AWS Well-Architected フレームワーク

セキュリティの柱

セキュリティの柱: AWS Well-Architected フレームワーク

Table of Contents

セキュリティの柱 - AWS Well-Architected フレームワーク

発行日: 2024 年 11 月 6 日 (ドキュメントの改訂)

このホワイトペーパーは、AWS Well-Architected フレームワークのセキュリティの柱に焦点を当てています。お客様が安全な AWS ワークロードの設計、配信、メンテナンスにベストプラクティスと最新のレコメンデーションを適用する上で役立つガイダンスを提供します。

序章

AWS Well-Architected フレームワークは、AWS でワークロードを構築する際の決定事項のトレードオフを理解するのに役立ちます。このフレームワークを使用することで、信頼性、安全性、効率性、コスト効率に優れ、持続可能なワークロードをクラウド内で設計および運用するための、アーキテクチャ上の最新のベストプラクティスを学ぶことができます。このフレームワークにより、ワークロードをベストプラクティスに照らし合わせて一貫して測定し、改善点を特定することができます。私たちは、well-architected ワークロードを設計することで、ビジネスの成功の可能性が大幅に高まると考えています。

このフレームワークは次の 6 つの柱に基づいています。

- 運用上の優秀性
- セキュリティ
- 信頼性
- パフォーマンス効率
- コスト最適化
- 持続可能性

このホワイトペーパーは、セキュリティの柱に焦点を当てています。現在の AWS レコメンデーションに従うことで、お客様のビジネスやレコメンデーションを満たすことに役立ちます。このホワイトペーパーは、最高技術責任者 (CTO)、最高情報セキュリティ責任者 (CSO/CISO)、アーキテクト、開発者、オペレーションチームメンバーなどの技術担当者を対象としています。

このホワイトペーパーを読むことで、AWS の最新のレコメンデーションと戦略を理解し、セキュリティを念頭に置いてクラウドアーキテクチャを設計することができます。このホワイトペーパーでは、実装の詳細やアーキテクチャのパターンについては説明していませんが、そのような情報を含むリソースへの参照を記載しています。このホワイトペーパーにある手法を採用することで、データと

システムを保護し、アクセスを制御し、セキュリティイベントに自動的に対応するアーキテクチャを構築できます。

セキュリティ基盤

セキュリティの柱では、クラウドテクノロジーを活用して、ユーザーのセキュリティ体制を向上させる方法でデータ、システム、アセットを保護する方法について説明します。このホワイトペーパーでは、AWS で安全なワークロードを設計するための詳細なベストプラクティスガイダンスを提供します。

設計原則

クラウドには、ワークロードのセキュリティ強化に役立つ多くの原則があります:

- 強力なアイデンティティ基盤を実装する: 最小特権の原則を実装し、お客様の AWS リソースのやり取りごとに適切な承認を得て、職務の分離を強制します。アイデンティティ管理を一元化し、長期的な静的認証情報への依存を排除することを目指します。
- トレーサビリティの維持: 環境に対して、リアルタイムでモニタリング、アラート、監査のアクションと変更を行います。ログとメトリクスの収集をシステムと統合して、自動的に調査してアクションを実行します。
- すべての層にセキュリティを適用する: 複数のセキュリティコントロールを使用して、詳細な防御アプローチを適用します。すべての層に適用します (ネットワークのエッジ、VPC、ロードバランシング、すべてのインスタンスとコンピューティングサービス、オペレーティングシステム、アプリケーション、コードなど)。
- セキュリティのベストプラクティスの自動化: 自動化されたソフトウェアベースのセキュリティメカニズムにより、迅速かつコスト効果に優れた方法で安全にスケールできます。バージョン管理されたテンプレートのコードとして定義および管理されるコントロールの実装を含む、安全なアーキテクチャを作成します。
- 転送中のデータおよび保管中のデータの保護: データを機密レベルに分類し、必要に応じて暗号化、トークン化、アクセス制御などのメカニズムを使用します。
- 人をデータから遠ざける: メカニズムとツールを使用して、データに直接アクセスしたり、手動でデータを処理したりする必要性を軽減または排除します。これにより、機密データを扱う際の誤処理や変更、人的ミスのリスクが軽減されます。
- セキュリティイベントの準備: 組織の要件に合わせたインシデント管理および調査のポリシーとプロセスを導入し、インシデントに備えます。インシデント対応シミュレーションを実行し、ツールと自動化により、検出、調査、復旧のスピードを上げます。

定義

クラウドのセキュリティには、次の 7 つの領域があります。

- セキュリティ基盤
- Identity and Access Management
- 検出
- インフラストラクチャの保護
- データ保護
- インシデントへの対応
- アプリケーションのセキュリティ

責任共有

セキュリティとコンプライアンスに関して、AWS とお客様の間で責任を共有します。この共有モデルにより、ホストオペレーティングシステムや仮想化レイヤーからサービスが運用されている施設の物理的なセキュリティに至るまで、さまざまなコンポーネントを AWS が運用、管理、制御するため、お客様の運用の負担が軽減されます。お客様は、AWS が提供するセキュリティグループのファイアウォール設定に加えて、ゲストオペレーティングシステム (更新やセキュリティパッチを含む) およびその他の関連アプリケーションソフトウェアを管理し、責任を負うものとします。お客様の責任範囲は、使用するサービス、IT 環境へのサービス統合、適用される法規制に応じて異なります。このため、お客様は選択するサービスを注意深く検討する必要があります。この責任共有モデルの性質によって柔軟性が得られ、お客様はデプロイを統制できます。下図に示すように、この責任分担は一般的に、クラウド「の」セキュリティと、クラウド「内の」セキュリティと呼ばれます。

AWS の責任「クラウドのセキュリティ」 – AWS クラウドで提供されるすべてのサービスを実行するインフラストラクチャの保護に関しては、AWS が責任を負います。このインフラストラクチャは、AWS クラウドサービスを実行するハードウェア、ソフトウェア、ネットワーク、施設で構成されます。

お客様の責任「クラウド内のセキュリティ」 – お客様の責任は、お客様が選択する AWS クラウドサービスによって決まります。これにより、お客様がセキュリティの責任の一部として実行する必要がある設定作業の量が決まります。例えば、Amazon Elastic Compute Cloud (Amazon EC2) などのサービスは Infrastructure as a Service (IaaS) に分類されるため、お客様は必要なセキュリティ設定と管理タスクをすべて実行する必要があります。Amazon EC2 インスタンスをデプロイするお客様は、ゲストオペレーティングシステムの管理 (更新やセキュリティパッチの適用を含む)、お客様が各

インスタンスにインストールするアプリケーションソフトウェアやユーティリティの管理、AWS が提供する各インスタンスのファイアウォール (セキュリティグループ) の設定について責任を負います。Amazon S3 や Amazon DynamoDB などの抽象化されたサービスについては、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、プラットフォームの運用を AWS が行い、お客様はエンドポイントにアクセスしてデータを保存、取得します。お客様は、データの管理 (暗号化オプションを含む)、アセットの分類、および IAM ツールを使用した適切なアクセス許可の適用について責任を負います。

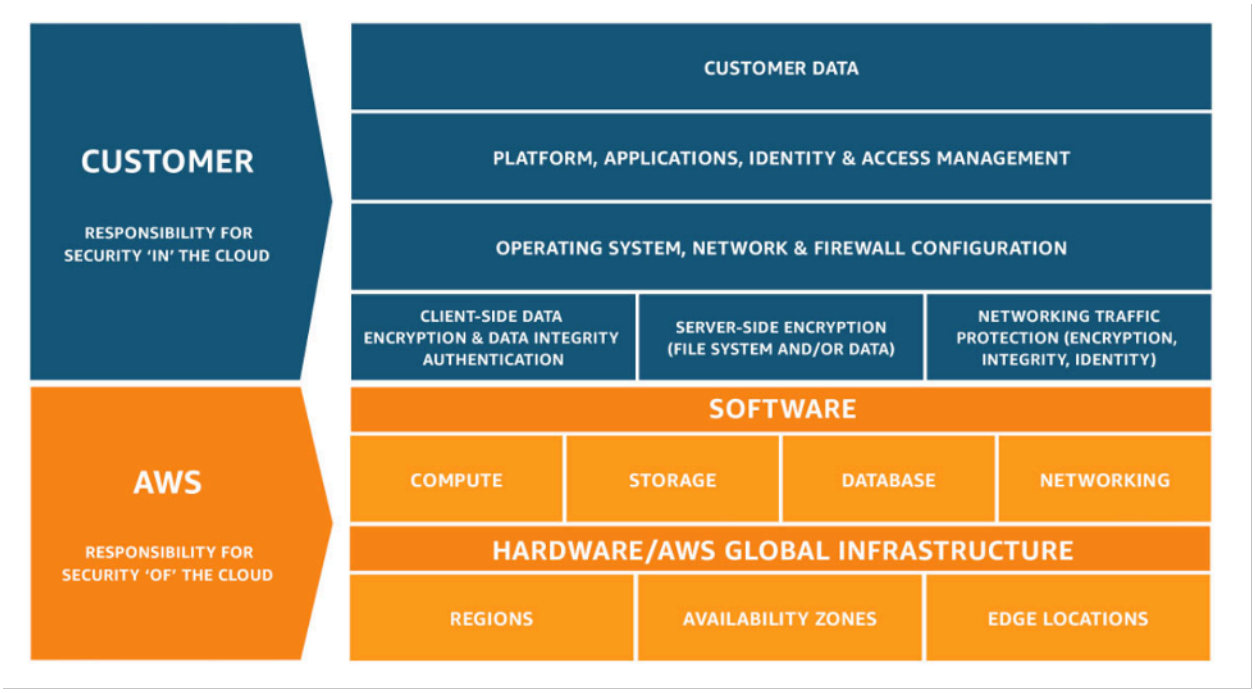


図 1: AWS の責任共有モデル。

このお客様/AWS の責任分担モデルは、IT 統制にも拡大されます。IT 環境を運用する責任がお客様と AWS との間で分担されるのと同様に、IT 統制の管理、運用、検証も分担されます。AWS は、これまでお客様が管理していた AWS 環境にデプロイされた物理インフラストラクチャに関連する統制を管理することで、お客様の運用管理の負担を軽減します。お客様によって AWS のデプロイ方法は異なるため、特定の IT 統制の管理を AWS に移行することで、(新たな)統制環境の分散化を図るというメリットが得られます。その後、お客様は利用可能な AWS の統制とコンプライアンス文書を使用し、必要に応じて統制の評価および検証手順を実行することができます。以下は、AWS と AWS のお客様、またはその両方によって管理される統制の例です。

継承された統制 – お客様が AWS から完全に継承する統制。

- 物理統制と環境統制

共有された統制 – インフラストラクチャレイヤーとお客様のレイヤーの両方に適用される統制。ただしコンテキストまたは観点は異なります。共有統制では、AWS はインフラストラクチャの要件を提供し、お客様は AWS サービスの使用中に独自の統制の実装を提供する必要があります。以下に例を示します。

- パッチ管理 – AWS はインフラストラクチャ内の欠陥のパッチ適用と修正に責任を負いますが、ゲストオペレーティングシステムとアプリケーションへのパッチ適用はお客様が責任を負います。
- 設定管理 – AWS はインフラストラクチャデバイスの設定を維持しますが、ゲストオペレーティングシステム、データベース、アプリケーションの設定はお客様が行います。
- 意識向上およびトレーニング – AWS は AWS の従業員にトレーニングを提供しますが、お客様は自社の従業員をトレーニングする必要があります。

お客様固有 – AWS サービス内にデプロイしているアプリケーションに基づいて、お客様が単独で責任を負う統制。以下に例を示します。

- サービスと通信の保護、またはゾーンセキュリティでは、お客様が特定のセキュリティ環境下で、データのルーティングやゾーニングを行わなければならない場合があります。

ガバナンス

セキュリティガバナンスは、全体的なアプローチのサブセットとしてリスク管理を支援するためのポリシーと管理目標を定義することにより、ビジネス目標をサポートすることを目的としています。セキュリティ管理目標に対して階層的アプローチ (各レイヤーを前のレイヤーの上に構築する) を取ることにより、リスク管理を達成します。AWS 共有責任モデルを理解することが、基礎のレイヤーとなります。この知識により、お客様側の責任は何か、AWS から何を継承するかが明確になります。有益なリソースは AWS アーティファクトです。これを使用すると、AWS のセキュリティとコンプライアンスのレポート、および一部のオンライン契約にオンデマンドでアクセスできます。

コントロールの目的のほとんどは、次のレイヤーで満たします。プラットフォーム全体の機能が備わっているのはこちらになります。例えば、このレイヤーには AWS アカウントの販売プロセス、AWS IAM アイデンティティセンター などの ID プロバイダーとの統合、共通の検出制御などが含まれます。プラットフォームガバナンスプロセスのアウトプットの一部もこちらにあります。新しい AWS のサービスを使用して開始する場合は、AWS Organizations サービスでサービスコントロールポリシー (SCP) を更新し、サービスの初期使用のためのガードレールを提供します。他の SCP を使用して、一般的にセキュリティ不変条件と呼ばれる共通のセキュリティ制御目標を実装できます。これらは、複数のアカウント、組織単位、または AWS 組織全体に適用する管理目標または設定です。典型的な例としては、インフラストラクチャが実行されるリージョンを制限したり、検出コント

ロールの無効化を防いだりすることが挙げられます。この中間レイヤーには、設定ルールやパイプラインのチェックなど、体系化されたポリシーも含まれます。

最上位のレイヤーは、製品チームが管理目標を達成する場所です。これは、製品チームが管理するアプリケーションで実装が行われるためです。これには、アプリケーションで入力確認を実装することや、マイクロサービス間で ID が正しく受け渡しされるのを確認することなどが考えられます。製品チームが設定を担当しますが、中間レイヤーから一部の機能を継承できます。

