリ結果 (2 つの可能なクラスのうちの 1 つ) を予測するプロセス。例えば、お客様の機械学習モデルで「この E メールはスパムですか、それともスパムではありませんか」などの問題を予測する必要があるかもしれません。または「この製品は書籍ですか、車ですか」などの問題を予測する必要があるかもしれません。

ブルームフィルター

要素がセットのメンバーであるかどうかをテストするために使用される、確率的でメモリ効率の高いデータ構造。

ブルー/グリーンデプロイ

それぞれが独立しているが、同一の環境を 2 つ作成するデプロイ戦略。現在のアプリケーションバージョンを 1 つの環境 (ブルー) で実行し、新しいアプリケーションバージョンを別の環境 (グリーン) で実行します。この戦略は、最小限の影響で迅速にロールバックするのに役立ちます。

ボット

インターネット経由で自動タスクを実行し、人間のアクティビティややり取りをシミュレートするソフトウェアアプリケーション。インターネット上の情報のインデックスを作成するウェブクローラーなど、一部のボットは有用または有益です。悪質なボットと呼ばれる他のボットの中には、個人や組織を混乱させたり、損害を与えたりすることを意図したものもあります。

ボットネット

マルウェアに感染しており、ボットハーダーまたはボットオペレーターと呼ばれる単一の当事者によって制御されているボットのネットワーク。ボットネットは、ボットとその影響力を拡大する仕組みとして、非常によく知られています。

ブランチ

コードリポジトリに含まれる領域。リポジトリに最初に作成するブランチは、メインブランチといいます。既存のブランチから新しいブランチを作成し、その新しいブランチで機能を開発したり、バグを修正したりできます。機能を構築するために作成するブランチは、通常、機能ブランチと呼ばれます。機能をリリースする準備ができたら、機能ブランチをメインブランチに統合します。詳細については、「ブランチの概要」(GitHub ドキュメント) を参照してください。

ブレークグラスアクセス

例外的な状況では、承認されたプロセスを通じて、通常アクセス許可 AWS アカウント を持たない にユーザーがすばやくアクセスできるようにします。詳細については、 AWS Well-Architected ガイダンスの「ブレークグラス手順の実装」インジケータを参照してください。

ブラウンフィールド戦略

環境の既存インフラストラクチャ。システムアーキテクチャにブラウンフィールド戦略を導入する場合、現在のシステムとインフラストラクチャの制約に基づいてアーキテクチャを設計します。既存のインフラストラクチャを拡張している場合は、ブラウンフィールド戦略とグリーンフィールド戦略を融合させることもできます。

バッファキャッシュ

アクセス頻度が最も高いデータが保存されるメモリ領域。

ビジネス能力

価値を生み出すためにビジネスが行うこと (営業、カスタマーサービス、マーケティングなど)。マイクロサービスのアーキテクチャと開発の決定は、ビジネス能力によって推進できます。詳細については、AWSでのコンテナ化されたマイクロサービスの実行ホワイトペーパーの「ビジネス機能を中心に組織化」セクションを参照してください。

ビジネス継続性計画 (BCP)

大規模移行など、中断を伴うイベントが運用に与える潜在的な影響に対処し、ビジネスを迅速に再開できるようにする計画。

C

CAF

「AWS クラウド導入フレームワーク」を参照してください

カナリアデプロイ

エンドユーザーへのバージョンリリースを、時間をかけて段階的に行うこと。確信が持てたら新規バージョンをデプロイして、現在のバージョン全体を置き換えます。

CCoE

「Cloud Center of Excellence」を参照してください。

CDC

「変更データキャプチャ」を参照してください。

変更データキャプチャ (CDC)

データソース (データベーステーブルなど) の変更を追跡し、その変更に関するメタデータを記録するプロセス。CDC は、ターゲットシステムでの変更を監査またはレプリケートして同期を維持するなど、さまざまな目的に使用できます。

カオスエンジニアリング

障害や破壊的なイベントを意図的に導入して、システムの耐障害性をテストすること。AWS Fault Injection Service (AWS FIS) を使用して、 AWS ワークロードにストレスを与え、その応答を評価する実験を実行できます。

CI/CD

「継続的インテグレーションと継続的デリバリー」を参照してください。

分類

予測を生成するのに役立つ分類プロセス。分類問題の機械学習モデルは、離散値を予測します。離散値は、常に互いに区別されます。例えば、モデルがイメージ内に車があるかどうかを評価する必要がある場合があります。

シチズンデベロッパー

専門的な技術スキルを持たないノーコード/ローコードプラットフォームを使用して AI アプリケーションを作成するビジネスユーザー。

クライアント側の暗号化

ターゲットがデータ AWS のサービス を受信する前のローカルでのデータの暗号化。

Cloud Center of Excellence (CCoE)

クラウドのベストプラクティスの作成、リソースの移動、移行のタイムラインの確立、大規模変革を通じて組織をリードするなど、組織全体のクラウド導入の取り組みを推進する学際的なチーム。詳細については、 AWS クラウド エンタープライズ戦略ブログの CCoE 投稿を参照してください。

クラウドコンピューティング

リモートデータストレージと IoT デバイス管理に通常使用されるクラウドテクノロジー。クラウドコンピューティングは、一般的に、エッジコンピューティングに接続されています。

クラウド運用モデル

IT 組織において、1 つ以上のクラウド環境を構築、成熟、最適化するために使用される運用モデル。詳細については、「クラウド運用モデルの構築」を参照してください。

導入のクラウドステージ

組織が、 AWS クラウドへの移行時に通常実行する 4 つの段階。

- プロジェクト — 概念実証と学習を目的として、クラウド関連のプロジェクトをいくつか実行する
- 基礎固め — お客様のクラウドの導入を拡大するための基礎的な投資 (ランディングゾーンの作成、CCoE の定義、運用モデルの確立など)
- 移行 — 個々のアプリケーションの移行
- 再発明 — 製品とサービスの最適化、クラウドでのイノベーション

これらのステージは、 AWS クラウド エンタープライズ戦略ブログのブログ記事「クラウドファーストへのジャーニー」と「導入のステージ」で Stephen Orban によって定義されました。移行戦略との関連性については、 AWS「移行準備ガイド」を参照してください。

CMDB

「構成管理データベース (CMDB)」を参照してください。

コードリポジトリ

ソースコードやその他の資産 (ドキュメント、サンプル、スクリプトなど) が保存され、バージョン管理プロセスを通じて更新される場所。一般的なクラウドリポジトリには、GitHub や Bitbucket Cloud があります。コードの各バージョンはブランチと呼ばれます。マイクロサービスの構造では、各リポジトリは 1 つの機能専用です。1 つの CI/CD パイプラインで複数のリポジトリを使用できます。

コールドキャッシュ

空である、または、かなり空きがある、もしくは、古いデータや無関係なデータが含まれているバッファキャッシュ。データベースインスタンスはメインメモリまたはディスクから読み取る必要があり、バッファキャッシュから読み取るよりも時間がかかるため、パフォーマンスに影響します。

コールドデータ

めったにアクセスされず、通常は過去のデータです。この種類のデータをクエリする場合、通常は低速なクエリでも問題ありません。このデータを低パフォーマンスで安価なストレージ階層またはクラスに移動すると、コストを削減することができます。

コンピュータビジョン (CV)

機械学習を使用してデジタルイメージやビデオといった、ビジュアル形式の情報を分析および抽出する AI の分野。例えば、Amazon SageMaker AI では、CV 用の画像処理アルゴリズムを利用できます。

設定ドリフト

ワークロードにおいて、設定が想定した状態から変化すること。これによって、ワークロードが非準拠になる可能性がありますが、この状態は、徐々に生じ、意図的なものではありません。

構成管理データベース (CMDB)

データベースとその IT 環境 (ハードウェアとソフトウェアの両方のコンポーネントとその設定を含む) に関する情報を保存、管理するリポジトリ。通常、CMDB のデータは、移行のポートフォリオの検出と分析の段階で使用します。

コンフォーマンスパック

コンプライアンスチェックとセキュリティチェックをカスタマイズするためにアセンブルできる AWS Config ルールと修復アクションのコレクション。YAML テンプレートを使用して、コンフォーマンスパックを AWS アカウント および リージョンの単一のエンティティとしてデプロイ

することも、組織全体にデプロイすることもできます。詳細については、 AWS Config ドキュメントの「コンフォーマンスパック」を参照してください。

継続的インテグレーションと継続的デリバリー (CI/CD)