統制を実装する場所がどこであろうと目的は同じです。すなわち「リスク管理」です。一連のリスク管理フレームワークは、特定の業界、リージョン、またはテクノロジーに適用されます。主な目標は「可能性と結果に基づいてリスクを強調すること」です。これが固有のリスクです。次に、可能性、結果、またはその両方を軽減する管理目標を定義します。続いて、管理策を実施することで、結果としてどのようなリスクが生じるのかを確認できます。これは残差リスクです。管理目標は、1 つ以上のワークロードに適用できます。次の図は、典型的なリスクマトリックスを示しています。可能性は過去の発生頻度に基づき、結果はイベントの金銭的、風評的、時間的コストに基づいています。

Likelihood	Risk Level				
Very Likely	Low	Medium	High	Critical	Critical
Likely	Low	Medium	Medium	High	Critical
Possible	Low	Low	Medium	Medium	High
Unlikely	Low	Low	Medium	Medium	High
Very unlikely	Low	Low		Medium	High
Consequence	Minimal	Low	Medium	High	Severe

図 2: リスクレベルの可能性マトリックス

AWS アカウントの管理と分離

当社では、組織のレポート構造を流用せずに、個別アカウントごとにワークロードを整理し、機能、コンプライアンス要件、共通のコントロールセットに基づいてアカウントをグループ化することを推

奨しています。AWS では、アカウントが強固な境界となります。例えば、開発およびテストのワークロードと本番稼働ワークロードを切り離すために、アカウントレベルの分離を強くお勧めします。

アカウントを一元管理する: AWS Organizations は、AWS アカウントの作成と管理、および作成後のアカウントの制御を自動化します。AWS Organizations を使用してアカウントを作成する場合、使用する E メールアドレスを検討することが重要です。なぜならば、E メールアドレスがパスワードのリセットを許可するルートユーザーとなるためです。組織では、アカウントを組織単位 (OU) にグループ化し、ワークロードの要件と目的に基づいてさまざまな環境を表すことができます。

制御を一元的に設定する: 適切なレベルで特定のサービス、リージョン、サービスアクションのみを許可することで、AWS アカウントで実行できる内容を制御します。AWS Organizations では、サービスコントロールポリシー (SCP) を使用し、組織、組織単位、またはアカウント レベルで、すべての AWS Identity and Access Management (IAM) ユーザーとロールに適用する、アクセス許可ガードレールを適用できます。例えば、SCP を適用して、明示的に許可されていないリージョン内のユーザーがリソースを起動することを制限できます。AWS Control Tower では、複数アカウントの効率的な設定と管理が可能です。これにより、AWS 組織内のアカウントの設定、プロビジョニングを自動化し、ガードレール (予防と検出を含む) を適用し、可視性のためのダッシュボードを提供します。

サービスとリソースを一元的に設定する: AWS Organizations は、すべてのアカウントに適用される AWSのサービスを設定するのに役立ちます。例えば、AWS CloudTrail を使用して、組織全体で実行されるすべてのアクションの中央ロギングを設定し、メンバーアカウントがロギングを無効化しないようにします。また、AWS Config を使用して、定義したルールのデータを一元的に集約することもできるため、ワークロードのコンプライアンスを監査して、変更に迅速に対応できます。AWS CloudFormationStackSets を使用すると、組織内の複数のアカウントと OU にまたがる AWS CloudFormation スタックを一元管理できます。これにより、新しいアカウントを自動的にプロビジョニングして、セキュリティ要件を満たすことができます。

セキュリティサービスの委任管理機能を使用して、管理に使用されるアカウントを組織の請求 (管理) アカウントから分離します。GuardDuty、Security Hub、AWS Config などのいくつかの AWS のサービスでは、管理機能に特定のアカウントを指定するなど、AWS Organizations との統合をサポートしています。

ベストプラクティス

- SEC01-BP01 アカウントを使用してワークロードを分ける:
- SEC01-BP02 セキュアアカウントのルートユーザーおよびプロパティ

SEC01-BP01 アカウントを使用してワークロードを分ける:

マルチアカウント戦略を取り、環境 (本番稼働、開発、テストなど) とワークロードの間に共通ガードレールを構成し、分離を確立します。アカウントレベルの分類は、セキュリティ、請求、アクセスのために強力な分離境界を提供するため、強く推奨されます。

期待される成果: クラウドオペレーション、関連しないワークロード、環境を別々のアカウントに分離し、クラウドインフラストラクチャ全体のセキュリティを強化するアカウント構造。

一般的なアンチパターン:

- データ重要度レベルの異なる複数の無関係のワークロードを同一アカウントに配置する。
- きちんと定義されていない組織単位 (OU) 構造。

このベストプラクティスを活用するメリット:

- 誤ってワークロードにアクセスした場合の影響範囲を軽減。
- AWS サービス、リソース、およびリージョンへのアクセスの一元的ガバナンス。
- ポリシーとセキュリティサービスの一元管理により、クラウドインフラストラクチャのセキュリティを維持する。
- アカウント作成とメンテナンスプロセスの自動化。
- コンプライアンスや規制要件に対応した、インフラストラクチャの集中監査。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

AWS アカウントは、さまざまなデータ重要度レベルで稼働するワークロードまたはリソース間にセキュリティ分離境界を提供します。AWS は、マルチアカウント戦略を通して大規模にクラウドワークロードを管理し、この分離境界を活用するためのツールを提供します。AWS でのマルチアカウント戦略の概念、パターン、実装に関するガイダンスについては、「複数のアカウントで AWS 環境を構成する」を参照してください。

一元管理下に複数の AWS アカウントがある場合、アカウントを組織単位 (OU) の層によって定義された階層に組織化する必要があります。次に、OU とメンバーアカウントに対してセキュリティ管理を組織化して適用することにより、組織内のメンバーアカウントに対して一貫性のある予防的制御を

確立できます。セキュリティ管理は継承されるため、OU 階層の下位レベルにあるメンバーアカウントに対するアクセス許可をフィルタリングすることができます。優れた設計では継承を利用して、各メンバーアカウントに対して望ましいセキュリティ管理を達成するのに必要なセキュリティポリシーの件数と複雑性を軽減します。

AWS Organizations と AWS Control Tower は、AWS 環境でマルチアカウント構造を実装および管理するサービスです。AWS Organizations では、アカウントを OU の 1 つ以上のレイヤーで定義された階層に整理できます。各 OU には複数のメンバーアカウントが含まれます。サービスコントロールポリシー (SCP) を使用すると、組織管理者はメンバーアカウントに対してきめ細かな予防的コントロールを確立できます。AWS Config を使用してメンバーアカウントに対してプロアクティブコントロールと検出コントロールを確立できます。多くの AWS サービスは AWS Organizations と統合して、委任された管理コントロールを提供し、組織内のすべてのメンバーアカウントでサービス固有のタスクを実行します。

AWS Organizations の上にレイヤー化される AWS Control Tower は、ランディングゾーンを持つマルチアカウント AWS 環境のワンクリックのベストプラクティス設定を提供します。ランディングゾーンは、Control Tower によって確立されるマルチアカウント環境への入口です。Control Tower には、AWS Organizations よりもいくつかの利点があります。アカウントガバナンスを改善する 3 つの利点には次のようなものがあります。

- 組織に対して承認されたアカウントに自動適用される、統合された必須のセキュリティコントロール。
- 所定の OU セットに対してオン/オフと切り替えられるオプションのコントロール。
- AWS Control Tower Account Factory は、事前承認されたベースラインと設定オプションを含むアカウントを組織内に自動デプロイします。

実装手順

1. 組織単位構造の設計: 適切に設計された組織単位構造により、サービスコントロールポリシーやその他のセキュリティコントロールの作成と維持に必要な管理負担が軽減されます。組織単位の構造は、ビジネスニーズ、データ機密性、ワークロード構造と整合させる必要があります。
2. マルチアカウント環境のランディングゾーンを作成する: ランディングゾーンは、組織がワークロードを迅速に開発、起動、デプロイできる一貫したセキュリティとインフラストラクチャの基盤を提供します。カスタム構築のランディングゾーンまたは AWS Control Tower を使用して、環境をオーケストレーションできます。
3. ガードレールの確立: ランディングゾーンを通して環境に一貫したセキュリティガードレールを実装します。AWS Control Tower には、デプロイできる一連の必須およびオプションのコントロー

ルが用意されています。必須コントロールは、Control Tower 実装時に自動的にデプロイされます。強く推奨されたコントロールとオプションのコントロールのリストを確認し、ニーズに適したコントロールを実装します。

4. 新しく追加されたリージョンへのアクセスを制限する: 新しい AWS リージョンについて、ユーザーやロールなどの IAM リソースは、有効にしたリージョンのみに伝播されます。このアクションは、Control Tower を使用する場合はコンソールから、または AWS Organizations の IAM アクセス権限ポリシーを調整することで実行できます。
5. AWS CloudFormation StackSets を検討する: StackSets を使用すると、IAM ポリシー、ロール、グループなどのリソースをさまざまな AWS アカウントとリージョンに承認されたテンプレートからデプロイできます。

リソース

関連するベストプラクティス:

- SEC02-BP04 一元化された ID プロバイダーを利用する

関連ドキュメント:

- AWS Control Tower
- AWS セキュリティ監査のガイドライン
- IAM のベストプラクティス
- CloudFormation StackSets を利用した複数の AWS アカウントやリージョンを横断したリソース展開
- Organizations に関するよくある質問
- AWS Organizations の用語と概念
- マルチアカウント環境における AWS Organizations サービスコントロールポリシーのベストプラクティス
- AWS アカウント管理リファレンスガイド
- 複数のアカウントで AWS 環境を構成する

関連動画:

- Enable AWS adoption at scale with automation and governance

- Security Best Practices the Well-Architected Way
- AWS Control Tower を使用した複数アカウントのビルドと管理
- 既存の組織で Control Tower を有効にする

SEC01-BP02 セキュアアカウントのルートユーザーおよびプロパティ

ルートユーザーは AWS アカウントで最も権限が高いユーザーであり、アカウント内の全リソースに対する完全な管理者アクセスがあるだけでなく、場合によってはセキュリティポリシーによる制限の対象外となります。ルートユーザーへのプログラムによるアクセスを無効化し、ルートユーザーに対する適切なコントロールを確立し、さらにルートユーザーの定期的使用を避けることにより、ルート認証情報を不用意に曝露するリスク、それによるクラウド環境の侵害を軽減することができます。

期待される成果: ルートユーザーのセキュリティを保護することで、ルートユーザーの認証情報が不正利用された場合に付帯的または意図的な損害が発生する可能性を抑えることができます。検出コントロールを確立することによっても、ルートユーザーを使ったアクションが取られると適切な担当者にアラートを送信できます。

一般的なアンチパターン:

- ルートユーザー認証情報を必要とする少数以外のタスクに対してもルートユーザーを使用する。
- 緊急時に重要なインフラストラクチャ、プロセス、担当者が正常に機能するかどうかを検証するために、定期的な緊急時対応計画のテストを怠っている。
- 典型的なアカウントログインフローのみを考慮し、代替アカウント回復方法を考慮することも、テストすることもしていない。
- DNS、E メールサーバー、および携帯電話会社がアカウント復旧フローで使用されるにもかかわらず、重要なセキュリティ境界の一部として対処していない。

このベストプラクティスを活用するメリット: ルートユーザーへのアクセスを確保することで、アカウントで行われるアクションは制御され監査されているという安心感が得られます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

AWS は、アカウントを保護するのに役立つ多くのツールを提供しています。ただし、これらの対策の一部は既定では有効になっていないため、実装するには直接的な措置を講じる必要があります。こ

れらの推奨事項を、AWS アカウントをセキュリティ保護するための基本的なステップと考えてください。これらのステップを実装する際、セキュリティ管理を継続的に評価およびモニタリングすることが重要となります。

AWS アカウントを初めて作成する際は、アカウント内のすべての AWS のサービスとリソースに完全なアクセス許可を持つ 1 つの ID から始めます。この ID は、AWS アカウントのルートユーザーと呼ばれます。アカウントの作成に使用したメールアドレスとパスワードを使用して、ルートユーザーとしてサインインできます。AWS ルートユーザーに付与されるアクセス権が昇格しているため、特にそれが必要となるタスクを実行する AWS ルートユーザーの使用は、制限する必要があります。ルートユーザーのログイン認証情報は厳重に保護し、AWS アカウントルートユーザーには必ず多要素認証 (MFA) を使用します。

ユーザー名、パスワード、多要素認証 (MFA) デバイスを使用してルートユーザーにログインする通常の認証フローに加えて、アカウントに関連付けられた E メールアドレスと電話番号にアクセスし、AWS アカウントルートユーザーにログインするためのアカウント復旧フローもあります。そのため、復旧メールを送信するルートユーザーの E メールアカウントと、そのアカウントに関連する電話番号をセキュリティ保護することも同程度に重要となります。また、ルートユーザーに関連付けられた E メールアドレスが、同じ AWS アカウントの E メールサーバーやドメインネームサービス (DNS) リソースでホストされている場合、潜在的な循環依存性についても考慮する必要があります。

AWS Organizations を使用する場合、それぞれにルートユーザーが含まれる AWS アカウントが複数あります。1 つのアカウントを管理アカウントに指定し、その管理アカウントの下に何層ものメンバーアカウントを追加することができます。管理アカウントのルートユーザーのセキュリティ保護を優先してから、メンバーアカウントのルートユーザーに対処してください。管理アカウントのルートユーザーをセキュリティ保護する戦略は、メンバーアカウントのルートユーザーとは異なり、メンバーアカウントのルートユーザーに対しては予防的なセキュリティコントロールを講じることができます。

実装手順

ルートユーザーのコントロールを確立するには、次の実装ステップが推奨されます。該当する場合、推奨事項は CIS AWS Foundations benchmark version 1.4.0 に対応します。AWS アカウントとリソースのセキュリティを保護するときは、これらのステップに加えて、AWS のベストプラクティスのガイドラインも参照してください。