ソフトウェアリリースプロセスのソース、ビルド、テスト、ステージング、本番の各ステージを自動化するプロセス。CI/CD は一般的にパイプラインと呼ばれます。プロセスの自動化、生産性の向上、コード品質の向上、配信の加速化を可能にします。詳細については、「継続的デリバリーの利点」を参照してください。CD は継続的デプロイ (Continuous Deployment) の略語でもあります。詳細については「継続的デリバリーと継続的なデプロイ」を参照してください。

CV

「コンピュータビジョン」を参照してください。

D

保管中のデータ

ストレージ内にあるデータなど、常に自社のネットワーク内にあるデータ。

データ分類

ネットワーク内のデータを重要度と機密性に基づいて識別、分類するプロセス。データに適した保護および保持のコントロールを判断する際に役立つため、あらゆるサイバーセキュリティのリスク管理戦略において重要な要素です。データ分類は、 AWS Well-Architected フレームワークのセキュリティの柱のコンポーネントです。詳細については、「データ分類」を参照してください。

データドリフト

実稼働データと ML モデルのトレーニングに使用されたデータとの間に有意な差異が生じたり、入力データが時間の経過と共に有意に変化したりすることです。データドリフトは、ML モデル予測の全体的な品質、精度、公平性を低下させる可能性があります。

転送中のデータ

ネットワーク内 (ネットワークリソース間など) を活発に移動するデータ。

データメッシュ

非一元的で分散型のデータ所有権を持つとともに、一元的な管理およびガバナンスを行えるアーキテクチャフレームワーク。

データ最小化

厳密に必要なデータのみを収集し、処理するという原則。でデータ最小化を実践 AWS クラウドすることで、プライバシーリスク、コスト、分析のカーボンフットプリントを削減できます。

データ境界

AWS 環境内の一連の予防ガードレール。信頼された ID のみが、期待されるネットワークから信頼されたリソースにアクセスできるようにします。詳細については、「 でのデータ境界の構築 AWS」を参照してください。

データの前処理

raw データをお客様の機械学習モデルで簡単に解析できる形式に変換すること。データの前処理とは、特定の列または行を削除して、欠落している、矛盾している、または重複する値に対処することを意味します。

データ出所

データの生成、送信、保存の方法など、データのライフサイクル全体を通じてデータの出所と履歴を追跡するプロセス。

データ件名

データを収集、処理している個人。

データウェアハウス

分析などのビジネスインテリジェンスをサポートするデータ管理システム。データウェアハウスには、一般的に、大量の履歴データが含まれており、多くの場合、それらはクエリや分析に使用されます。

データベース定義言語 (DDL)

データベース内のテーブルやオブジェクトの構造を作成または変更するためのステートメントまたはコマンド。

データベース操作言語 (DML)

データベース内の情報を変更 (挿入、更新、削除) するためのステートメントまたはコマンド。

DDL

「データベース定義言語」を参照してください。

ディープアンサンブル

予測のために複数の深層学習モデルを組み合わせます。ディープアンサンブルを使用して、より正確な予測を取得したり、予測の不確実性を推定したりできます。

深層学習

人工ニューラルネットワークの複数層を使用して、入力データと対象のターゲット変数の間のマッピングを識別する機械学習サブフィールド。

多層防御

一連のセキュリティメカニズムとコントロールをコンピュータネットワーク全体に層状に重ねて、ネットワークとその内部にあるデータの機密性、整合性、可用性を保護する情報セキュリティの手法。この戦略を採用するときは AWS、 AWS Organizations 構造の異なるレイヤーに複数のコントロールを追加して、リソースの安全性を確保します。たとえば、多層防御アプローチでは、多要素認証、ネットワークセグメンテーション、暗号化を組み合わせることができます。

委任管理者

では AWS Organizations、互換性のあるサービスが AWS メンバーアカウントを登録して組織のアカウントを管理し、そのサービスのアクセス許可を管理できます。このアカウントを、そのサービスの委任管理者と呼びます。詳細、および互換性のあるサービスの一覧は、 AWS Organizations ドキュメントの「AWS Organizationsで使用できるサービス」を参照してください。

トラブルシューティング

アプリケーション、新機能、コードの修正をターゲットの環境で利用できるようにするプロセス。デプロイでは、コードベースに変更を施した後、アプリケーションの環境でそのコードベースを構築して実行します。

開発環境

「環境」を参照してください。

検出管理

イベントが発生したときに、検出、ログ記録、警告を行うように設計されたセキュリティコントロール。これらのコントロールは副次的な防衛手段であり、実行中の予防的コントロールをすり抜けたセキュリティイベントをユーザーに警告します。詳細については、「 AWSでのセキュリティコントロールの実装」の「検出的コントロール」を参照してください。

開発バリューストリームマッピング (DVSM)

ソフトウェア開発ライフサイクルのスピードと品質に悪影響を及ぼす制約を特定し、優先順位を付けるために使用されるプロセス。DVSM は、もともとリーンマニュファクチャリング・プラクティスのために設計されたバリューストリームマッピング・プロセスを拡張したものです。ソフトウェア開発プロセスを通じて価値を創造し、動かすために必要なステップとチームに焦点を当てています。

デジタルツイン

建物、工場、産業機器、生産ラインなど、現実世界のシステムを仮想的に表現したものです。デジタルツインは、予知保全、リモートモニタリング、生産最適化をサポートします。

ディメンションテーブル

スタースキーマにおいて、ファクトテーブルの定量データに関するデータ属性が含まれる小さいテーブル。ディメンションテーブルの属性は、通常、テキストフィールド、またはテキストのように扱える個別の数値で示されます。これらの属性は、一般的に、クエリの制約、フィルタリング、結果セットのラベル付けに使用されます。

ディザスタ

ワークロードまたはシステムが、導入されている主要な場所でのビジネス目標の達成を妨げるイベント。これらのイベントは、自然災害、技術的障害、または意図しない設定ミスやマルウェア攻撃などの人間の行動の結果である場合があります。

ディザスタリカバリ (DR)

ディザスタによるダウンタイムとデータ損失を最小限に抑えるための戦略とプロセス。詳細については、 AWS Well-Architected フレームワークの「 でのワークロードのディザスタリカバリ AWS: クラウドでのリカバリ」を参照してください。

DML

「データベース操作言語」を参照してください。

ドメイン駆動型設計

各コンポーネントが提供している変化を続けるドメイン、またはコアビジネス目標にコンポーネントを接続して、複雑なソフトウェアシステムを開発するアプローチ。この概念は、エリック・エヴァンスの著書、Domain-Driven Design: Tackling Complexity in the Heart of Software (ドメイン駆動設計:ソフトウェアの中心における複雑さへの取り組み) で紹介されています (ボストン: Addison-Wesley Professional、2003)。strangler fig パターンでドメイン駆動型設計を使用す

る方法の詳細については、「コンテナと Amazon API Gateway を使用して、従来の Microsoft ASP.NET (ASMX) ウェブサービスを段階的にモダナイズ」を参照してください。

DR

「ディザスタリカバリ」を参照してください。

ドリフト検出

ベースライン設定からの偏差を追跡します。たとえば、 AWS CloudFormation を使用してシステムリソースのドリフトを検出したり、 を使用して AWS Control Tower 、ガバナンス要件への準拠に影響するランディングゾーンの変更を検出したりできます。

DVSM

「開発バリューストリームマッピング」を参照してください。

E

EDA

「探索的データ分析」を参照してください。

EDI

「電子データ交換」を参照してください。

エッジコンピューティング

IoT ネットワークのエッジにあるスマートデバイスの計算能力を高めるテクノロジー。クラウドコンピューティングと比較すると、エッジコンピューティングは通信レイテンシーを短縮し、応答時間を改善できます。

電子データ交換 (EDI)

組織間で行う、ビジネスドキュメントの自動交換。詳細については、「電子データ交換とは」を参照してください。

暗号化

人間が読み取り可能なプレーンテキストデータを暗号文に変換するコンピューティング処理。

暗号化キー

暗号化アルゴリズムが生成した、ランダム化されたビットからなる暗号文字列。キーの長さは決まっておらず、各キーは予測できないように、一意になるように設計されています。

エンディアン

コンピュータメモリにバイトが格納される順序。ビッグエンディアンシステムでは、最上位バイトが最初に格納されます。リトルエンディアンシステムでは、最下位バイトが最初に格納されます。