予防的コントロール

1. アカウントの正確な連絡先情報を設定します。

a. この情報は、紛失したパスワードの復旧フロー、紛失した MFA デバイスアカウントの復旧フロー、およびチームとの重要なセキュリティ関連のコミュニケーションに使用されます。

b. 企業ドメインによってホストされた E メールアドレスを使用します (ルートユーザーの E メールアドレスとしては、できれば配布リストのほうが望ましい)。個人の E メールアカウントではなく配布リストを使うことにより、長期的にはルートアカウントへのアクセスに対して冗長性と継続性を追加することになります。

c. 連絡先情報に記載された電話番号は、この目的専用の安全なものである必要があります。この電話番号をどこかに記載したり、誰かと共有したりしないでください。

2. ルートユーザーにはアクセスキーを作成しないでください。アクセスキーが存在する場合は、それを削除します (CIS 1.4)。

a. ルートユーザーに対する長期保存可能なプログラム認証情報 (アクセスキーとシークレットキー) は排除します。

b. ルートユーザーのアクセスキーが既にある場合は、そのキーを使用するプロセスを移行させて、AWS Identity and Access Management (IAM) ロールの一時的なアクセスキーを使用した後、ルートユーザーのアクセスキーを削除する必要があります。

3. ルートユーザーの認証情報を保管する必要があるかどうかを決定します。

a. AWS Organizations を使用して新しいアカウントを作成している場合、新規メンバーアカウントのルートユーザーの初期パスワードはランダムな値に設定され、決して公開されることはありません。必要に応じてメンバーアカウントへのアクセス権を取得するときは、AWS Organization 管理アカウントのパスワードリセットフローを使用することを検討します。

b. スタンドアロン AWS アカウントまたは管理 AWS Organization アカウントに対しては、ルートユーザーの認証情報を作成して安全に保管することを検討してください。ルートユーザーに MFA を使用します。

4. AWS マルチアカウント環境のメンバーアカウントのルートユーザーには、予防的コントロールを使用します。

a. メンバーアカウントには、ルートユーザー向けのルートアクセスキーの作成を拒否する予防的ガードレールの使用を検討します。

b. メンバーアカウントには、ルートユーザーとしてのアクションを拒否する予防的ガードレールの使用を検討します。

5. ルートユーザーの認証情報が必要な場合:

a. 複雑なパスワードを使用します。

b. ルートユーザー、特に AWS Organizations 管理 (支払者) アカウント (CIS 1.5) に対しては多要素認証 (MFA) を有効化します。

c. 回復力とセキュリティのために、ハードウェア MFA デバイスを検討してください。これは、単回使用デバイスを使用することにより、MFA コードを含むデバイスが他の目的に再使用される可能性が少なくなるためです。電池式のハードウェア MFA デバイスが定期的に交換されていることを検証してください。(CIS 1.6)

- ルートユーザーに対して MFA を設定するときは、仮想 MFA またはハードウェア MFA デバイスのいずれかを作成する手順に従います。

d. バックアップ用に複数の MFA デバイスを登録することを検討します。MFA デバイスは 1 アカウントにつき 8 台まで登録できます。

- ルートユーザーに対して複数の MFA デバイスを登録すると、MFA デバイス紛失時にアカウントを復旧するフローが自動的に無効になります。

e. パスワードは安全に保管し、電子的にパスワードを保管する際は循環依存関係を検討してください。入手するために同じ AWS アカウントへのアクセスが必要となる方法でパスワードを保管しないでください。

6. オプション: ルートユーザーに対して定期的なパスワードローテーションスケジュールを設定することを検討します。

- 認証情報管理のベストプラクティスは、規制およびポリシー要件によって異なります。MFA によって保護されるルートユーザーは、認証の単一要素としてパスワードに依存しません。
- ルートユーザーのパスワードを定期的に変更することで、意図せず漏洩したパスワードが不正利用されるリスクを減らせます。

発見的コントロール

- ルート認証情報の使用を検出するアラームを作成します (CIS 1.7)。Amazon GuardDuty を使用すると、RootCredentialUsage の検出結果を使ってルートユーザー API 認証情報の使用状況をモニタリングしたり、アラートを発したりすることができます。
- AWS Config 用の AWS Well-Architected セキュリティの柱コンフォーマンスパックに含まれる検出管理を、評価し実装します。AWS Control Tower を使用している場合は、Control Tower 内で提供されている強く推奨されるコントロールを評価し実装します。

運用ガイダンス

- 組織で、ルートユーザー認証情報へのアクセスが必要な担当者を決定します。
  - 1 人の担当者がすべての必要な認証情報とルートユーザーアクセスを取得するために MFA にアクセスするのを回避するため、2 人制を採用します。

  - アカウントに関連付けられた電話番号と E メールエイリアス (パスワードリセットと MFA リセットフローに使用される) は、個人ではなく、組織が管理するよう徹底してください。
- ルートユーザーは例外的にのみ使用します (CIS 1.7)。
  - AWS のルートユーザーを、たとえ運営業務であっても日常的なタスクに使用してはなりません。ルートユーザーを必要とする AWS タスクを実行するときは、必ずルートユーザーとしてログインします。その他すべてのアクションは、適切なロールを持つ他のユーザーが実行しなければなりません。
- ルートユーザーにアクセスできることを定期的にチェックし、ルートユーザー認証情報を使用する必要がある緊急事態の前に手順をテストしておきます。
- アカウントに関連付けられている E メールアドレスと代理連絡先に記載されている E メールアドレスが有効であることを定期的にチェックします。これらの E メールの受信箱に、セキュリティに関する通知が <abuse@amazon.com> から届いていないか監視します。また、アカウントに関連付けられた電話番号があれば、それが通じることも確認してください。
- ルートアカウントの不正使用に対処するインシデント対応手順を準備しておきます。AWS アカウントに対するインシデント対応戦略策定に関する詳細は、「AWS セキュリティインシデント対応ガイド」と、ホワイトペーパー「セキュリティの柱」の「インシデント対応」セクションにあるベストプラクティスを参照してください。

リソース

関連するベストプラクティス:

- SEC01-BP01 アカウントを使用してワークロードを分ける:
- SEC02-BP01 強力なサインインメカニズムを使用する
- SEC03-BP02 最小特権のアクセスを付与します
- SEC03-BP03 緊急アクセスのプロセスを確立する
- SEC10-BP05 アクセスを事前プロビジョニングする

関連ドキュメント:

- AWS Control Tower
- AWS セキュリティ監査のガイドライン
- IAM のベストプラクティス
- Amazon GuardDuty – root credential usage alert

- CloudTrail を使用してルート認証情報の使用状況をモニタリングするためのステップバイステップガイド
- での使用が承認された MFA トークンAWS
- AWS に Break Glass アクセスを実装する
- のセキュリティを改善するための 10 個の項目AWS アカウント
- 自分の AWS アカウントで不正なアクティビティに気づいた場合はどうすればよいですか?

関連動画:

- Enable AWS adoption at scale with automation and governance
- Security Best Practices the Well-Architected Way
- Limiting use of AWS root credentials from AWS re:inforce 2022 – Security best practices with AWS IAM

ワークロードを安全に運用する

ワークロードの安全な運用は、設計から、ビルド、実行、継続的改善まで、ワークロードのライフサイクル全体が対象になります。クラウドで安全に運営する能力を改善する方法の 1 つは、ガバナンスに組織的アプローチを取り入れることです。ガバナンスとは、関与する担当者の適切な判断のみに依存することなく、意思決定が一貫して導かれる方法です。ガバナンスモデルとプロセスとは、「特定のワークロードの管理目標が満たされ、そのワークロードに適切であることをどのように確認すればよいか?」という質問に答える方法です。意思決定へのアプローチが一貫していると、ワークロードのデプロイが加速し、組織内のセキュリティ機能の水準の向上に役立ちます。

ワークロードを安全に運用するためには、セキュリティのすべての領域に包括的なベストプラクティスを適用する必要があります。「運用上の優秀性」で定義された要件とプロセスを組織やワークロードのレベルで作成し、あらゆる領域に適用します。AWS や業界の推奨事項と脅威インテリジェンスを最新の状態に保つことで、脅威モデルを進化させ、目標を制御できます。セキュリティプロセス、テスト、検証を自動化することで、セキュリティ運用の規模を拡大することができます。

自動化は、プロセスの一貫性と再現性を可能にします。人は多くのことに長けていますが、同じことをミスなく一貫して繰り返し行うことは、人の得意なことではありません。きちんとしたランブックを作成しても、繰り返し行うべき作業を一貫して行うことができないというリスクがあります。特に、担当業務が多岐にわたり、不慣れなアラートに対応しなければならない場合は、その傾向が顕著になります。しかし、自動化は毎回同じように反応します。アプリケーションをデプロイする最良

の方法は、自動化です。デプロイを実行するコードをテストし、それを使用してデプロイを実施します。これにより、変更プロセスに対する信頼性が高まり、変更に失敗するリスクが軽減されます。

設定が管理目標を満たしていることを確認するために、まず非本番環境で、自動化とデプロイされたアプリケーションをテストします。これにより、自動化をテストして、すべてのステップを正しく実行したことを証明できます。また、開発とデプロイサイクルの早期にフィードバックを得られるため、再作業を減らすことができます。デプロイエラーの可能性を減らすために、設定の変更は、人ではなくコードで行うようにしましょう。アプリケーションを再デプロイする必要がある場合は、自動化により、極めて簡単に行うことができます。追加の管理目標を定義すると、すべてのワークロードの自動化に簡単に追加することができます。

個々のワークロード所有者がワークロードに固有のセキュリティに投資する代わりに、共通の機能と共有コンポーネントを使用することで、時間を節約できます。複数のチームが利用できるサービスの例として、AWS アカウントの作成プロセス、人の ID の一元化、ロギングの共通設定、AMI やコンテナのベースイメージの作成などがあります。このアプローチにより、ビルダーはワークロードのサイクル時間を改善し、セキュリティ管理目標を一貫して達成することができます。チームの一貫性を高めることで、管理目標を検証し、管理態勢とリスクポジションを関係者に適切に報告できるようになります。

ベストプラクティス

- SEC01-BP03 管理目標を特定および検証する:
- SEC01-BP04 セキュリティの脅威と推奨事項の最新情報を入手する
- SEC01-BP05 セキュリティ管理のスコープを縮小する
- SEC01-BP06 標準的なセキュリティ統制のデプロイを自動化する
- SEC01-BP07 脅威モデルを使用して脅威を特定し、緩和策の優先順位を付ける
- SEC01-BP08 新しいセキュリティサービスと機能を定期的に評価および実装する

SEC01-BP03 管理目標を特定および検証する:

脅威モデルから特定されたコンプライアンス要件とリスクに基づいて、ワークロードに適用する必要がある管理目標および管理を導き出し、検証します。管理目標と制御を継続的に検証することは、リスク軽減の効果測定に役立ちます。

期待される成果: ビジネスのセキュリティコントロール上の目標が、コンプライアンス要件に一致するように明確に定義されます。自動化とポリシーを通じて統制が実装および実施され、目標を達成するために有効かどうかが継続的に評価されています。ある時点および一定期間の双方における有効性を示す証拠を、監査担当者にすぐに報告可能です。

一般的なアンチパターン:

- セキュリティを保証するために守るべき規制要件、市場の期待、業界標準についての理解が、ビジネスにとって不十分である
- サイバーセキュリティフレームワークと統制目標が、ビジネスの要件とかみ合っていない
- 実施されている統制が、測定可能な方法で統制目標にしっかりと適合していない
- 統制の有効性の報告に自動化を使っていない

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

セキュリティ統制目標の基礎として活用できる、一般的なサイバーセキュリティフレームワークは多数あります。ビジネスの規制要件、市場の期待、業界標準を考慮し、どのフレームワークがニーズに最も適しているかを判断してください。例えば、AICPA SOC 2、HITRUST、PCI-DSS、ISO 27001、NIST SP 800-53 などがあります。

特定した統制目標について、利用する AWS サービスがそれらの目標の達成にどのように役立つかを理解してください。目標とするフレームワークに沿ったドキュメントやレポートを探すときは AWS Artifact を使用します。これらの文書には、AWSが引き受ける責任の範囲と、顧客が引き受けるそれ以外の範囲についてのガイダンスが記されています。さまざまなフレームワークのコントロールステートメントに沿った、サービス固有のガイダンスの詳細については、「AWS Customer Compliance Guides」を参照してください。

目標達成を目指して統制を定義する際は、予防的統制を用いて実施について明文化し、発見的統制を用いて緩和策を自動化します。サービスコントロールポリシー (SCP) を使用すると、AWS Organizations 全体で非準拠のリソース構成やアクションを予防することができます。非準拠のリソースを監視および報告するときは AWS Config にルールを適用し、動作を確認できたらルールを強制モデルに切り替えます。自社のサイバーセキュリティのフレームワークに合わせて、事前定義されたマネージドルール一式を導入するときは、最初の選択肢として AWS Security Hub CSPM の標準の使用を検討します。AWS Foundational Service Best Practices (FSBP) 標準と CIS AWS Foundations Benchmark は、複数の標準フレームワークに共通の多数の目標に沿った統制を実現できるため、出発点として優れています。Security Hub CSPM に希望する統制検出の機能が組み込まれていない場合は、AWS Config コンフォーマンスパックで補完できます。

AWS Global Security and Compliance Acceleration (GSCA) チームが推奨する APN パートナーバンドルを使用すると、セキュリティアドバイザー、コンサルティング機関、証拠収集および報告のシステム、監査人、その他補完サービスによる支援を必要に応じて利用できます。