エンドポイント

「サービスエンドポイント」を参照してください。

エンドポイントサービス

仮想プライベートクラウド (VPC) 内でホストして、他のユーザーと共有できるサービス。を使用してエンドポイントサービスを作成し AWS PrivateLink 、他の AWS アカウント または AWS Identity and Access Management (IAM) プリンシパルにアクセス許可を付与できます。これらのアカウントまたはプリンシパルは、インターフェイス VPC エンドポイントを作成することで、エンドポイントサービスにプライベートに接続できます。詳細については、Amazon Virtual Private Cloud (Amazon VPC) ドキュメントの「エンドポイントサービスを作成する」を参照してください。

エンタープライズリソースプランニング (ERP)

エンタープライズの主要なビジネスプロセス (会計、MES、プロジェクト管理など) を自動化および管理するシステム。

エンベロープ暗号化

暗号化キーを、別の暗号化キーを使用して暗号化するプロセス。詳細については、 AWS Key Management Service (AWS KMS) ドキュメントの「エンベロープ暗号化」を参照してください。

環境

実行中のアプリケーションのインスタンス。クラウドコンピューティングにおける一般的な環境の種類は以下のとおりです。

- 開発環境 — アプリケーションのメンテナンスを担当するコアチームのみが使用できる、実行中のアプリケーションのインスタンス。開発環境は、上位の環境に昇格させる変更をテストするときに使用します。このタイプの環境は、テスト環境と呼ばれることもあります。
- 下位環境 — 初期ビルドやテストに使用される環境など、アプリケーションのすべての開発環境。
- 本番環境 — エンドユーザーがアクセスできる、実行中のアプリケーションのインスタンス。CI/CD パイプラインでは、本番環境が最後のデプロイ環境になります。

- 上位環境 — コア開発チーム以外のユーザーがアクセスできるすべての環境。これには、本番環境、本番前環境、ユーザー承認テスト環境などが含まれます。

エピック

アジャイル方法論で、お客様の作業の整理と優先順位付けに役立つ機能カテゴリ。エピックでは、要件と実装タスクの概要についてハイレベルな説明を提供します。例えば、 AWS CAF セキュリティエピックには、ID とアクセスの管理、検出コントロール、インフラストラクチャセキュリティ、データ保護、インシデント対応が含まれます。 AWS 移行戦略のエピックの詳細については、プログラム実装ガイドを参照してください。

ERP

「エンタープライズリソース計画」を参照してください。

探索的データ分析 (EDA)

データセットを分析してその主な特性を理解するプロセス。お客様は、データを収集または集計してから、パターンの検出、異常の検出、および前提条件のチェックのための初期調査を実行します。EDA は、統計の概要を計算し、データの可視化を作成することによって実行されます。

F

ファクトテーブル

スタースキーマの中央にあるテーブル。ビジネスオペレーションに関する定量的データが保存されます。一般的に、ファクトテーブルは、2 種類の列で構成されます。1 つは測定値が含まれる列、もう 1 つはディメンションテーブルへの外部キーが含まれる列です。

フェイルファスト

開発ライフサイクルを短縮するために、頻繁かつ段階的にテストを行う哲学であり、アジャイルアプローチでは、この考え方がきわめて重要です。

障害分離境界

では AWS クラウド、アベイラビリティーゾーン AWS リージョン、コントロールプレーン、データプレーンなどの境界で、障害の影響を制限し、ワークロードの耐障害性を向上させるのに役立ちます。詳細については、「AWS 障害分離境界」を参照してください。

機能ブランチ

「ブランチ」を参照してください。

特徴量

お客様が予測に使用する入力データ。例えば、製造コンテキストでは、特徴量は製造ラインから定期的にキャプチャされるイメージの可能性もあります。

特徴量重要度

モデルの予測に対する特徴量の重要性。これは通常、Shapley Additive Deskonations (SHAP) や積分勾配など、さまざまな手法で計算できる数値スコアで表されます。詳細については、「 を使用した機械学習モデルの解釈可能性 AWS」を参照してください。

機能変換

追加のソースによるデータのエンリッチ化、値のスケーリング、単一のデータフィールドからの複数の情報セットの抽出など、機械学習プロセスのデータを最適化すること。これにより、機械学習モデルはデータの恩恵を受けることができます。例えば、「2021-05-27 00:15:37」の日付を「2021 年」、「5 月」、「木」、「15」に分解すると、学習アルゴリズムがさまざまなデータコンポーネントに関連する微妙に異なるパターンを学習するのに役立ちます。

数ショットプロンプト

LLM に、タスクと望ましい出力を示す例を少数提示した後に、類似のタスクを実行させること。この手法は、プロンプトに記述された例 (ショット) からモデルが学習する「インコンテキスト学習」の一種です。数ショットプロンプトは、特定のフォーマット、推論、専門知識が必要なタスクに効果的です。「ゼロショットプロンプト」も参照してください。

FGAC

「きめ細かなアクセス制御」を参照してください。

きめ細かなアクセス制御 (FGAC)

複数の条件を使用してアクセス要求を許可または拒否すること。

フラッシュカット移行

変更データのキャプチャによる継続的なデータ複製を利用して、段階的なアプローチではなく、可能な限り短時間でデータを移行するデータベース移行方法。目的はダウンタイムを最小限に抑えることです。

FM

「基盤モデル」を参照してください。

基盤モデル (FM)

大規模な深層学習ニューラルネットワークであり、一般化およびラベル付けされていないデータからなる大規模データセットでトレーニングされています。FM により、言語理解、テキストおよび画像生成、自然言語での会話といった、一般的な各種タスクを実行できます。詳細については、「基盤モデルとは何ですか?」を参照してください。

FM ゲートウェイ

基盤モデルへのアクセスを制御および正規化する一元化された仲介者。LLM ゲートウェイとも呼ばれます。

G

生成 AI

AI モデルのサブセット。大量のデータでトレーニングされており、シンプルなテキストプロンプトを使用して、画像、動画、テキスト、オーディオなどの新しいコンテンツやアーティファクトを作成できます。詳細については、「生成 AI とは何ですか?」を参照してください。

ジオブロッキング

「地理的制限」を参照してください。

地理的制限 (ジオブロッキング)

特定の国のユーザーがコンテンツ配信にアクセスできないようにするための、Amazon CloudFront のオプション。アクセスを許可する国と禁止する国は、許可リストまたは禁止リストを使って指定します。詳細については、CloudFront ドキュメントの「コンテンツの地理的ディストリビューションの制限」を参照してください。

Gitflow ワークフロー

下位環境と上位環境が、ソースコードリポジトリでそれぞれ異なるブランチを使用する方法。Gitflow ワークフローは古いと見なされている方法であり、トランクベースのワークフローは推奨されている新しい方法です。

ゴールデンイメージ

システムまたはソフトウェアのスナップショットであり、システムまたはソフトウェアの新規インスタンスをデプロイするテンプレートとして使用されます。製造の例で言えば、ゴールデンイメージを使用すると、複数のデバイスにソフトウェアをプロビジョニングして、デバイス製造オペレーションの速度、スケーラビリティ、生産性を向上させることができます。

グリーンフィールド戦略

新しい環境に既存のインフラストラクチャが存在しないこと。システムアーキテクチャにグリーンフィールド戦略を導入する場合、既存のインフラストラクチャ (別名ブラウンフィールド) との互換性の制約を受けることなく、あらゆる新しいテクノロジーを選択できます。既存のインフラストラクチャを拡張している場合は、ブラウンフィールド戦略とグリーンフィールド戦略を融合させることもできます。

ガードレール

組織単位 (OU) 全般のリソース、ポリシー、コンプライアンスを管理するのに役立つ概略的なルール。予防ガードレールは、コンプライアンス基準に一致するようにポリシーを実施します。これらは、サービスコントロールポリシーと IAM アクセス許可の境界を使用して実装されます。検出ガードレールは、ポリシー違反やコンプライアンス上の問題を検出し、修復のためのアラートを発信します。これらは AWS Config、、Amazon GuardDuty AWS Security Hub CSPM、、 AWS Trusted Advisor Amazon Inspector、およびカスタム AWS Lambda チェックを使用して実装されます。

ガードレール (AI)

エージェントの入力と出力をフィルタリング、検証、制約して、責任ある安全な AI 動作を確保するのに役立つ安全メカニズム。

H

HA

「高可用性」を参照してください。

異種混在データベースの移行

別のデータベースエンジンを使用するターゲットデータベースへお客様の出典データベースの移行 (例えば、Oracle から Amazon Aurora)。異種間移行は通常、アーキテクチャの再設計作業の一部であり、スキーマの変換は複雑なタスクになる可能性があります。AWS は、スキーマの変換に役立つ AWS SCTを提供します。

高可用性 (HA)