実装手順

1. 一般的なサイバーセキュリティフレームワークを評価し、選択したフレームワークに合わせて統制目標を定めます。
2. AWS Artifact を使用して、フレームワークのガイダンスと責任に関する関連文書を入手します。責任共有モデルにおいて、コンプライアンスのどの部分が AWS 側の責任で、どの部分が貴社の責任であるかを理解します。
3. SCP、リソースポリシー、ロール信頼ポリシー、その他のガードレールを使用して、非準拠のリソース構成やアクションを防止します。
4. 自社の統制目標に沿った Security Hub CSPM の標準と AWS Config コンフォーマンスパックの導入を評価します。

リソース

関連するベストプラクティス:

- SEC03-BP01 アクセス要件を定義する
- SEC04-BP01 サービスとアプリケーションのログ記録を設定する
- SEC07-BP01 データ分類スキームを理解する
- OPS01-BP03 ガバナンス要件を評価する
- OPS01-BP04 コンプライアンス要件を評価する
- PERF01-BP05 ポリシーとリファレンスアーキテクチャを使用する
- COST02-BP01 組織の要件に基づいてポリシーを策定する

関連ドキュメント:

- AWS Customer Compliance Guides

関連ツール:

- AWS Artifact

SEC01-BP04 セキュリティの脅威と推奨事項の最新情報を入手する

業界の脅威インテリジェンスの公開情報やデータフィードが更新されていないか監視して、脅威や緩和策の最新情報を常に把握します。最新の脅威データに基づいて自動更新されるマネージドサービスを評価してください。

期待される成果: 業界で最新の脅威や推奨事項が公表されると同時にそれらを把握できます。 自動化を活用して、新たな脅威を特定した時点で、潜在的な脆弱性やエクスポージャを検出します。これらの脅威に対して緩和措置を講じます。 最新の脅威インテリジェンスで自動的に更新される AWS サービスを採用します。

一般的なアンチパターン:

- 最新の脅威インテリジェンスを常に把握するための、信頼性と再現性が高いメカニズムがない。
- テクノロジーポートフォリオ、ワークロード、依存関係の手動インベントリを保持していて、潜在的な脆弱性やエクスポージャについて人間がレビューする必要がある。
- ワークロードと依存関係を、既知の脅威に対する緩和策が盛り込まれた最新バージョンに更新するメカニズムが導入されていない。

このベストプラクティスを活用するメリット: 脅威インテリジェンスの情報源から最新の情報を入手することで、脅威の分野における重要な変化を見落としてビジネスに影響を及ぼすリスクを避けることができます。 ワークロードやその依存関係をスキャンして、脆弱性やエクスポージャが潜んでいる箇所を検出および修正するための自動化体制が整い、手動での代替手段と比較して、リスクを迅速かつ予測どおりに軽減できます。 これにより、脆弱性の緩和に関連する時間やコストを抑えることができます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

信頼できる脅威インテリジェンスの公開情報を確認して、脅威の状況を常に把握してください。 既に周知の脅威への対抗手段、テクニック、手順 (TTP) に関するドキュメントは、MITRE ATT&CK のナレッジベースでご覧いただけます。MITRE の「Common Vulnerabilities and Exposures (CVE)」リストでは、ご利用の製品に関する既知の脆弱性情報を入手できます。Open Worldwide Application Security Project (OWASP) の中でも人気の高い OWASP Top 10 プロジェクトでは、ウェブアプリケーションの重大なリスクを把握できます。

AWS のセキュリティイベントと推奨される修復手順に関する最新情報は、CVE の AWS セキュリティ速報で入手できます。

最新情報を入手するための負担やオーバーヘッドを全体的に減らすために、新しい脅威インテリジェンスを自動で随時取り込む AWS サービスの使用を検討してください。 例えば、Amazon GuardDuty は業界の脅威インテリジェンスの最新情報を常に把握し、アカウント内の異常動作や脅威の兆候の検出に役立てています。 Amazon Inspector は、継続スキャンに使用している CVE のデータベースを自動更新しています。 AWS WAF と AWS Shield Advanced はどちらも、新しい脅威が登場すると自動的に更新されるマネージドルールグループを利用しています。

フリートの自動管理とパッチ適用については、「運用上の優秀性の柱 – AWS Well-Architected フレームワーク」を参照してください。

実装手順

- ビジネスや業界に関連する脅威インテリジェンスの最新公開情報を購読します。AWS セキュリティ速報を購読します。
- Amazon GuardDuty や Amazon Inspector など、新しい脅威インテリジェンスを自動的に組み込むサービスの採用を検討してください。
- Well-Architected 運用上の優秀性の柱のベストプラクティスに沿って、フリート管理とパッチ適用の戦略をデプロイします。

リソース

関連するベストプラクティス:

- SEC01-BP07 脅威モデルを使用して脅威を特定し、緩和策の優先順位を付ける
- OPS01-BP05 脅威の状況を評価する
- OPS11-BP01 継続的改善のプロセスを用意する

SEC01-BP05 セキュリティ管理のスコープを縮小する

特定の統制の管理を AWS に移行する AWS サービス (マネージドサービス) を利用することで、セキュリティの範囲を縮小できるか判断します。そうしたサービスを導入することで、インフラストラクチャのプロビジョニング、ソフトウェアのセットアップ、パッチ適用、バックアップなどのセキュリティメンテナンスのタスクを軽減できます。

期待される成果: ワークロードに適した AWS サービスを選ぶときに、セキュリティ管理の範囲を検討します。管理オーバーヘッドとメンテナンスタスクのコスト (総保有コスト (TCO)) が、Well-

Architected の他の考慮事項に加えて、選択したサービスのコストと比較検討されます。統制の評価と検証の手順に、AWS の統制とコンプライアンスの文書が組み込まれています。

一般的なアンチパターン:

- 選択したサービスの責任共有モデルをしっかりと理解しないまま、ワークロードをデプロイする。
- データベースやその他のテクノロジーを、同等のマネージドサービスを評価することなく仮想マシンでホストする。
- マネージドサービスオプションと比較する際に、仮想マシンでテクノロジーをホストする場合の総保有コスト (TCO) にセキュリティ管理タスクを考慮していない。

このベストプラクティスを活用するメリット: マネージドサービスを使用すると、運用上のセキュリティ統制を管理する全体的な負担を軽減でき、セキュリティリスクと総保有コストを減らすことができます。本来なら特定のセキュリティタスクに費やしていた時間を、ビジネスに付加価値をもたらすタスクに使うことができます。マネージドサービスを利用すれば、一部の統制要件を AWS に移し、コンプライアンス要件のスコープを縮小することもできます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

AWS では、多数の方法でワークロードのコンポーネントを統合できます。多くの場合、Amazon EC2 インスタンスにテクノロジーをインストールして実行するには、セキュリティの責任の大部分を自社で引き受けなければなりません。特定の統制の運用負担を軽減するには、責任共有モデルにおける自社の責任範囲が狭くなる AWS マネージドサービスを特定し、それらを既存のアーキテクチャでどのように使用できるかを理解してください。例えば、データベースのデプロイに Amazon Relational Database Service (Amazon RDS) を使用する、コンテナのオーケストレーションに Amazon Elastic Kubernetes Service (Amazon EKS) または Amazon Elastic Container Service (Amazon ECS) を使用する、もしくはサーバーレスのオプションを使用する、といったことが挙げられます。新しいアプリケーションを構築するときは、セキュリティ統制の実装と管理に関して、どのサービスが時間とコストの削減に役立つかを考えてください。

コンプライアンス要件も、サービス選択時の検討材料となり得ます。マネージドサービスでは、一部の要件のコンプライアンスを AWS に移すことができます。サービスの自社で運用管理する側面の監査や、関連する AWS 監査報告書の統制に関するステートメントの受け入れがどの程度容易かをコンプライアンスチームと話し合ってください。AWS Artifact で検出した監査アーティファクトは、AWS セキュリティ統制の証拠として監査人または規制当局に提出することができます。また、AWS の監査アーティファクトの一部によって提供された責任ガイダンスを、「AWS Customer

Compliance Guides」と併せて使用することで、アーキテクチャを設計することもできます。このガイダンスは、システムの特定のユースケースをサポートするために導入すべき追加のセキュリティ統制を決定するうえで役立ちます。

マネージドサービスを使用するときは、リソースを新しいバージョンに更新するプロセス (Amazon RDS で管理されるデータベースのバージョンや、AWS Lambda 関数のプログラミング言語ランタイムの更新など) をよく理解しておきましょう。そうした操作はマネージドサービスで実行される場合もありますが、更新のタイミングを設定し、運用への影響を理解することは依然としてお客様の責任です。AWS Health のようなツールを使用すると、こうした更新を自社の環境全体で追跡し管理することができます。

実装手順

1. マネージドサービスで置き換え可能なワークロードのコンポーネントを評価します。
    a. ワークロードを AWS に移行する場合は、ワークロードのリホスト、リファクタリング、リプラットフォーム、再構築、または交換が必要かどうかを評価する際に、管理 (時間と費用) の削減とリスクの軽減を考慮してください。移行の開始時に追加投資を行うことで、長期的には大幅な節約になる場合があります。
2. 独自のテクノロジーデプロイをインストールして管理する代わりに、Amazon RDS などのマネージドサービスを導入することを検討します。
3. AWS Artifact の責任ガイダンスを参考にして、ワークロードに対して導入すべきセキュリティ統制を決定します。
4. 使用中のリソースのインベントリを保管し、新しいサービスやアプローチに関する最新情報を入手して、スコープを縮小する新たな機会を特定します。

リソース

関連するベストプラクティス:

- PERF02-BP01 ワークロードに最適なコンピューティングオプションを選択する
- PERF03-BP01 データアクセスとストレージの要件に最適な専用データストアを使用する
- SUS05-BP03 マネージドサービスを使用する

関連ドキュメント:

- Planned lifecycle events for AWS Health

関連ツール:

- AWS Health
- AWS Artifact
- AWS Customer Compliance Guides

関連動画:

- How do I migrate to an Amazon RDS or Aurora MySQL DB instance using AWS DMS?
- AWS re:Invent 2023 - Manage resource lifecycle events at scale with AWS Health

SEC01-BP06 標準的なセキュリティ統制のデプロイを自動化する

あらゆる AWS 環境で標準とするセキュリティ統制の開発とデプロイに際しては、最新の DevOps プラクティスを適用してください。 標準的なセキュリティ統制と構成を Infrastructure as Code (IaC) テンプレートに定義し、バージョン管理システムで変更を取り込み、CI/CD パイプラインの一環として変更をテストし、AWS 環境への変更のデプロイを自動化します。

期待される成果: IaC テンプレートで標準化されたセキュリティ統制が定義され、バージョン管理システムにコミットされます。 変更を検出し、テストと AWS 環境へのデプロイを自動化する CI/CD パイプラインが整備されています。 ガードレールが効いていて、テンプレート内の設定ミスをデプロイ前に検出し、警告します。 標準の統制が効いている環境にワークロードがデプロイされます。 チームには、承認済みのサービス構成をセルフサービスメカニズムを通じてデプロイする権限が与えられています。 統制の構成、スクリプト、関連データに対して、安全なバックアップと復旧の戦略が実施されています。

一般的なアンチパターン:

- 標準のセキュリティ統制に対する変更をウェブコンソールやコマンドラインインターフェイスを使用して手作業で行っている。
- 中央のチームが定義した統制の実装は、個々のワークロードチームによる手作業に頼っている。
- ワークロードチームの要求に応じてワークロードレベルの統制をデプロイするのは、中央のセキュリティチームに一任されている。
- セキュリティ統制の自動化スクリプトの開発、テスト、デプロイを同じ個人またはチームが担当でき、職務分離やチェックアンドバランス (抑制と均衡) が適切に機能していない。

このベストプラクティスを活用するメリット: 標準のセキュリティ統制をテンプレートに定義しておくことで、バージョン管理システムを使って変化を経時的に追跡し、比較することができます。 変更のテストとデプロイを自動化することで、プロセスが標準化されて予測可能性が高まり、デプロイの成功率が上がり、繰り返しの手作業を省くことができます。 承認済みのサービスと構成をワークロードチームがデプロイできるセルフサービスのメカニズムが用意されているため、構成ミスや誤用のリスクが軽減されます。また、開発プロセスの早い段階で統制を組み込むことができます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

「SEC01-BP01 アカウントを使用してワークロードを分ける」に記載のベストプラクティスに従うと、AWS Organizations を使って管理している異なる環境ごとに、複数の AWS アカウントアカウントを抱えることになります。 これらの環境やワークロードで個別のセキュリティ統制が必要になる場合がある一方で、一部のセキュリティ統制は標準化して組織全体に適用できます。 例えば、一元管理の ID プロバイダーの統合、ネットワークとファイアウォールの定義、ログの保管と分析のための標準の場所の設定などが該当します。 Infrastructure as Code (IaC) を使用すると、アプリケーションコードの開発と同等の厳格さをインフラストラクチャのプロビジョニングに適用できるように、IaC を使用すると、標準のセキュリティ統制を同様に定義してデプロイすることができます。

セキュリティ統制は、可能な限り宣言的な方法で定義し (AWS CloudFormation で定義する場合と同じ)、ソース管理システムに保存します。 DevOps のプラクティスを使用して、統制のデプロイを自動化してリリースの予測可能性を高め、AWS CloudFormation Guard などのツールを使ってテストを自動化し、デプロイした統制の、目的とする構成からの逸脱を検出します。 CI/CD パイプラインの構築には、AWS CodePipeline、AWS CodeBuild、AWS CodeDeploy などのサービスを使用できます。これらのサービスを、他のデプロイパイプラインから分離された専用のアカウントで設定するときは、「複数のアカウントで AWS 環境を構成する」のガイダンスを考慮します。

テンプレートを定義して、AWS アカウント、サービス、構成の定義とデプロイを標準化することもできます。 この方法なら、それらの定義を中央のセキュリティチームが管理し、セルフサービスアプローチでワークロードチームに提供できます。 これを実現する方法の 1 つが Service Catalog を使用した方法です。この方法では、ワークロードチームが独自のパイプラインデプロイに組み込みこむことのできる製品として、テンプレートをパブリッシュできます。 AWS Control Tower を使用している場合、出発点として使用できるテンプレートや統制がいくつか用意されています。 Control Tower には、ワークロードチームが、ユーザーが定義した標準を使って新しい AWS アカウントアカウントを作成できる、Account Factory という機能もあります。 新しいアカウントが必要だとワークロードチームが判断した際に、その承認と作成を中央のチームに依存する必要がなくなります。 これらのアカウントは、さまざまなワークロードコンポーネントを、それぞれの機能や動作、処理対象のデータの機密性といった理由で分離する場合に必要になることがあります。

実装手順

1. テンプレートをバージョン管理システムに保存し、管理する方法を決定します。
2. テンプレートをテストおよびデプロイする CI/CD パイプラインを作成します。 設定ミスがないかチェックし、テンプレートが会社の標準に準拠していることを確認するテストを定義します。
3. ワークロードチームが要件に従って AWS アカウントやサービスをデプロイできるように、標準化されたテンプレートのカタログを作成しておきます。
4. 統制の構成、スクリプト、関連データに対して、安全なバックアップと復旧の戦略を実装します。

リソース

関連するベストプラクティス:

- OPS05-BP01 バージョン管理を使用する
- OPS05-BP04 構築およびデプロイ管理システムを使用する
- REL08-BP05 オートメーションを使用して変更をデプロイする
- SUS06-BP01 持続可能性の改善を迅速に導入できる方法を採用する

関連ドキュメント:

- 複数のアカウントで AWS 環境を構成する

関連する例:

- Automate account creation, and resource provisioning using Service Catalog, AWS Organizations, and AWS Lambda
- Strengthen the DevOps pipeline and protect data with AWS Secrets Manager, AWS KMS, and AWS Certificate Manager

関連ツール:

- AWS CloudFormation Guard
- Landing Zone Accelerator on AWS

SEC01-BP07 脅威モデルを使用して脅威を特定し、緩和策の優先順位を付ける

脅威モデリングを実行し、ワークロードの潜在的脅威と関連付けられた緩和策を特定し、最新の状態を維持します。脅威に優先順位を付け、セキュリティコントロール緩和策を調整して防止、検出、対応を行います。ワークロードの内容、および進化するセキュリティ環境の状況に応じてセキュリティコントロールを保持および維持します。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

脅威モデリングとは?

「脅威モデリングとは、価値あるものを保護する状況において、脅威とその緩和策を特定し、伝達して理解する取り組みをいう」。– The Open Web Application Security Project (OWASP) Application Threat Modeling

なぜ脅威モデリングが必要なのか?

システムは複雑であり、時代とともに次第に複雑かつ高性能となり、提供するビジネス価値は向上し、顧客満足度とエンゲージメントは強化されています。つまり、IT 設計を決定する際は、増え続けるユースケースの件数を考慮する必要があるということです。このような複雑で数が多いユースケースの組み合わせは、非構造化アプローチでは一般に脅威の検出と緩和に効果がありません。代わりに必要となるのは、システムに対する潜在的な脅威を列挙し、緩和策を考案し、その緩和策に優先順位をつけて、組織の限定的なリソースがシステム全体のセキュリティ体制の改善に最大の効果を発揮できるような体系的アプローチです。

脅威モデリングは、このような体系的アプローチを提供する設計となっており、その狙いは、ライフサイクルの後半と比較すると相対的にコストと労力が低い設計プロセスの早い段階で問題を発見し、対処することです。このアプローチは、業界の原則であるセキュリティ戦略「シフトレフト」と一致しています。最終的に脅威モデリングは組織のリスク管理プロセスと統合し、脅威駆動型アプローチを使用して、実装するコントロールの決定を促します。

脅威モデリングを実行するタイミング

ワークロードのライフサイクルにおけるできるだけ早い段階で脅威モデリングを開始することにより、より柔軟に特定した脅威への対策を実施できるようになります。ソフトウェアのバグと同様、脅威を特定するのが早いほど、その対策のコスト効率が向上します。脅威モデルはライブドキュメント

であり、ワークロードの変化に応じて進化し続ける必要があります。大きな変化、脅威の状況における変化が生じた場合や、新たな機能またはサービスを採用した場合などを含む、経時的な脅威モデルを保持します。

実装手順

脅威モデリングの実行方法

脅威モデリングにはさまざまな実行方法があります。プログラミング言語と同様、それぞれに長所と短所があり、自分に最も適した方法を選択する必要があります。そのうちの 1 つが「Shostack's 4 Question Frame for Threat Modeling」から始める方法です。ここでは、自由形式の質問をたずねることで脅威モデリングに枠組みを与えます。

1. 現在取り組んでいることは何か。

   この質問の目的は、構築しているシステム、さらにはセキュリティに関連するシステムに関する詳細を理解してそれに合意するのを支援することです。この質問に答える最も一般的な方法は、モデルや図を作成することです。それにより、現在構築しているものをデータフロー図などを使って視覚化することができます。システムに関する推測と重要な詳細を書き留めることも、対象範囲を定義するのに役立ちます。これにより、脅威モデルに取り組む担当者全員の目指す方向が合致し、対象範囲外のトピック (システムの古いバージョンなど) に脱線して時間を浪費する事態を回避できます。例えば、ウェブアプリケーションを構築している場合、ブラウザクライアントのオペレーティングシステムの信頼できるブートシーケンスをモデル化する脅威については、あまり時間をかける価値があるとは思えません。

2. 問題化する可能性があるものは何か?

   ここで、システムに対する脅威を特定します。脅威とは、望ましくない影響を生じさせ、システムのセキュリティに悪影響を及ぼす恐れのある、偶発的または意図的なアクションや事象を指します。どのような問題が起きるかをはっきりと理解していなければ、何も対策は打てません。

   何が問題になるのかに関して、定型的なリストは存在しません。このリストを作成するには、チームのメンバーと脅威モデリングに参加する関係者との全員による、ブレインストーミングとコラボレーションが必要となります。ブレインストーミングは、STRIDE など、脅威を特定するモデルを使用するとスムーズに行うことができます。STRIDE は、評価すべきさまざまなカテゴリ (なりすまし、改ざん、否認、情報漏洩、サービス妨害、権限昇格) を提案するものです。また、OWASP Top 10、HiTrust Threat Catalog、組織独自の脅威カタログなど、既存のリストを見直して調査することもブレインストーミングの刺激となり役に立ちます。

3. どのように対処すべきか?

前の質問と同様、考えられる緩和策について定型的なリストはありません。このステップに対する入力項目は、特定された脅威、アクター、および前のステップからの改善点です。

セキュリティとコンプライアンスは、AWS とユーザー間で共有される責任です。「それをどうするのですか?」という質問を行うときは、「誰がその責任者なのか?」ということもたずねていると理解することが重要です。ユーザーと AWS との間の、責任のバランスを理解することにより、ユーザーのコントロール下にある脅威モデリングの範囲を理解することができます。こちらは通常、AWS サービスの設定オプションと、ユーザー独自のシステムごとの緩和策とを組み合わせたものになります。

共有責任の AWS の担当部分については、AWS サービスが多くのコンプライアンスプログラムの対象範囲内であることがわかるはずです。これらのプログラムは、セキュリティとクラウドのコンプライアンスを維持するためにAWS に配置された堅牢なコントロールを理解するのに役立ちます。これらのプログラムの監査レポートは、AWS の顧客は AWS Artifact からダウンロードできます。

どの AWS サービスを使用していても、必ずお客様の責任となる要素が存在し、これらの責任に合わせた緩和策を脅威モデルに組み込む必要があります。AWS サービス自体のセキュリティコントロール緩和のためには、例えば、Identity and Access Management (認証と承認)、データ保護 (静止時と転送時)、インフラストラクチャセキュリティ、ログ、モニタリングなどのドメインを含む、さまざまなドメイン全体にセキュリティコントロールの実装を検討することが推奨されます。各 AWS サービスのドキュメントにはそれぞれセキュリティに関する項目があり、緩和策として利用できるセキュリティコントロールについてのガイダンスが記されています。重要ですので、記述しているコードとコード依存関係を考慮し、それらの脅威に対応するために設定できるコントロールについて考えてください。こうしたコントロールには、入力の検証、セッションの処理、境界の処理などが含まれます。多くの場合、脆弱性の大部分はカスタムコードで発生するため、この領域を注視してください。

4. 十分に優れた仕事をしたか?

狙いは、チームと組織が脅威モデルの質と、脅威モデリングを行う際の時間的な速さを改善することです。これらの改善は、練習、学習、指導、レビューを組み合わせることで実現します。十分に理解して実践的な経験を積むには、お客様とお客様のチームメンバー全員が「Threat modeling the right way for builders training course」またはワークショップを受講することが推奨されます。さらに、組織のアプリケーション開発のライフサイクルに、脅威モデリングを組み込む方法について知りたい方は、AWS セキュリティブログの「脅威モデリングのアプローチ方法」を参照してください。

Threat Composer

脅威モデリングをサポートし実行を導くには、Threat Composer ツールの使用を検討します。このツールは、脅威モデリングの価値を実感できるまでにかかる時間を、短縮するためのツールです。このツールは、以下の用途で役立ちます。

- 自然な非線形のワークフローに該当する、Threat grammar に沿った有益な脅威のステートメントを記述する
- 人間が読める脅威モデルを生成する
- 機械可読な脅威モデルを生成して、脅威モデルをコードとして扱えるようにする
- Insights Dashboard を使用して、品質や対象範囲を改善できる分野をすばやく特定する

詳細については、Threat Composer にアクセスした後、システムで定義されたExample Workspace に切り替えます。

リソース

関連するベストプラクティス:

- SEC01-BP03 管理目標を特定および検証する:
- SEC01-BP04 セキュリティの脅威と推奨事項の最新情報を入手する
- SEC01-BP05 セキュリティ管理のスコープを縮小する
- SEC01-BP08 新しいセキュリティサービスと機能を定期的に評価および実装する

関連ドキュメント:

- 脅威モデリングのアプローチ方法 (AWS セキュリティブログ)
- NIST: Guide to Data-Centric System Threat modeling

関連動画:

- AWS Summit ANZ 2021 - How to approach threat modeling
- AWS Summit ANZ 2022 - Scaling security – Optimise for fast and secure delivery

関連するトレーニング:

- Threat modeling the right way for builders – AWS Skill Builder virtual self-paced training
- Threat modeling the right way for builders – AWS Workshop

関連ツール:

- Threat Composer

SEC01-BP08 新しいセキュリティサービスと機能を定期的に評価および実装する

ワークロードのセキュリティ体制を進化させるために役立つ、AWS および AWS パートナーのセキュリティサービスと機能を評価および実装します。

期待される成果: AWS や AWS パートナーがリリースした新しい機能やサービスについて知らせるための標準的な実践方法が確立されています。これらの新機能が、環境とワークロードに対する既存および新規の統制の設計にどのように影響するかを評価します。

一般的なアンチパターン:

- 関連する新しい機能やサービスの情報を迅速に入手するために、AWS のブログや RSS フィードを購読していない
- セキュリティサービスや機能に関するニュースや最新情報を二次情報源から入手している
- 組織内の AWS ユーザーに、常に最新情報に触れるよう推奨していない

このベストプラクティスを活用するメリット: 新しいセキュリティサービスや機能を常に把握していれば、クラウド環境やワークロードへの統制の実装について、情報に基づいて決断を下せます。進化するセキュリティ環境や、新たに出現した脅威への対策として AWS サービスを活用する方法を把握するうえで、それらの情報源が役に立ちます。

このベストプラクティスを活用しない場合のリスクレベル: 低

実装のガイダンス

AWS では、新しいセキュリティサービスや機能について、複数のチャネルを通じてお客様にご案内しています。

- AWS の最新情報

- AWS ニュースブログ
- AWS セキュリティブログ
- AWS セキュリティ速報
- AWS ドキュメントの概要

Amazon Simple Notification Service (Amazon SNS) を使用して AWS Daily Feature Updates トピックを購読し、最新情報の包括的なサマリーを確認できます。Amazon GuardDuty や AWS Security Hub CSPM などの一部のセキュリティサービスは、独自の SNS トピックで各サービスに関する新しい標準、検出結果、その他の最新情報を配信しています。

また、毎年世界中で開催されるカンファレンス、イベント、ウェビナーでも、新しいサービスや機能が発表され、詳しく説明されています。中でも注目すべきは、毎年恒例の AWS re:Inforce セキュリティカンファレンスと、より一般的な AWS re:Invent カンファレンスです。前述の AWS のニュースチャネルでは、これらのカンファレンスでセキュリティやその他のサービスに関して発表した内容を共有しています。また、YouTube の AWS Events チャンネルでは、深く掘り下げて学ぶブレイクアウトセッションをオンラインでご視聴いただけます。

セキュリティサービスの最新情報や新しい推奨事項について、AWS アカウントチームにお問い合わせいただくこともできます。セールスサポートチーム直通の連絡先情報がお手元にない場合は、セールスサポートフォームからお問い合わせください。同様に、AWS エンタープライズサポートにご登録いただいている場合は、Technical Account Manager (TAM) が毎週最新情報をお知らせします。また、TAM との定期的なレビューミーティングを設定できます。

実装手順

1. お気に入りの RSS リーダーでさまざまなブログや速報を購読するか、「Daily Feature Updates」SNS トピックを購読します。
2. 新しい機能やサービスについて直に学ぶため、どの AWS イベントに参加すべきかを評価します。
3. セキュリティサービスや機能の更新について質問がある場合は、AWS アカウントチームとのミーティングを設定します。
4. エンタープライズサポートへの登録を検討します。登録すると、Technical Account Manager (TAM) に定期的に相談できます。