課題や災害が発生した場合に、介入なしにワークロードを継続的に運用できること。HA システムは、自動的にフェイルオーバーし、一貫して高品質のパフォーマンスを提供し、パフォーマンスへの影響を最小限に抑えながらさまざまな負荷や障害を処理するように設計されています。

ヒストリアンのモダナイゼーション

製造業のニーズによりよく応えるために、オペレーションテクノロジー (OT) システムをモダナイズし、アップグレードするためのアプローチ。ヒストリアンは、工場内のさまざまなソースからデータを収集して保存するために使用されるデータベースの一種です。

ホールドアウトデータ

機械学習モデルのトレーニング用データセットから保留される、ラベル付き履歴データの一部。ホールドアウトデータを使用すると、モデル予測をホールドアウトデータと比較して、モデルのパフォーマンスを評価できます。

ヒューman-in-the-loop (HitL)

エージェント???の実行が重要な決定時点で人間によるレビューと承認のために一時停止するワークフローパターン。

同種データベースの移行

お客様の出典データベースを、同じデータベースエンジンを共有するターゲットデータベース (Microsoft SQL Server から Amazon RDS for SQL Server など) に移行する。同種間移行は、通常、リホストまたはリプラットフォーム化の作業の一部です。ネイティブデータベースユーティリティを使用して、スキーマを移行できます。

ホットデータ

リアルタイムデータや最近の翻訳データなど、頻繁にアクセスされるデータ。通常、このデータには高速なクエリ応答を提供する高性能なストレージ階層またはクラスが必要です。

ホットフィックス

本番環境の重大な問題を修正するために緊急で配布されるプログラム。緊急性が高いため、通常の DevOps のリリースワークフローからは外れた形で実施されます。

ハイパーケア期間

カットオーバー直後、移行したアプリケーションを移行チームがクラウドで管理、監視して問題に対処する期間。通常、この期間は 1～4 日です。ハイパーケア期間が終了すると、アプリケーションに対する責任は一般的に移行チームからクラウドオペレーションチームに移ります。

I

IaC

「Infrastructure as Code」を参照してください。

ID ベースのポリシー

AWS クラウド 環境内のアクセス許可を定義する 1 つ以上の IAM プリンシパルにアタッチされたポリシー。

アイドル状態のアプリケーション

90 日間の平均的な CPU およびメモリ使用率が 5～20% のアプリケーション。移行プロジェクトでは、これらのアプリケーションを廃止するか、オンプレミスに保持するのが一般的です。

IIoT

「インダストリアル IoT」を参照してください。

イミュータブルインフラストラクチャ

既存インフラストラクチャの更新、パッチ適用、変更などを行わずに、本番環境ワークロードに使用する新規インフラストラクチャをデプロイするモデル。本質的に、イミュータブルインフラストラクチャは、ミュータブルインフラストラクチャよりも一貫性、信頼性、予測性に優れています。詳細については、 AWS Well-Architected フレームワークにある「イミュータブルインフラストラクチャを使用してデプロイする」のベストプラクティスを参照してください。

インバウンド (受信) VPC

AWS マルチアカウントアーキテクチャでは、アプリケーションの外部からネットワーク接続を受け入れ、検査し、ルーティングする VPC。AWS Security Reference Architecture では、アプリケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

増分移行

アプリケーションを 1 回ですべてカットオーバーするのではなく、小さい要素に分けて移行するカットオーバー戦略。例えば、最初は少数のマイクロサービスまたはユーザーのみを新しいシステムに移行する場合があります。すべてが正常に機能することを確認できたら、残りのマイクロサービスやユーザーを段階的に移行し、レガシーシステムを廃止できるようにします。この戦略により、大規模な移行に伴うリスクが軽減されます。

インダストリー 4.0

2016 年に Klaus Schwab 氏が提唱した用語で、接続、リアルタイムデータ、オートメーション、分析、AI/ML の進歩による、ビジネスプロセスのモダナイズを意味します。

インフラストラクチャ

アプリケーションの環境に含まれるすべてのリソースとアセット。

Infrastructure as Code (IaC)

アプリケーションのインフラストラクチャを一連の設定ファイルを使用してプロビジョニングし、管理するプロセス。IaC は、新しい環境を再現可能で信頼性が高く、一貫性のあるものにするため、インフラストラクチャを一元的に管理し、リソースを標準化し、スケールを迅速に行えるように設計されています。

インダストリアル IoT (IIoT)

製造、エネルギー、自動車、ヘルスケア、ライフサイエンス、農業などの産業部門におけるインターネットに接続されたセンサーやデバイスの使用。詳細については、「インダストリアル IoT (IIoT) デジタルトランスフォーメーション戦略の構築」を参照してください。

インスペクション VPC

AWS マルチアカウントアーキテクチャでは、VPC (同一または異なる 内 AWS リージョン)、インターネット、オンプレミスネットワーク間のネットワークトラフィックの検査を管理する一元化された VPCs。AWS Security Reference Architecture では、アプリケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

IoT

インターネットまたはローカル通信ネットワークを介して他のデバイスやシステムと通信する、センサーまたはプロセッサが組み込まれた接続済み物理オブジェクトのネットワーク。詳細については、「IoT とは」を参照してください。

解釈可能性

機械学習モデルの特性で、モデルの予測がその入力にどのように依存するかを人間が理解できる度合いを表します。詳細については、「 を使用した機械学習モデルの解釈可能性 AWS」を参照してください。

IoT

「IoT」を参照してください。

IT 情報ライブラリ (ITIL)

IT サービスを提供し、これらのサービスをビジネス要件に合わせるための一連のベストプラクティス。ITIL は ITSM の基盤を提供します。

IT サービス管理 (ITSM)

組織の IT サービスの設計、実装、管理、およびサポートに関連する活動。クラウドオペレーションと ITSM ツールの統合については、オペレーション統合ガイドを参照してください。

ITIL

「IT 情報ライブラリ」を参照してください。

ITSM

「IT サービス管理」を参照してください。

L

ラベルベースアクセス制御 (LBAC)

強制アクセス制御 (MAC) の実装で、ユーザーとデータ自体にそれぞれセキュリティラベル値が明示的に割り当てられます。ユーザーセキュリティラベルとデータセキュリティラベルが交差する部分によって、ユーザーに表示される行と列が決まります。

ランディングゾーン

ランディングゾーンは、スケーラブルで安全な、適切に設計されたマルチアカウント AWS 環境です。これは、組織がセキュリティおよびインフラストラクチャ環境に自信を持ってワークロードとアプリケーションを迅速に起動してデプロイできる出発点です。ランディングゾーンの詳細については、「安全でスケーラブルなマルチアカウント AWS 環境のセットアップ」を参照してください。

大規模言語モデル (LLM)

大量のデータで事前トレーニングされた深層学習 AI モデル。LLM では、質問への回答、ドキュメントの要約、他言語へのテキスト翻訳、文を完成させるなど、さまざまなタスクを実行できます。詳細については、「大規模言語モデル (LLM) とは何ですか?」を参照してください。

大規模な移行

300 台以上のサーバの移行。

LBAC

「ラベルベースアクセス制御」を参照してください。

最小特権

タスクの実行には必要最低限の権限を付与するという、セキュリティのベストプラクティス。詳細については、IAM ドキュメントの「最小特権アクセス許可を適用する」を参照してください。

リフトアンドシフト

「7 Rs」を参照してください。

リトルエンディアンシステム

最下位バイトを最初に格納するシステム。「エンディアン性」もご覧ください。

LLM

「大規模言語モデル」を参照してください。

下位環境

「環境」を参照してください。

M

機械学習 (ML)

パターン認識と学習にアルゴリズムと手法を使用する人工知能の一種。ML は、モノのインターネット (IoT) データなどの記録されたデータを分析して学習し、パターンに基づく統計モデルを生成します。詳細については、「機械学習」を参照してください。

メインブランチ

「ブランチ」を参照してください。

マルウェア

コンピュータのセキュリティやプライバシーを侵害するように設計されたソフトウェア。マルウェアは、コンピュータシステムの中断、機密情報の漏洩、不正アクセスを招く可能性があります。マルウェアの例には、ウイルス、ワーム、ランサムウェア、トロイの木馬、スパイウェア、キーロガーなどがあります。

マネージドサービス

AWS のサービス はインフラストラクチャレイヤー、オペレーティングシステム、プラットフォーム AWS を運用し、エンドポイントにアクセスしてデータを保存および取得します。

マネージドサービスの例として、Amazon Simple Storage Service (Amazon S3) と Amazon DynamoDB が挙げられます。このサービスは、抽象化されたサービスとも呼ばれます。

製造実行システム (MES)