リソース

関連するベストプラクティス:

- PERF01-BP01 利用可能なクラウドサービスと機能について学び、理解する
- COST01-BP07 新しいサービスリリースに関する最新情報を把握しておく

Identity and Access Management

AWS のサービスを使用するには、ユーザーとアプリケーションに AWS アカウントのリソースへのアクセス権限を与える必要があります。AWS で実行するワークロードの増加に伴い、適切なユーザーが適切な条件で適切なリソースにアクセスできるようにするためには、強固な ID 管理とアクセス許可が必要です。AWS は、幅広い機能の選択肢を提供することによって、ユーザーとマシンの ID および権限の管理を支援しています。これらの機能のベストプラクティスは、次の 2 つの領域に大きく分類されます。

トピック

ID 管理

安全な AWS ワークロードの運用へのアプローチでは、管理が必要な 2 つのタイプの ID があります。

- 人的 ID: AWS 環境やアプリケーションへのアクセスを必要とする人間の ID は、ワークフォース、サードパーティー、ユーザーの 3 つのグループに分類できます。

  ワークフォースグループには、組織のメンバーである管理者、デベロッパー、オペレーターが含まれます。AWS リソースを管理、構築、運用するためのアクセス権が必要です。

  サードパーティーとは、請負業者、ベンダー、パートナーなどの外部協力者です。サードパーティーは、組織とのエンゲージメントの一環として AWS リソースとやり取りします。

  ユーザーとは、アプリケーションの消費者です。ウェブブラウザ、クライアントアプリケーション、モバイルアプリケーション、またはインタラクティブなコマンドラインツールを介して AWS リソースにアクセスします。
- マシン ID: ワークロードアプリケーション、運用ツール、コンポーネントには、データ読み取りなどのため、AWS のサービスにリクエストを送信できる ID が必要です。これらの ID には、Amazon EC2 インスタンスや AWS Lambda 関数などの AWS 環境で実行中のマシンも含まれます。また、AWS 環境へのアクセスを必要とする外部関係者のマシン ID、または AWS 外のマシンのマシン ID を管理することもできます。

ベストプラクティス

- SEC02-BP01 強力なサインインメカニズムを使用する
- SEC02-BP02 一時的な認証情報を使用する
- SEC02-BP03 シークレットを安全に保存して使用する
- SEC02-BP04 一元化された ID プロバイダーを利用する
- SEC02-BP05 定期的に認証情報を監査およびローテーションする
- SEC02-BP06 ユーザーグループと属性を採用する

SEC02-BP01 強力なサインインメカニズムを使用する

サインイン (サインイン認証情報を使った認証) は、多要素認証 (MFA) などのメカニズムを使わない場合、特にサインイン認証情報が不用意に開示されたり、容易に推測されたりする場合に、リスクが発生する恐れがあります。MFA や強力なパスワードポリシーを要求することで、これらのリスクを軽減する強力なサインインのメカニズムを使用します。

期待される成果: AWS Identity and Access Management (IAM) ユーザー、AWS アカウントルートユーザー、AWS IAM アイデンティティセンター、およびサードパーティー ID プロバイダーに強力なサインインメカニズムを使用することで、AWS の認証情報への意図しないアクセスのリスクを軽減します。これは、MFA が必須となり、強力なパスワードポリシーが適用され、異常なログイン動作が検出されることを意味します。

一般的なアンチパターン:

- 複雑なパスワードや MFA など、自分のアイデンティティに対して強力なパスワードポリシーを適用しない。
- 複数のユーザー間で同一の認証情報を共有する。
- 疑わしいサインインに対して検出コントロールを使用しない。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

人的 ID が AWS にサインインする方法は多数あります。AWS への認証時に、フェデレーション (AWS IAM と集中型 IdP 間の直接 SAML 2.0 フェデレーション、またはAWS IAM アイデンティティセンターを使用) を使用して一元化された ID プロバイダーに依存するのが AWS ベストプラクティス

です。この場合、ID プロバイダーまたは Microsoft Active Directory を使って、セキュアなサインインプロセスを確立します。

最初に AWS アカウントを開いたとき、AWS アカウントルートユーザーから始めます。アカウントのルートユーザーは、ユーザー (およびルートユーザーを必要とするタスク) のアクセスを設定するときのみに使用することを推奨します。AWS アカウント を開いた直後にアカウントのルートユーザーに対して多要素認証 (MFA) を有効にし、AWS ベストプラクティスガイドを使用してルートユーザーを保護することが重要です。

AWS IAM アイデンティティセンターはワークフォースユーザー向けに設計されており、サービス内でユーザー ID を作成および管理し、MFA を使用してサインインプロセスを保護できます。AWS一方、Cognito は、アプリケーションの外部ユーザー ID のユーザープールと ID プロバイダーを提供するカスタマー ID とアクセス管理 (CIAM) 用に設計されています。

AWS IAM アイデンティティセンターでユーザーを作成する場合は、そのサービスでサインインプロセスを保護し、MFA をオンにします。アプリケーション内の外部ユーザー ID については、Amazon Cognito ユーザープールを使用して、そのサービス内または Amazon Cognito ユーザープールがサポートする ID プロバイダーのいずれかを通じてサインインプロセスを保護できます。

さらに、AWS IAM アイデンティティセンターのユーザーの場合、AWS Verified Access を使用して、AWS リソースへのアクセス権が付与される前にユーザーの ID とデバイスの状態を検証して、セキュリティレイヤーを強化できます。

AWS Identity and Access Management (IAM) ユーザーを使用している場合、IAM を使用してサインインプロセスをセキュリティ保護することになります。

AWS IAM アイデンティティセンターとダイレクト IAM フェデレーションの両方を同時に使用して、AWS へのアクセスを管理できます。IAM フェデレーションを使用して AWS マネジメントコンソール およびサービスへのアクセスを管理し、IAM アイデンティティセンターを使用して Quick や Amazon Q Business などのビジネスアプリケーションへのアクセスを管理できます。

サインイン方法に関係なく、強力なサインインポリシーを適用することが不可欠です。

実装手順

一般的な強力なサインインに関する推奨事項は次のとおりです。実際に行う設定は、組織のポリシーによって設定するか、または NIST 800-63 のような標準を使います。

- MFA が必要。人的 ID とワークロードに対しては、MFA を義務付けることが IAM のベストプラクティスです。MFA を有効にすることで、追加のセキュリティ層が提供されます。この層では、

ユーザーがサインイン認証情報、ワンタイムパスワード (OTP)、またはハードウェアデバイスから暗号的に検証および生成された文字列を提供することが求められます。

- 最小パスワード文字数を適用します。これは、パスワードの強さにおける主な要素です。
- パスワードの複雑性を適用すると、パスワードを推測しにくくなります。
- 自分のパスワードの変更をユーザーに許可します。
- 共有認証情報ではなく、個別の ID を作成します。個別の ID を作成することで、各ユーザーに固有のセキュリティ認証情報を付与することができます。個別のユーザーを作成することによって、各ユーザーのアクティビティを監査する機能が利用できます。

IAM アイデンティティセンターレコメンデーション:

- IAM アイデンティティセンターは、デフォルトディレクトリを使用する際、パスワードの文字数、複雑性、および再使用要件を確立する、事前定義されたパスワードポリシーを提供します。
- MFA を有効にし、アイデンティティソースがデフォルトディレクトリ、AWS Managed Microsoft AD、または AD Connector の場合、MFA に対してコンテキストアウェアまたは常時オン設定を行います。
- ユーザーが自分の MFA デバイスを登録できるようにします。

Amazon Cognito ユーザープールディレクトリのレコメンデーション:

- パスワードの強さ設定を行います。
- ユーザーに MFA を義務付けます。
- 疑わしいサインインをブロックできる適応型認証などの機能に対して、Amazon Cognito ユーザープール上級セキュリティ設定を使用します。

IAM ユーザーのレコメンデーション:

- IAM アイデンティティセンターまたは直接フェデレーションを使用することが理想的です。しかし、IAM ユーザー向けのニーズもあるでしょう。その場合は、IAM ユーザー向けにパスワードポリシーを設定します。パスワードポリシーを使用して、最小文字数、またはアルファベット以外の文字が必要かどうかなどの要件を定義できます。
- IAM ポリシーを作成して、MFA サインインを適用し、ユーザーが自分のパスワードと MFA デバイスを管理できるようにします。

リソース

関連するベストプラクティス:

- SEC02-BP03 シークレットを安全に保存して使用する
- SEC02-BP04 一元化された ID プロバイダーを利用する
- SEC03-BP08 組織内でリソースを安全に共有する

関連ドキュメント:

- AWS IAM アイデンティティセンターパスワードポリシー
- IAM ユーザーのパスワードポリシー
- AWS アカウントルートユーザーのパスワードの設定
- Amazon Cognito パスワードポリシー
- AWS 認証情報
- IAM セキュリティのベストプラクティス

関連動画:

- Managing user permissions at scale with AWS IAM Identity Center
- Mastering identity at every layer of the cake

SEC02-BP02 一時的な認証情報を使用する

何らかの認証を行う際、認証情報が誤って開示、共有、盗難されたりなどのリスクを軽減または排除するには、長期的認証情報ではなく一時的な認証情報を使うことが推奨されます。

期待される成果: 長期的認証情報のリスクを軽減するには、人的および機械両方の ID にできるだけ一時的な認証情報を使用するようにします。長期認証情報は、公開リポジトリへのアップロードによる情報漏洩など、多くのリスクが発生します。一時的な認証情報を使うことにより、認証情報が侵害されるリスクが大幅に減少します。

一般的なアンチパターン:

- 開発者が、フェデレーションを使って CLI から一時的な認証情報を取得するのではなく、IAM ユーザーからの長期的なアクセスキーを使用する。

- 開発者がコードに長期的アクセスキーを埋め込んで、そのコードをパブリック Git リポジトリにアップロードする。
- 開発者が、モバイルアプリに長期的アクセスキーを埋め込んで、アプリストアで公開する。
- ユーザーが長期的アクセスキーを他のユーザー、または従業員と共有し、長期的アクセスキーを所有したまま離職する。
- 一時的認証情報を使用できるのに、マシン ID に対して長期的なアクセスキーを使用する。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

すべての AWS API と CLI リクエストに対して、長期的認証情報ではなく一時的なセキュリティ認証情報を使用します。AWS サービスに対する API および CLI リクエストは、ほとんどの場合、AWS アクセスキーを使って署名する必要があります。これらのリクエストの署名に使用する認証情報は、一時的でも長期的でもかまいません。長期的認証情報 (長期的アクセスキー) を使用すべき唯一の状況は、IAM ユーザーまたは AWS アカウントルートユーザーを使用している場合です。AWS に対してフェデレーションを行うか、または他の方法により IAM ロールを担う場合、一時的認証情報が生成されます。サインイン認証情報を使って AWS マネジメントコンソールにアクセスしても、AWS サービスへのコールを行うために一時的な認証情報が生成されます。長期的認証情報が必要な状況はほとんどなく、一時的な認証情報でほとんどのタスクを遂行できます。

一時的な認証情報を優先して長期的な認証情報の使用を回避することは、フェデレーションと IAM ロールを優先して IAM ユーザーの使用を減少させる戦略と一致していなければなりません。IAM ユーザーは過去に人的とマシン ID 両方に対して使用されましたが、長期的アクセスキー使用におけるリスクを回避するため、それを使用しないよう推奨しています。

実装手順

人的 ID

従業員、管理者、デベロッパー、およびオペレーターなどのワークフォース ID の場合:

- 一元化された ID プロバイダーに依存して、人間ユーザーが一時的な認証情報を使って AWS にアクセスするには、ID プロバイダーにフェデレーションを使用することを義務付ける必要があります。ユーザーに対するフェデレーションは、各 AWS アカウント の直接のフェデレーション、または AWS IAM アイデンティティセンターと任意の ID プロバイダーを使用して行うことができます。フェデレーションは、長期的な認証情報を排除するだけでなく、IAM ユーザーを使用する場

合と比較して多数の利点があります。ユーザーは直接フェデレーション用のコマンド行から、または IAM Identity Center を使用して、一時的な認証情報をリクエストすることができます。つまり、IAM ユーザーまたは、ユーザー向けの長期的認証情報を必要なケースはほとんどないということです。

サードパーティー ID の場合:

- Software as a Service (SaaS) などのサードパーティーに、AWS アカウントのリソースへのアクセスを付与する際、クロスアカウントロールおよびリソースベースポリシーを使用できます。また、B2B SaaS の顧客またはパートナーには、Amazon Cognito OAuth 2.0 付与クライアント認証情報フローを使用することもできます。

ウェブブラウザ、クライアントアプリケーション、モバイルアプリケーション、またはインタラクティブなコマンドラインツールを介して AWS リソースにアクセスするユーザー ID:

- 消費者や顧客向けのアプリケーションに AWS リソースへのアクセスを許可する必要がある場合、Amazon Cognito アイデンティティプールまたは Amazon Cognito ユーザープールを使用して、一時的な認証情報を提供できます。認証情報に対するアクセス許可は、IAM ロールを介して設定されます。また、認証されていないゲストユーザーに対して、権限が制限された個別の IAM ロールを定義することもできます。

マシン ID

マシン ID の場合、長期的認証情報を使用しなければならない場合があります。これらの場合、IAM ロールで AWS にアクセスする際に、ワークロードが一時的な認証情報を使用するよう義務付ける必要があります。

- Amazon Elastic Compute Cloud (Amazon EC2) の場合、Amazon EC2 に対してロールを使用できます。
- AWS Lambda では、一時的な認証情報を使って AWS アクションを実行するためのサービス権限を付与する Lambda 実行ロールを設定できます。AWS サービスが、IAM ロールを使って一時的な認証情報を付与する類似モデルは多数あります。
- IoT デバイスの場合、AWS IoT Core 認証情報プロバイダーを使って、一時的な認証情報をリクエストできます。
- オンプレミスのシステム、または AWS 外で実行され、AWS リソースへアクセスする必要があるシステムの場合、IAM Roles Anywhere を使用できます。

一時的な認証情報がサポートされていないシナリオがあり、その場合は長期認証情報を使用する必要があります。これらの状況では、定期的にこれらの認証情報を監査してローテーションし、さらに定期的にアクセスキーをローテーションします。制限の厳しい IAM ユーザーアクセスキーについては、以下の追加のセキュリティ対策を検討してください。

- 次のように、制限の厳しいアクセス許可を付与します。
  - 最小特権の原則 (アクション、リソース、条件を具体的に指定します) に従います。
  - IAM ユーザーに、特定の 1 つのロールに対する AssumeRole オペレーションのみを付与することを検討してください。オンプレミスアーキテクチャの一部では、このアプローチを使用することにより、長期 IAM 認証情報を分離して安全に保護できます。
- IAM ロール信頼ポリシーで許可されるネットワークソースと IP アドレスを制限します。
- 使用状況をモニタリングし、未使用のアクセス許可や誤使用に対してアラートを設定します (AWSCloudWatch Logs メトリクスフィルターとアラームを使用)。
- アクセス許可の境界 を適用します (サービスコントロールポリシー (SCP) は粗く、アクセス許可の境界はきめ細かく、相互に補完する関係にあります)。
- 認証情報をプロビジョニングして、安全に (オンプレミスのボールトに) 保存するプロセスを実装します。

長期認証情報が必要なシナリオのその他のオプションには、次のようなものがあります。

- (Amazon API Gateway を使用して) 独自のトークン供給 API を構築する。
- 長期認証情報や AWS アクセスキー以外の認証情報 (データベースログインなど) を使用する必要があるシナリオでは、AWS Secrets Manager など、シークレット管理を処理するために設計されたサービスを使用できます。Secrets Manager は、暗号化されたシークレットの管理、ローテーション、安全なストレージを簡素化します。多くの AWS サービスでは、Secrets Manager との直接統合をサポートしています。
- マルチクラウド統合では、ソース認証情報サービスプロバイダー (CSP) の認証情報に基づいた ID フェデレーションを使用できます (「AWS STS AssumeRoleWithWebIdentity」を参照)。

長期的認証情報のローテーションについては、「アクセスキーのローテーション」を参照してください。

リソース

関連するベストプラクティス:

- SEC02-BP03 シークレットを安全に保存して使用する
- SEC02-BP04 一元化された ID プロバイダーを利用する
- SEC03-BP08 組織内でリソースを安全に共有する

関連ドキュメント:

- 一時的な認証情報
- AWS 認証情報
- IAM セキュリティのベストプラクティス
- IAM ロール
- AWS IAM アイデンティティセンター
- ID プロバイダーとフェデレーション
- アクセスキーの更新
- AWS セキュリティコンピテンシーパートナー
- AWS アカウントのルートユーザー
- Google Cloud Platform ネイティブワークロード ID を使用して AWS にアクセスする
- AWS Security Token Service を使用して Microsoft Entra ID テナントから AWS リソースにアクセスする方法

関連動画:

- Managing user permissions at scale with AWS IAM Identity Center
- Mastering identity at every layer of the cake

SEC02-BP03 シークレットを安全に保存して使用する

ワークロードには、データベース、リソース、およびサードパーティーサービスにアイデンティティを証明するための自動機能が必要となります。これは、API アクセスキー、パスワード、および OAuth トークンなどの、シークレットアクセス認証情報を使って実現されます。これらの認証情報を保存、管理、ローテーションする専用のサービスを使用することで、認証情報が侵害される可能性を低減することができます。

期待される成果: 次の目標を達成するアプリケーションの認証情報を安全に管理するメカニズムを実装します。

- ワークロードに必要なシークレットを特定する。
- 長期的認証情報を短期的認証情報と置き換える (可能な場合) ことによりその数を減らす。
- 安全なストレージと、残りの長期的認証情報の自動化されたローテーションを確立する。
- ワークロードに存在するシークレットへのアクセスを監査する。
- 開発プロセス中、ソースコードに組み込まれたシークレットがないことを継続的に監視する。
- 認証情報が誤って開示される可能性を減らす。

一般的なアンチパターン:

- 認証情報をローテーションしない。
- ソースコードまたは設定ファイルに長期的認証情報を保管する。
- 認証情報を暗号化せずに保管する。

このベストプラクティスを活用するメリット:

- シークレットが、保管時と転送時に暗号化される。
- 認証情報へのアクセスが、API (認証情報の自動販売機として捉える) 経由でゲート化される。
- 認証情報へのアクセス (読み出しと書き込み) が監査およびログ記録される。
- 懸念事項の分離: 認証情報のローテーションは、アーキテクチャの他の部分から分離できる別のコンポーネントによって実行されます。
- シークレットは、ソフトウェアコンポーネントに対してオンデマンドで配布され、中央ロケーションでローテーションが発生する。
- 認証情報へのアクセスは、非常にきめ細やかに制御できます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

従来、データベースやサードパーティーの API、トークンなどの認証に使用する認証情報は、ソースコードや環境ファイルに埋め込まれている場合がありました。AWS は、これらの認証情報を安全に保管し、自動的にローテーションし、その使用を監査するメカニズムを複数提供しています。

シークレット管理に対する最善のアプローチは、削除、置換、ローテーションのガイダンスに従うことです。最も安全な認証情報は、保管、管理、処理が不要なものです。認証情報によっては、ワークロードの機能にとって不要となった、安全に削除できるものもあります。

ワークロードの正常な機能に依然として必要な認証情報については、長期的認証情報を一時的または短期的な認証情報と置換する機会があるかもしれません。例えば、AWS シークレットアクセスキーをハードコーディングする代わりに、IAM ロールを使って長期的認証情報を一時的認証情報と置換することを検討してみてください。

存続期間の長いシークレットによっては、削除も置換もできないものがあります。これらのシークレットは、AWS Secrets Manager などのサービスに保管して、一元的に保管、管理したり、定期的にローテーションしたりすることができます。

ワークロードのソースコードと設定ファイルの監査を行うと、さまざまなタイプの認証情報が明らかになる可能性があります。次の表は、一般的なタイプの認証情報を取り扱うための戦略をまとめたものです。

認証情報のタイプ	説明	推奨される戦略
IAM アクセスキー	ワークロード内で IAM ロールを引き受けるために使用される AWS IAM アクセスとシークレットキー	置き換え: 代わりに、コンピューティングインスタンス (Amazon EC2 や AWS Lambda など) に割り当てられた IAM ロールを使用します。AWS アカウント内のリソースへのアクセスを必要とするサードパーティーとの相互運用性については、AWS クロスアカウントアクセスをサポートしているかどうかを確認してください。モバイルアプリの場合は、Amazon Cognito アイデンティティプール (フェデレーティッド ID) を介して一時的な認証情報を使用することを検討してください。AWS の外部で実行されているワークロードについては、IAM Roles Anywhere または AWS Systems Manager ハイブリッドアクティベーションを検討してください。コン

認証情報のタイプ	説明	推奨される戦略
		テナについては、「Amazon ECS タスクの IAM ロール」または「Amazon EKS ノードの IAM ロール」を参照してください。
SSH キー	Linux EC2 インスタンスへの手動または自動プロセスの一環としてのログインに使用されるセキュアシェルプライベートキー	置き換え: AWS Systems Manager または EC2 Instance Connect を使用して、IAM ロールを使用して EC2 インスタンスへのプログラムおよび人間によるアクセスを提供します。
アプリケーションとデータベースの認証情報	パスワード – プレーンテキスト文字列	ローテーション: 認証情報を AWS Secrets Manager に保存し、可能であれば自動ローテーションを確立します。
Amazon RDS と Aurora 管理データベースの認証情報	パスワード – プレーンテキスト文字列	置き換え: Secrets Manager と Amazon RDS または Amazon Aurora の統合を使用します。さらに、一部の RDS データベースタイプでは、一部のユースケースでパスワードの代わりに IAM ロールを使用できます (詳細については、「IAM データベース認証」を参照してください)。
OAuth トークン	シークレットトークン – プレーンテキスト文字列	ローテーション: トークンを AWS Secrets Manager に保存し、自動ローテーションを設定します。

認証情報のタイプ	説明	推奨される戦略
API トークンとキー	シークレットトークン – プレーンテキスト文字列	ローテーション: AWS Secrets Manager に保存し、可能であれば自動ローテーションを確立します。

一般的なアンチパターンは、ソースコード、設定ファイル、またはモバイルアプリ内に IAM アクセスキーを埋め込むことです。AWS サービスと通信するために IAM アクセスキーが必要な場合は、一時的な (短期的な) セキュリティ認証情報を使用します。これらの短期認証情報は、EC2 インスタンスの IAM ロール、Lambda 関数の実行ロール、モバイルユーザーアクセス用の Cognito IAM ロール、および IoT デバイス用の IoT Core ポリシーを通じて提供できます。サードパーティーとやり取りする場合は、IAM ユーザーをサーバーして、サードパーティーにそのユーザー向けのシークレットアクセスキーを送信するよりも、アカウントのリソースへの必要なアクセス権を持つ IAM ロールにアクセスを委譲する方法を優先します。

ワークロードに、他のサービスやリソースとの相互運用に必要なシークレットの保管が必要となるケースが多数あります。AWS Secrets Manager は、これらの認証情報の安全な管理、さらには API トークン、パスワード、およびその他の認証情報の保管、使用、ローテーション専用です。

AWS Secrets Manager には、機密認証情報の安全なストレージと処理を確保するための 5 つの主要な機能があります。保管時の暗号化、転送中の暗号化、包括的な監査、きめ細かなアクセスコントロール、および拡張可能な認証情報ローテーションです。AWS パートナーによるその他のシークレット管理サービス、または類似の機能や保証を提供するローカルで開発されたソリューションも使用できます。

シークレットを取得するときに、Secrets Manager のクライアント側のキャッシュコンポーネントを使用して、将来使用するためにキャッシュすることができます。キャッシュされたシークレットの取得は、Secrets Manager からの取得よりも高速です。さらに、Secrets Manager API を呼び出すにはコストがかかるため、キャッシュを使用するとコストを削減できます。シークレットを取得するすべての方法については、「シークレットの取得」を参照してください。

> **Note**
>
> 一部の言語では、クライアント側のキャッシュ用に独自のインメモリ暗号化を実装する必要がある場合があります。

実装手順

1. Amazon CodeGuru などの自動ツールを使用して、ハードコードされた認証情報を含むコードパスを特定します。
   a. Amazon CodeGuru を使って、コードリポジトリをスキャンします。レビューが完了したら、CodeGuru で Type=Secrets をフィルタリングして、問題のあるコード行を見つけます。
2. 削除または置換できる認証情報を特定します。
   a. 既に不要な認証情報を特定して、削除用にマークします。
   b. ソースコードに埋め込まれた AWS シークレットキーについては、必要なリソースに関連付けられた IAM ロールと置換します。ワークロードの一部が AWS 外であるにもかかわらず AWS リソースにアクセスする IAM 認証情報が必要な場合、IAM Roles Anywhere または AWS Systems Manager ハイブリッドアクティベーションを検討してください。
3. ローテーション戦略を使用すべきその他のサードパーティー、存続期間の長いシークレットについては、Secrets Manager をコードに統合して、ランタイムにサードパーティーのシークレットを取得します。
   a. CodeGuru コンソールは、検出された認証情報を使って Secrets Manager にシークレットを作成できます。
   b. Secrets Manager から取得したシークレットをアプリケーションコードに統合します。
      i. サーバーレス Lambda 関数では、言語に依存しない Lambda 拡張機能を使用できます。
      ii. EC2 インスタンスまたはコンテナに対しては、AWS が複数のよく使用されるプログラミング言語で、Secrets Manager からシークレットを取得するためのクライアント側コードの例を提供しています。
4. 定期的にコードベースをレビューして再スキャンすることで、コードに新たなシークレットが追加されていないことを確認します。
   a. git-secrets などのツールを使って、ソースコードリポジトリに新しいシークレットがコミットされるのを防止することを検討してください。
5. 予想外の使用、不適切なシークレットへのアクセス、またはシークレットの削除試行がないかどうか、Secrets Manager アクティビティをモニタリングします。
6. 認証情報に対する人的曝露を減少させます。この目的に特化した IAM ロールに対する認証情報を読み出し、書き込み、および変更するためのアクセスを制限し、一部の運用ユーザーにのみ、その役割を担うためのアクセスを提供します。

リソース

関連するベストプラクティス:

- SEC02-BP02 一時的な認証情報を使用する
- SEC02-BP05 定期的に認証情報を監査およびローテーションする

関連ドキュメント:

- AWS Secrets Manager の開始方法
- ID プロバイダーとフェデレーション
- Amazon CodeGuru Introduces Secrets Detector
- AWS Secrets Manager での AWS Key Management Service の使用方法
- Secret encryption and decryption in Secrets Manager
- Secrets Manager ブログエントリ
- Amazon RDS と AWS Secrets Manager の統合を発表

関連動画:

- Best Practices for Managing, Retrieving, and Rotating Secrets at Scale
- Find Hard-Coded Secrets Using Amazon CodeGuru Secrets Detector
- Securing Secrets for Hybrid Workloads Using AWS Secrets Manager

関連するワークショップ:

- Store, retrieve, and manage sensitive credentials in AWS Secrets Manager
- AWS Systems Manager のハイブリッドアクティベーション

SEC02-BP04 一元化された ID プロバイダーを利用する

ワークフォースユーザー ID (従業員と契約社員) の場合、ID を一元管理できる ID プロバイダーを利用します。一つの場所から権限の作成、割り当て、管理、取り消し、監査を行うため、複数のアプリケーションおよびシステムにまたがる権限を効率的に管理できます。