生産プロセスを追跡、モニタリング、文書化、制御するソフトウェアシステムであり、工場では、これによって、原材料から製品を完成させます。

MAP

「Migration Acceleration Program」を参照してください。

MCP

「モデルコンテキストプロトコル」を参照してください。

モデルコンテキストプロトコル (MCP)

エージェントツーツール通信のステートレスプロトコル。

MCP サーバー

Model Context Protocol を通じて 1 つ以上のツールを公開するサービス。

メカニズム

ツールを作成してその導入を推進し、導入結果を調べて調整を行うための包括的なプロセス。メカニズムとは、運用中にそれ自体を強化し改善するサイクルを意味します。詳細については、AWS 「 Well-Architected フレームワーク」の「メカニズムの構築」を参照してください。

メンバーアカウント

組織の一部である管理アカウント AWS アカウント 以外のすべて AWS Organizations。 アカウントが 組織のメンバーになることができるのは、一度に 1 つのみです。

MES

「製造実行システム」を参照してください。

Message Queuing Telemetry Transport (MQTT)

発行/サブスクライブのパターンに基づく、軽量のマシンツーマシン (M2M) 通信プロトコルであり、リソースに限りのある IoT デバイスに使用されます。

マイクロサービス

明確に定義された API を介して通信し、通常は小規模な自己完結型のチームが所有する、小規模で独立したサービスです。例えば、保険システムには、販売やマーケティングなどのビジネス機能、または購買、請求、分析などのサブドメインにマッピングするマイクロサービスが含まれ

る場合があります。マイクロサービスの利点には、俊敏性、柔軟なスケーリング、容易なデプロイ、再利用可能なコード、回復力などがあります。詳細については、AWS 「サーバーレスサービスを使用したマイクロサービスの統合」を参照してください。

マイクロサービスアーキテクチャ

各アプリケーションプロセスをマイクロサービスとして実行する独立したコンポーネントを使用してアプリケーションを構築するアプローチ。これらのマイクロサービスは、軽量 API を使用して、明確に定義されたインターフェイスを介して通信します。このアーキテクチャの各マイクロサービスは、アプリケーションの特定の機能に対する需要を満たすように更新、デプロイ、およびスケーリングできます。詳細については、「 でのマイクロサービスの実装 AWS」を参照してください。

Migration Acceleration Program (MAP)

組織がクラウドに移行するための強力な運用基盤を構築し、移行の初期コストを相殺するのに役立つコンサルティングサポート、トレーニング、サービスを提供する AWS プログラム。MAP には、組織的な方法でレガシー移行を実行するための移行方法論と、一般的な移行シナリオを自動化および高速化する一連のツールが含まれています。

大規模な移行

アプリケーションポートフォリオの大部分を次々にクラウドに移行し、各ウェーブでより多くのアプリケーションを高速に移動させるプロセス。この段階では、以前の段階から学んだベストプラクティスと教訓を使用して、移行ファクトリー チーム、ツール、プロセスのうち、オートメーションとアジャイルデリバリーによってワークロードの移行を合理化します。これは、AWS 移行戦略 の第 3 段階です。

移行ファクトリー

自動化された俊敏性のあるアプローチにより、ワークロードの移行を合理化する部門横断的なチーム。移行ファクトリーチームには、通常、運用、ビジネスアナリストおよび所有者、移行エンジニア、デベロッパー、およびスプリントで作業する DevOps プロフェッショナルが含まれます。エンタープライズアプリケーションポートフォリオの 20～50% は、ファクトリーのアプローチによって最適化できる反復パターンで構成されています。詳細については、このコンテンツセットの移行ファクトリーに関する解説とCloud Migration Factory ガイドを参照してください。

移行メタデータ

移行を完了するために必要なアプリケーションおよびサーバーに関する情報。移行パターンごとに、異なる一連の移行メタデータが必要です。移行メタデータの例としては、ターゲットサブネット、セキュリティグループ、 AWS アカウントなどがあります。

移行パターン

移行戦略、移行先、および使用する移行アプリケーションまたはサービスを詳述する、反復可能な移行タスク。例: AWS Application Migration Service を使用して Amazon EC2 への移行をリホストします。

Migration Portfolio Assessment (MPA)

オンラインツール。これによって、 AWS クラウドに移行するビジネスケースの検証に必要な情報を得られます。MPA は、詳細なポートフォリオ評価 (サーバーの適切なサイジング、価格設定、TCO 比較、移行コスト分析) および移行プラン (アプリケーションデータの分析とデータ収集、アプリケーションのグループ化、移行の優先順位付け、およびウェーブプランニング) を提供します。MPA ツール (ログインが必要) は、すべての AWS コンサルタントと APN パートナーコンサルタントが無料で利用できます。

移行準備状況評価 (MRA)

AWS CAF を使用して、組織のクラウド準備状況に関するインサイトを取得し、長所と短所を特定し、特定されたギャップを埋めるためのアクションプランを構築するプロセス。詳細については、移行準備状況ガイドを参照してください。MRA は、AWS 移行戦略の第一段階です。

移行戦略

ワークロードを AWS クラウドに移行するために使用するアプローチ。詳細については、この用語集の 7 Rs エントリと、「組織を動員して大規模な移行を加速する」を参照してください。

ML

「機械学習」を参照してください。

モダナイゼーション

古い (レガシーまたはモノリシック) アプリケーションとそのインフラストラクチャをクラウド内の俊敏で弾力性のある高可用性システムに変換して、コストを削減し、効率を高め、イノベーションを活用します。詳細については、「AWS クラウドでのアプリケーションのモダナイズ戦略」を参照してください。

モダナイゼーション準備状況評価

組織のアプリケーションのモダナイゼーションの準備状況を判断し、利点、リスク、依存関係を特定し、組織がこれらのアプリケーションの将来の状態をどの程度適切にサポートできるかを決定するのに役立つ評価。評価の結果として、ターゲットアーキテクチャのブループリント、モダナイゼーションプロセスの開発段階とマイルストーンを詳述したロードマップ、特定された

ギャップに対処するためのアクションプランが得られます。詳細については、「AWS クラウドでのアプリケーションのモダナイゼーションの準備状況を評価する」を参照してください。

モノリシックアプリケーション (モノリス)

緊密に結合されたプロセスを持つ単一のサービスとして実行されるアプリケーション。モノリシックアプリケーションにはいくつかの欠点があります。1 つのアプリケーション機能エクスペリエンスの需要が急増する場合は、アーキテクチャ全体をスケーリングする必要があります。モノリシックアプリケーションの特徴を追加または改善することは、コードベースが大きくなると複雑になります。これらの問題に対処するには、マイクロサービスアーキテクチャを使用できます。詳細については、「モノリスをマイクロサービスに分解する」を参照してください。

MPA

「Migration Portfolio Assessment」を参照してください。

MQTT

「Message Queuing Telemetry Transport」を参照してください。

多クラス分類

複数のクラスの予測を生成するプロセス (2 つ以上の結果の 1 つを予測します)。例えば、機械学習モデルが、「この製品は書籍、自動車、電話のいずれですか?」 または、「このお客様にとって最も関心のある商品のカテゴリはどれですか?」と聞くかもしれません。

ミュータブルなインフラストラクチャ

本番ワークロードに使用する既存のインフラストラクチャを更新および変更するためのモデル。Well-Architected AWS フレームワークでは、一貫性、信頼性、予測可能性を向上させるために、イミュータブルインフラストラクチャの使用をベストプラクティスとして推奨しています。

O

OAC

「オリジンアクセス制御」を参照してください。

OAI

「オリジンアクセスアイデンティティ」を参照してください。

OCM

「組織変更管理」を参照してください。

オフライン移行

移行プロセス中にソースワークロードを停止させる移行方法。この方法はダウンタイムが長くなるため、通常は重要ではない小規模なワークロードに使用されます。

OI

「オペレーション統合」を参照してください。

Ola

「オペレーショナルレベルアグリーメント」を参照してください。

オンライン移行

ソースワークロードをオフラインにせずにターゲットシステムにコピーする移行方法。ワークロードに接続されているアプリケーションは、移行中も動作し続けることができます。この方法はダウンタイムがゼロから最小限で済むため、通常は重要な本番稼働環境のワークロードに使用されます。

OPC-UA

「Open Process Communications - Unified Architecture」を参照してください。

Open Process Communications - Unified Architecture (OPC-UA)

産業オートメーション用のマシンツーマシン (M2M) 通信プロトコル。OPC-UA により、相互運用の際に、データ暗号化、認証、認可の各スキームを標準化できます。

オペレーショナルレベルアグリーメント (OLA)

サービスレベルアグリーメント (SLA) をサポートするために、どの機能的 IT グループが互いに提供することを約束するかを明確にする契約。

運用準備状況レビュー (ORR)

質問と関連するベストプラクティスのチェックリスト。インシデントや起こり得る障害を理解、評価、防止したり、その範囲を縮小したりする際に役立ちます。詳細については、 AWS Well-Architected フレームワークの「Operational Readiness Reviews (ORR)」を参照してください。

運用テクノロジー (OT)

産業オペレーション、機器、インフラストラクチャを制御するために物理環境と連携させるハードウェアおよびソフトウェアシステム。製造分野では、Industry 4.0 への変革を進める上で、OT と情報技術 (IT) システムの統合に焦点が当てられています。

オペレーション統合 (OI)

クラウドでオペレーションをモダナイズするプロセスには、準備計画、オートメーション、統合が含まれます。詳細については、オペレーション統合ガイドを参照してください。

組織の証跡

組織 AWS アカウント 内のすべての のすべてのイベント AWS CloudTrail をログに記録する によって作成された証跡 AWS Organizations。証跡は、組織に含まれている各 AWS アカウント に作成され、各アカウントのアクティビティを追跡します。詳細については、CloudTrail ドキュメントの「組織の証跡の作成」を参照してください。

組織変更管理 (OCM)

人材、文化、リーダーシップの観点から、主要な破壊的なビジネス変革を管理するためのフレームワーク。OCM は、変化の導入を加速し、移行問題に対処し、文化や組織の変化を推進することで、組織が新しいシステムと戦略の準備と移行するのを支援します。 AWS 移行戦略では、クラウド導入プロジェクトに必要な変化のスピードにより、このフレームワークは人材アクセラレーションと呼ばれます。詳細については、OCM ガイドを参照してください。

オリジンアクセス制御 (OAC)

Amazon Simple Storage Service (Amazon S3) コンテンツを保護するための、CloudFront のアクセス制限の強化オプション。OAC は AWS リージョン、すべての S3 バケット、 AWS KMS (SSE-KMS) によるサーバー側の暗号化、S3 バケットへの動的 PUT および DELETEリクエストをサポートします。

オリジンアクセスアイデンティティ (OAI)

CloudFront の、Amazon S3 コンテンツを保護するためのアクセス制限オプション。OAI を使用すると、CloudFront が、Amazon S3 に認証可能なプリンシパルを作成します。認証されたプリンシパルは、S3 バケット内のコンテンツに、特定の CloudFront ディストリビューションを介してのみアクセスできます。OAC も併せて参照してください。OAC では、より詳細な、強化されたアクセス制御が可能です。

ORR

「運用準備状況レビュー」を参照してください。

OT

「運用テクノロジー」を参照してください。

アウトバウンド (送信) VPC

AWS マルチアカウントアーキテクチャでは、アプリケーション内から開始されたネットワーク接続を処理する VPC。AWS Security Reference Architecture では、アプリケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

P

アクセス許可の境界

ユーザーまたはロールが使用できるアクセス許可の上限を設定する、IAM プリンシパルにアタッチされる IAM 管理ポリシー。詳細については、IAM ドキュメントのアクセス許可の境界を参照してください。

個人を特定できる情報 (PII)

直接閲覧した場合、または他の関連データと組み合わせた場合に、個人の身元を合理的に推測するために使用できる情報。PII の例には、氏名、住所、連絡先情報などがあります。

PII

「個人を特定できる情報」を参照してください。

プレイブック

クラウドでのコアオペレーション機能の提供など、移行に関連する作業を取り込む、事前定義された一連のステップ。プレイブックは、スクリプト、自動ランブック、またはお客様のモダナイズされた環境を運用するために必要なプロセスや手順の要約などの形式をとることができます。

PLC

「プログラマブルロジックコントローラー」を参照してください。

PLM

「製品ライフサイクル管理」を参照してください。

ポリシー

次の操作を可能にするオブジェクト: アクセス許可を定義する (ID ベースのポリシーを参照)。アクセス条件を指定する (リソースベースのポリシーを参照)。 AWS Organizations の組織における全アカウントにアクセス許可の上限を定義する (サービスコントロールポリシーを参照)。

多言語の永続性

データアクセスパターンやその他の要件に基づいて、マイクロサービスのデータストレージテクノロジーを個別に選択します。マイクロサービスが同じデータストレージテクノロジーを使用している場合、実装上の問題が発生したり、パフォーマンスが低下する可能性があります。マイクロサービスは、要件に最も適合したデータストアを使用すると、より簡単に実装でき、パフォーマンスとスケーラビリティが向上します。

ポートフォリオ評価

移行を計画するために、アプリケーションポートフォリオの検出、分析、優先順位付けを行うプロセス。詳細については、「移行の準備状況の評価」を参照してください。

述語

`true` または `false` を返すためのクエリ条件。一般的に、`WHERE` 句に記述されます。

述語プッシュダウン

データベースクエリを最適化する手法。これによって、転送前にクエリ内のデータをフィルタリングします。この手法を取ると、リレーショナルデータベースから取得し処理する必要のあるデータの量が減少するため、クエリのパフォーマンスが向上します。

予防的コントロール

イベントの発生を防ぐように設計されたセキュリティコントロール。このコントロールは、ネットワークへの不正アクセスや好ましくない変更を防ぐ最前線の防御です。詳細については、「AWSでのセキュリティコントロールの実装」の「予防的コントロール」を参照してください。

プリンシパル

アクションを実行し AWS 、リソースにアクセスできる のエンティティ。このエンティティは通常、、IAM AWS アカウントロール、または ユーザーのルートユーザーです。詳細については、IAM ドキュメントの「ロールに関する用語と概念」にあるプリンシパルを参照してください。

プライバシーバイデザイン

開発プロセス全体を通してプライバシーが考慮されているシステムエンジニアリングのアプローチ。

プライベートホストゾーン

1 つ以上の VPC 内のドメインとそのサブドメインへの DNS クエリに対し、Amazon Route 53 がどのように応答するかに関する情報を保持するコンテナ。詳細については、Route 53 ドキュメントの「プライベートホストゾーンの使用」を参照してください。

プロアクティブコントロール

非準拠リソースのデプロイ防止を目的としたセキュリティコントロール。このコントロールにより、プロビジョニング前にリソースをスキャンします。コントロールに準拠していないリソースは、プロビジョニングされません。詳細については、 AWS Control Tower ドキュメントの「コントロールリファレンスガイド」および「 セキュリティコントロールの実装」の「プロアクティブコントロール」を参照してください。 AWS

製品ライフサイクル管理 (PLM)

製品の設計、開発、発売から、成長、成熟、衰退、廃棄に至る、製品のライフサイクル全体を通してデータとプロセスを管理すること。

本番環境

「環境」を参照してください。

プログラマブルロジックコントローラー (PLC)

製造分野で使用される、信頼性と適応性に優れたコンピュータであり、これによって、マシンをモニタリングするとともに、製造プロセスを自動化します。

プロンプトチェイニング

1 つの LLM プロンプトによる出力を次のプロンプトの入力に使用して、より良いレスポンスを生成します。この手法を使用すると、複雑なタスクをサブタスクに分割したり、事前レスポンスを繰り返し改良または拡張したりできます。これによって、モデルのレスポンスの精度と関連性が向上し、粒度の高いパーソナライズされた結果を得られます。

仮名化

データセット内の個人識別子をプレースホルダー値に置き換えるプロセス。仮名化は個人のプライバシー保護に役立ちます。仮名化されたデータは、依然として個人データとみなされます。

発行/サブスクライブ (pub/sub)

マイクロサービス間の非同期通信を可能にするパターン。これにより、スケーラビリティと応答性を向上させます。例えば、マイクロサービスベースの MES の場合、マイクロサービスは、他のマイクロサービスがサブスクライブ可能なチャネルにイベントメッセージを発行できます。このシステムでは、発行サービスの変更なしに、新規マイクロサービスを追加できます。

Q

クエリプラン

手順などの一連のステップであり、SQL リレーショナルデータベースシステムのデータにアクセスするために使用されます。

クエリプランのリグレッション

データベースサービスのオプティマイザーが、データベース環境に特定の変更が加えられる前に選択されたプランよりも最適性の低いプランを選択すること。これは、統計、制限事項、環境設定、クエリパラメータのバインディングの変更、およびデータベースエンジンの更新などが原因である可能性があります。

R

RACI マトリックス

「実行責任者、説明責任者、協業先、報告先 (RACI)」を参照してください。

RAG

「検索拡張生成」を参照してください。

ランサムウェア

決済が完了するまでコンピュータシステムまたはデータへのアクセスをブロックするように設計された、悪意のあるソフトウェア。

RASCI マトリックス

「実行責任者、説明責任者、協業先、報告先 (RACI)」を参照してください。

RCAC

「行と列のアクセス制御」を参照してください。

リードレプリカ

読み取り専用に使用されるデータベースのコピー。クエリをリードレプリカにルーティングして、プライマリデータベースへの負荷を軽減できます。

リアーキテクト

「7 Rs」を参照してください。

目標復旧時点 (RPO)

最後のデータリカバリポイントからの最大許容時間です。これにより、最後の回復時点からサービスが中断されるまでの間に許容できるデータ損失の程度が決まります。

目標復旧時間 (RTO)

サービスの中断から復旧までの最大許容遅延時間。

リファクタリング

「7 Rs」を参照してください。

リージョン

地理的エリア内の AWS リソースのコレクション。各 AWS リージョン は、耐障害性、安定性、耐障害性を提供するために、他の とは独立しています。詳細については、「アカウントが使用できる AWS リージョン を指定する」を参照してください。

リグレッション

数値を予測する機械学習手法。例えば、「この家はどれくらいの値段で売れるでしょうか?」という問題を解決するために、機械学習モデルは、線形回帰モデルを使用して、この家に関する既知の事実 (平方フィートなど) に基づいて家の販売価格を予測できます。

リホスト

「7 Rs」を参照してください。

リリース

デプロイプロセスで、変更を本番環境に昇格させること。

再配置

「7 Rs」を参照してください。

リプラットフォーム

「7 Rs」を参照してください。

再購入

「7 Rs」を参照してください。

回復性

中断に抵抗または中断から回復するアプリケーションの機能。 AWS クラウドでの回復力を計画する際には、一般的に、高可用性とディザスタリカバリが考慮されます。詳細については、「AWS クラウド の耐障害性」を参照してください。

リソースベースのポリシー

Amazon S3 バケット、エンドポイント、暗号化キーなどのリソースにアタッチされたポリシー。このタイプのポリシーは、アクセスが許可されているプリンシパル、サポートされているアクション、その他の満たすべき条件を指定します。

実行責任者、説明責任者、協業先、報告先 (RACI) に基づくマトリックス

移行活動とクラウド運用に関わるすべての関係者の役割と責任を定義したマトリックス。マトリックスの名前は、マトリックスで定義されている責任の種類、すなわち責任 (R) 、説明責任 (A) 、協議 (C) 、情報提供 (I) に由来します。サポート (S) タイプはオプションです。サポートが含まれる場合は RASCI マトリックスと呼ばれ、含まれない場合は RACI マトリックスと呼ばれます。

レスポンシブコントロール

有害事象やセキュリティベースラインからの逸脱について、修復を促すように設計されたセキュリティコントロール。詳細については、「 AWSでのセキュリティコントロールの実装」の「レスポンシブコントロール」を参照してください。

保持

「7 Rs」を参照してください。

廃止

「7 Rs」を参照してください。

検索拡張生成 (RAG)

生成 AI の技術。これにより、LLM では、レスポンスの生成前に、トレーニングデータソースの外部にある信頼できるデータソースが参照されます。例えば、RAG モデルによって、組織のナレッジベースまたはカスタムデータのセマンティック検索を実行できる場合があります。細については、「RAG (検索拡張生成) とは何ですか?」を参照してください。

ローテーション

定期的にシークレット情報を更新して、攻撃者が認証情報にアクセスするのをより困難にするプロセス。

行と列のアクセス制御 (RCAC)

アクセスルールが定義された、基本的で柔軟な SQL 表現の使用。RCAC は行権限と列マスクで構成されています。

RPO

「目標復旧時点」を参照してください。

RTO

「目標復旧時間」を参照してください。

ランブック

特定のタスクを実行するために必要な手動または自動化された一連の手順。これらは通常、エラー率の高い反復操作や手順を合理化するために構築されています。

S

SAML 2.0

多くの ID プロバイダー (IdP) が使用しているオープンスタンダード。この機能を使用すると、フェデレーティッドシングルサインオン (SSO) が有効になるため、ユーザーは組織内のすべてのユーザーを IAM で作成しなくても、 にログイン AWS マネジメントコンソール したり AWS 、API オペレーションを呼び出すことができます。SAML 2.0 ベースのフェデレーションの詳細については、IAM ドキュメントの「SAML 2.0 ベースのフェデレーションについて」を参照してください。

SCADA

「監視制御とデータ取得」を参照してください。

SCP

「サービスコントロールポリシー」を参照してください。

シークレット

暗号化された形式で保存する AWS Secrets Managerパスワードやユーザー認証情報などの機密情報または制限付き情報。シークレット値とそのメタデータで構成されます。シークレット値には、バイナリ、1 つの文字列、複数の文字列を指定できます。詳細については、Secrets Manager ドキュメントの「Secrets Manager シークレットの概要」を参照してください。

セキュリティバイデザイン

開発プロセス全体を通してセキュリティが考慮されているシステムエンジニアリングのアプローチ。

セキュリティコントロール

脅威アクターによるセキュリティ脆弱性の悪用を防止、検出、軽減するための、技術上または管理上のガードレール。セキュリティコントロールには、主に 4 つの種類があります。4 つとは、予防、検出、レスポンシブ、プロアクティブです。

セキュリティ強化

アタックサーフェスを狭めて攻撃への耐性を高めるプロセス。このプロセスには、不要になったリソースの削除、最小特権を付与するセキュリティのベストプラクティスの実装、設定ファイル内の不要な機能の無効化、といったアクションが含まれています。

Security Information and Event Management (SIEM) システム

セキュリティ情報管理 (SIM) とセキュリティイベント管理 (SEM) のシステムを組み合わせたツールとサービス。SIEM システムは、サーバー、ネットワーク、デバイス、その他ソースからデータを収集、モニタリング、分析して、脅威やセキュリティ違反を検出し、アラートを発信します。

セキュリティレスポンスの自動化

セキュリティイベントへの自動レスポンスまたは自動修復を目的として、事前定義およびプログラムされたアクション。これらの自動化は、セキュリティのベストプラクティスを実装するのに役立つ検出的または応答的な AWS セキュリティコントロールとして機能します。自動レスポンスアクションの例には、VPC セキュリティグループの変更、Amazon EC2 インスタンスへのパッチ適用、認証情報の更新などがあります。

サーバー側の暗号化

送信先で、それ AWS のサービス を受け取る によるデータの暗号化。

サービスコントロールポリシー (SCP)

AWS Organizationsの組織内の、すべてのアカウントのアクセス許可を一元的に管理するポリシー。SCP は、管理者がユーザーまたはロールに委任するアクションに、ガードレールを定義したり、アクションの制限を設定したりします。SCP は、許可リストまたは拒否リストとして、許可または禁止するサービスやアクションを指定する際に使用できます。詳細については、 AWS Organizations ドキュメントの「サービスコントロールポリシー」を参照してください。

サービスエンドポイント

のエントリポイントの URL AWS のサービス。ターゲットサービスにプログラムで接続するには、エンドポイントを使用します。詳細については、「AWS 全般のリファレンス」の「AWS のサービス エンドポイント」を参照してください。

サービスレベルアグリーメント (SLA)

サービスのアップタイムやパフォーマンスなど、IT チームがお客様に提供すると約束したものを明示した合意書。

サービスレベルインジケータ (SLI)

エラー率、可用性、スループットといった、サービスパフォーマンス面の指標。

サービスレベル目標 (SLO)

サービスレベルインジケータによって測定され、サービスの状態を表すターゲットメトリクス。

責任共有モデル

クラウドのセキュリティとコンプライアンス AWS について と共有する責任を説明するモデル。AWS はクラウドのセキュリティを担当しますが、お客様はクラウドのセキュリティを担当します。詳細については、「責任共有モデル」を参照してください。

シャドウ AI

組織内の管理対象チャネルの外部で構築または使用される認可されていない AI アプリケーション。

SIEM

「Security Information and Event Management システム」を参照してください。

単一障害点 (SPOF)

特定のアプリケーションを構成する単一の重要なコンポーネントで発生し、システム稼働に支障をきたす可能性のある障害。

SLA

「サービスレベルアグリーメント」を参照してください。

SLI

「サービスレベルインジケータ」を参照してください。

SLO

「サービスレベルの目標」を参照してください。

スプリットアンドシードモデル

モダナイゼーションプロジェクトのスケーリングと加速のためのパターン。新機能と製品リリースが定義されると、コアチームは解放されて新しい製品チームを作成します。これにより、お

客様の組織の能力とサービスの拡張、デベロッパーの生産性の向上、迅速なイノベーションのサポートに役立ちます。詳細については、「AWS クラウドでのアプリケーションをモダナイズするための段階的アプローチ」を参照してください。

SPOF

「単一障害点」を参照してください。

スタースキーマ

データベースの編成構造を意味し、1 つの大きいファクトテーブルにトランザクションデータまたは測定データが保存され、1 つ以上の小さいディメンションテーブルにデータ属性が保存されます。この構造は、データウェアハウスやビジネスインテリジェンスを用途とするように設計されています。

strangler fig パターン

レガシーシステムが廃止されるまで、システム機能を段階的に書き換えて置き換えることにより、モノリシックシステムをモダナイズするアプローチ。このパターンは、宿主の樹木から根を成長させ、最終的にその宿主を包み込み、宿主に取って代わるイチジクのつるを例えています。そのパターンは、モノリシックシステムを書き換えるときのリスクを管理する方法として Martin Fowler により提唱されました。このパターンの適用方法の例については、「コンテナと Amazon API Gateway を使用して、従来の Microsoft ASP.NET (ASMX) ウェブサービスを段階的にモダナイズ」を参照してください。

サブネット

VPC 内の IP アドレスの範囲。サブネットは、1 つのアベイラビリティーゾーンに存在する必要があります。

監視制御とデータ取得 (SCADA)

製造分野において、ハードウェアとソフトウェアを使用して物理アセットと本番運用をモニタリングするシステム。

対称暗号化

データの暗号化と復号に同じキーを使用する暗号化のアルゴリズム。

合成テスト

ユーザーとのやり取りをシミュレートして、起こり得る問題を検出したり、パフォーマンスをモニタリングしたりすることで、システムをテストします。Amazon CloudWatch Synthetics を使用すると、こうしたテストを作成できます。

システムプロンプト

コンテキスト、指示、ガイドラインなどを提示して、LLM に動作を指示する手法。システムプロンプトは、コンテキストを設定して、ユーザーとやり取りするルールを確立するのに有用です。

T

タグ

AWS リソースを整理するためのメタデータとして機能するキーと値のペア。タグは、 リソースの管理、識別、整理、検索、フィルタリングに役立ちます。詳細については、「AWS リソースのタグ付け」を参照してください。

ターゲット変数

監督された機械学習でお客様が予測しようとしている値。これは、結果変数のことも指します。例えば、製造設定では、ターゲット変数が製品の欠陥である可能性があります。

タスクリスト

ランブックの進行状況を追跡するために使用されるツール。タスクリストには、ランブックの概要と完了する必要のある一般的なタスクのリストが含まれています。各一般的なタスクには、推定所要時間、所有者、進捗状況が含まれています。

テスト環境

「環境」を参照してください。

トレーニング

お客様の機械学習モデルに学習するデータを提供すること。トレーニングデータには正しい答えが含まれている必要があります。学習アルゴリズムは入力データ属性をターゲット (お客様が予測したい答え) にマッピングするトレーニングデータのパターンを検出します。これらのパターンをキャプチャする機械学習モデルを出力します。そして、お客様が機械学習モデルを使用して、ターゲットがわからない新しいデータでターゲットを予測できます。

tool

エージェントが外部システムでオペレーションを実行するために呼び出すことができる関数または API。

トランジットゲートウェイ

VPC とオンプレミスネットワークを相互接続するために使用できる、ネットワークの中継ハブ。詳細については、 AWS Transit Gateway ドキュメントの「トランジットゲートウェイとは」を参照してください。

トランクベースのワークフロー

デベロッパーが機能ブランチで機能をローカルにビルドしてテストし、その変更をメインブランチにマージするアプローチ。メインブランチはその後、開発環境、本番前環境、本番環境に合わせて順次構築されます。

信頼されたアクセス

ユーザーに代わって AWS Organizations およびそのアカウントで組織内でタスクを実行するために指定したサービスにアクセス許可を付与します。信頼されたサービスは、サービスにリンクされたロールを必要なときに各アカウントに作成し、ユーザーに代わって管理タスクを実行します。詳細については、 ドキュメントの「 を他の AWS のサービス AWS Organizations で使用する AWS Organizations 」を参照してください。

チューニング

機械学習モデルの精度を向上させるために、お客様のトレーニングプロセスの側面を変更する。例えば、お客様が機械学習モデルをトレーニングするには、ラベル付けセットを生成し、ラベルを追加します。これらのステップを、異なる設定で複数回繰り返して、モデルを最適化します。

ツーピザチーム

2 枚のピザを分け合えることができるくらい小さな DevOps チーム。ツーピザチームの規模では、ソフトウェア開発におけるコラボレーションに最適な機会が確保されます。

U

不確実性

予測機械学習モデルの信頼性を損なう可能性がある、不正確、不完全、または未知の情報を指す概念。不確実性には、次の 2 つのタイプがあります。認識論的不確実性は、限られた、不完全なデータによって引き起こされ、弁論的不確実性 は、データに固有のノイズとランダム性によって引き起こされます。

未分化なタスク

ヘビーリフティングとも呼ばれ、アプリケーションの作成と運用には必要だが、エンドユーザーに直接的な価値をもたらさなかったり、競争上の優位性をもたらしたりしない作業です。未分化なタスクの例としては、調達、メンテナンス、キャパシティプランニングなどがあります。

上位環境

「環境」を参照してください。

V

バキューミング

ストレージを再利用してパフォーマンスを向上させるために、増分更新後にクリーンアップを行うデータベースのメンテナンス操作。

バージョンコントロール

リポジトリ内のソースコードへの変更など、変更を追跡するプロセスとツール。

VPC ピアリング

プライベート IP アドレスを使用してトラフィックをルーティングできる、2 つの VPC 間の接続。詳細については、Amazon VPC ドキュメントの「VPC ピア機能とは」を参照してください。

脆弱性

システムのセキュリティを脅かすソフトウェアまたはハードウェアの欠陥。

W

ウォームキャッシュ

頻繁にアクセスされる最新の関連データを含むバッファキャッシュ。データベースインスタンスはバッファキャッシュから、メインメモリまたはディスクからよりも短い時間で読み取りを行うことができます。

ウォームデータ

アクセス頻度の低いデータ。この種類のデータをクエリする場合、通常は適度に遅いクエリでも問題ありません。

ウィンドウ関数

現在のレコードに何らかの形で関連している行のグループに計算を実行する SQL 関数。ウィンドウ関数は、移動平均を計算したり、現在の行の相対位置に基づいて他の行の値にアクセスするといったタスクの処理に役立ちます。

ワークロード

ビジネス価値をもたらすリソースとコード (顧客向けアプリケーションやバックエンドプロセスなど) の総称。

ワークストリーム

特定のタスクセットを担当する移行プロジェクト内の機能グループ。各ワークストリームは独立していますが、プロジェクト内の他のワークストリームをサポートしています。たとえば、ポートフォリオワークストリームは、アプリケーションの優先順位付け、ウェーブ計画、および移行メタデータの収集を担当します。ポートフォリオワークストリームは、これらの設備を移行ワークストリームで実現し、サーバーとアプリケーションを移行します。

WORM

「Write-Once-Read-Many」を参照してください。

WQF

「AWS ワークロード資格フレームワーク」を参照してください

Write-Once-Read-Many (WORM)

データを 1 回のみ書き込むことで、データの削除や変更を防ぐストレージモデル。承認済みユーザーは、必要な回数だけデータを読み取ることができますが、変更することはできません。このデータストレージインフラストラクチャは、イミュータブルと見なされます。

Z

ゼロデイエクスプロイト

ゼロデイ脆弱性を悪用した攻撃 (一般的にマルウェアによる)。

ゼロデイ脆弱性

実稼働システムにおける未解決の欠陥または脆弱性。脅威アクターは、このような脆弱性を利用してシステムを攻撃する可能性があります。開発者は、よく攻撃の結果で脆弱性に気付きます。

ゼロショットプロンプト

LLM にタスク実行の手順は提示するが、実行のガイドとして役立つ例 (ショット) は提示しない方法。LLM は、事前トレーニング済みの知識を使用してタスクを処理する必要があります。ゼロショットプロンプトの有効性は、タスクの複雑さとプロンプトの品質によって異なります。「数ショットプロンプト」も参照してください。

ゾンビアプリケーション

平均 CPU およびメモリ使用率が 5% 未満のアプリケーション。移行プロジェクトでは、これらのアプリケーションを廃止するのが一般的です。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。