期待される成果: 一元化された ID プロバイダーを使用して、ワークフォースユーザー、認証ポリシー (多要素認証 (MFA) の要求など)、システムやアプリケーションへの承認 (ユーザーのグループメンバーシップや属性に基づくアクセスの割り当てなど) を一元管理します。ワークフォースユーザーは一元化された ID プロバイダーにサインインし、内部アプリケーションと外部アプリケーションにフェデレーション (シングルサインオン) します。これにより、ユーザーは複数の認証情報を覚えておく必要がなくなります。ID プロバイダーは人事 (HR) システムと統合されているため、人事上の変更は ID プロバイダーと自動的に同期されます。例えば、誰かが組織を離れた場合、フェデレーションされたアプリケーションやシステム (AWS を含む) へのアクセスを自動的に取り消すことができます。ID プロバイダーで詳細な監査ログを有効にし、これらのログでユーザーの異常な行動がないか監視します。

一般的なアンチパターン:

- フェデレーションとシングルサインオンを使用しない。ワークフォースユーザーが、複数のアプリケーションやシステムで個別のユーザーアカウントと認証情報を作成する。
- ID プロバイダーを人事システムに統合するなど、ワークフォースユーザーのアイデンティティのライフサイクルを自動化していない。ユーザーが組織を離れたり、役割を変更したりした場合に、複数のアプリケーションやシステムのレコードを手動のプロセスで削除または更新する。

このベストプラクティスを活用するメリット: 一元化された ID プロバイダーを使用することで、ワークフォースユーザーのアイデンティティとポリシーを 1 か所で管理でき、ユーザーやグループにアプリケーションへのアクセス権を割り当てたり、ユーザーのサインインアクティビティを監視したりできます。人事 (HR) システムと統合することで、ユーザーの役割が変更された場合は、これらの変更が ID プロバイダーと同期され、ユーザーに割り当てられたアプリケーションと権限が自動的に更新されます。ユーザーが組織を離れると、そのユーザーのアイデンティティは ID プロバイダーで自動的に無効になり、フェデレーションアプリケーションおよびシステムへのアクセス権が取り消されます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

AWS にアクセスするワークフォースユーザー向けのガイダンス組織内の従業員や契約社員などのワークフォースユーザーは、AWS マネジメントコンソール または AWS Command Line Interface (AWS CLI) を使って職務を遂行するため、AWS へのアクセス権を必要とする場合があります。一元化された ID プロバイダーから 2 つのレベルで AWS にフェデレーションすることで、ワークフォースユーザーに AWS へのアクセス権を付与できます。1 つは各 AWS アカウントへの直接フェデレーション、もう 1 つは AWS 組織内の複数のアカウントへのフェデレーションです。

ワークフォースユーザーをそれぞれの AWS アカウントと直接フェデレーションするには、一元化された ID プロバイダーを使用して、そのアカウントの AWS Identity and Access Management にフェデレーションできます。IAM の柔軟性により、SAML 2.0 または Open ID Connect (OIDC) という別々の ID プロバイダーを各 AWS アカウントで有効にして、アクセスコントロールにはフェデレーションユーザー属性を使用することができます。ワークフォースユーザーはウェブブラウザを使用し、認証情報 (パスワードや MFA トークンコードなど) を入力して ID プロバイダーにサインインします。ID プロバイダーは、AWS マネジメントコンソールのサインイン URL に送信される SAML アサーションをユーザーのブラウザに発行して、IAM ロールを引き受けることで、ユーザーが AWS マネジメントコンソールにシングルサインオンできるようにします。ユーザーは、ID プロバイダーからの SAML アサーションを使用して、IAM ロールを引き受けることで、AWS CLI や AWS STS の AWS SDK で使用する 一時的な AWS API 認証情報を 取得することもできます。

ワークフォースユーザーを AWS Organization の複数のアカウントにフェデレーションするには、AWS IAM アイデンティティセンター を使用して、AWS アカウント やアプリケーションへのワークフォースユーザーのアクセスを一元管理できます。組織のアイデンティティセンターを有効にし、ID ソースを設定します。IAM Identity Center は、ユーザーやグループの管理に使用できるデフォルトの ID ソースディレクトリを提供します。または、SAML 2.0 を使用して外部 ID プロバイダーに接続し、 SCIM を使用してユーザーとグループを自動的にプロビジョニングするか、または Directory Service を使用して Microsoft AD Directory に接続することで、外部 ID ソースを選択することもできます。ID ソースを設定したら、アクセス許可セットで最小権限ポリシーを定義して、ユーザーとグループに AWS アカウントへのアクセス権を割り当てることができます。ワークフォースユーザーは一元化された ID プロバイダーを通じて認証を行い、AWS アクセスポータルにサインインして、自分に割り当てられた AWS アカウントとクラウドアプリケーションにシングルサインオンします。ユーザーは AWS CLI v2 を設定して、アイデンティティセンターで認証を行い、AWS CLI コマンドを実行するための認証情報を取得できます アイデンティティセンターでは、AWS アプリケーション (Amazon SageMaker AI Studio や AWS IoT Sitewise Monitor ポータルなど) へのアクセスにシングルサインオンも使用できます。

前述のガイダンスに従うと、ワークフォースユーザーは AWS でワークロードを管理する際、通常の操作で IAM ユーザーおよびグループを使用する必要がなくなります。管理するのではなく、ユーザーとグループは AWS の外部で管理され、ユーザーはフェデレーション ID として AWS リソースにアクセスできます。フェデレーション ID では、一元化された ID プロバイダーで定義されたグループを使用します。AWS アカウントで不要になった IAM グループ、IAM ユーザー、および永続的なユーザー認証情報 (パスワードとアクセスキー) を特定して削除する必要があります。また、IAM 認証情報レポートを使用して、未使用の認証情報を検索し、該当する IAM ユーザーや IAM グループを削除できます。 組織にサービスコントロールポリシー (SCP) を適用して、新しい IAM ユーザーや

グループが作成されないようにし、フェデレーション ID を介した AWS へのアクセスを強制できます。

> **Note**
>
> SCIM アクセストークンのローテーションの処理については、自動プロビジョニングのドキュメントに記載されているようにユーザーが行う必要があります。さらに、ID フェデレーションをサポートする証明書のローテーションは、ユーザーが行う必要があります。

アプリケーションのユーザー向けガイダンスモバイルアプリなどのアプリケーションのユーザーの ID を管理するには、一元化された ID プロバイダーとして Amazon Cognito を使用できます。Amazon Cognito は、ウェブおよびモバイルアプリの認証、認可、およびユーザー管理を可能にします。Amazon Cognito は数百万人のユーザーにスケール可能な ID ストアを備え、ソーシャル ID フェデレーションとエンタープライズ ID フェデレーションをサポートし、ユーザーとビジネスの保護に役立つ高度なセキュリティ機能を提供します。カスタムのウェブまたはモバイルアプリケーションを Amazon Cognito と統合すると、アプリケーションへのユーザー認証とアクセスコントロールを数分で追加できます。SAML や Open ID Connect (OIDC) などのオープン ID 標準に基づいて構築された Amazon Cognito は、さまざまなコンプライアンス規制に対応し、フロントエンドおよびバックエンドの開発リソースと統合します。

実装手順

ワークフォースユーザーの AWS へのアクセス手順

- 以下のいずれかの方法を使用し、一元化された ID プロバイダーを使用して、ワークフォースユーザーを AWS にフェデレーションします。
  - IAM Identity Center を使用し、ID プロバイダーとフェデレーションすることで、AWS 組織内の複数の AWS アカウントへのシングルサインオンを有効にします。
  - IAM を使用して、ID プロバイダーを各 AWS アカウントに直接接続し、フェデレーションによるきめ細かいアクセスを可能にします。
- フェデレーション ID で置き換えられた IAM ユーザーとグループを特定して削除します。

アプリケーションのユーザー向けの手順

- アプリケーション用の一元化された ID プロバイダーとして Amazon Cognito を使用します。
- OpenID Connect と OAuth を使用して、カスタムアプリケーションを Amazon Cognito と統合します。認証のための Amazon Cognito など、さまざまな AWS サービスと統合するためのシンプル

なインターフェイスを提供する Amplify ライブラリを使用して、カスタムアプリケーションを開発できます。

リソース

関連するベストプラクティス:

- SEC02-BP06 ユーザーグループと属性を採用する
- SEC03-BP02 最小特権のアクセスを付与する
- SEC03-BP06 ライフサイクルに基づいてアクセスを管理する

関連ドキュメント:

- AWS での ID フェデレーション
- IAM でのセキュリティのベストプラクティス
- AWS Identity and Access Managementベストプラクティス
- Getting started with IAM Identity Center delegated administration
- How to use customer managed policies in IAM Identity Center for advanced use cases
- AWS CLI v2: IAM Identity Center credential provider

関連動画:

- AWS re:Inforce 2022 - AWS Identity and Access Management (IAM) deep dive
- AWS re:Invent 2022 - Simplify your existing workforce access with IAM Identity Center
- AWS re:Invent 2018: Mastering Identity at Every Layer of the Cake

関連する例:

- ワークショップ: Using AWS IAM Identity Center to achieve strong identity management

関連ツール:

- AWS セキュリティコンピテンシーパートナー: ID およびアクセスの管理
- saml2aws

SEC02-BP05 定期的に認証情報を監査およびローテーションする

認証情報を定期的に監査およびローテーションして、リソースへのアクセスに認証情報を使用できる期間を制限します。長期的認証情報を使用すると多くのリスクが生じますが、これらのリスクは長期的認証情報を定期的にローテーションすることで軽減できます。

期待される成果: 長期的認証情報の使用に関連するリスクを軽減するために、認証情報のローテーションを実装します。認証情報ローテーションポリシーの不遵守を定期的に監査して、是正します。

一般的なアンチパターン:

- 認証情報の使用を監査しない。
- 必要がないのに長期的認証情報を使用する。
- 長期的認証情報を使用しているが、定期的にローテーションしない。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

一時的な認証情報に頼れず、長期的認証情報が必要な場合は、認証情報を監査して、多要素認証 (MFA) などの定義された管理方法が実施され、定期的にローテーションされて、アクセスレベルが適切であることを確認する必要があります。

正しい管理方法が実施されていることを確認するには、定期的な検証、できれば自動化ツールによる検証が必要です。人的 ID の場合は、パスワードを定期的に変更し、一時的な認証情報を優先してアクセスキーを廃止するように、ユーザーに要求する必要があります。AWS Identity and Access Management (IAM) ユーザーから一元化された ID に移行すると、認証情報レポートを生成してユーザーを監査できます。

ID プロバイダーで MFA を実施およびモニタリングすることもお勧めします。AWS Config ルールを設定するか、AWS Security Hub CSPM セキュリティ標準を使用して、ユーザーが MFA を設定しているかどうかをモニタリングできます。IAM Roles Anywhere を使用して、マシン ID の一時的な認証情報を付与することを検討してください。IAM ロールと一時的な認証情報を使用できないときは、アクセスキーの監査とローテーションの頻度を高めることが重要です。

実装手順

- 認証情報を定期的に監査する: ID プロバイダーと IAM で設定されている ID を監査することで、認証された ID のみがワークロードにアクセスできるようになります。これらの ID は、IAM ユー

ザー、AWS IAM アイデンティティセンターユーザー、Active Directory ユーザー、またはさまざまなアップストリーム ID プロバイダーのユーザーを含みますが、これらに限定されるものではありません。例えば、組織を離れた人を削除したり、不要になったクロスアカウントのロールを削除したりします。IAM エンティティがアクセスするサービスへのアクセス許可を定期的に監査するプロセスを用意します。これにより、未使用のアクセス許可を削除するために変更する必要があるポリシーを特定できます。認証情報レポートと AWS Identity and Access Management Access Analyzer を使用して、IAM 認証情報とアクセス許可を監査します。Amazon CloudWatch を使用して、AWS 環境内で呼び出される特定の API コールのアラームを設定できます。Amazon GuardDuty は、IAM 認証情報へのアクセスが過度に許可されているか、意図しないアクセスを示している可能性のある、予期しないアクティビティを警告することもできます。

- 認証情報を定期的にローテーションする: 一時的な認証情報を使用できない場合は、IAM アクセスキーを定期的に (最大 90 日ごとに) ローテーションします。知らない間にアクセスキーが開示された場合でも、ローテーションを行うことで、該当する認証情報を使用してリソースにアクセスされる期間を制限できます。アクセスキーのローテーションの詳細については、IAM ユーザーの「アクセスキーのローテーション」を参照してください。
- IAM アクセス許可を確認する: AWS アカウントのセキュリティを改善するには、各 IAM ポリシーを定期的に確認してモニタリングします。ポリシーが最小特権の原則に従っていることを確認します。
- IAM リソースの作成と更新の自動化を検討する: IAM アイデンティティセンターは、ロールやポリシーの管理など、多くの IAM タスクを自動化します。または、AWS CloudFormation を使用することで、テンプレートを検証してバージョンを管理できるため、ロールやポリシーを含む IAM リソースのデプロイを自動化して、人為的ミスが生じる可能性を軽減できます。
- IAM Roles Anywhere を使用してマシン ID の IAM ユーザーを置き換える: IAM Roles Anywhere を使用すると、オンプレミスサーバーなど、従来は使用できなかった領域でロールを使用できます。IAM Roles Anywhere は、信頼された X.509 証明書を使用して AWS を認証し、一時的な認証情報を受け取ります。IAM Roles Anywhere を使用することで、長期的認証情報がオンプレミス環境に保存されなくなるため、これらの認証情報をローテーションする必要がなくなります。X.509 証明書の有効期限が近づいたら、モニタリングとローテーションが必要となることに注意してください。

リソース

関連するベストプラクティス:

- SEC02-BP02 一時的な認証情報を使用する
- SEC02-BP03 シークレットを安全に保存して使用する

関連ドキュメント:

- AWS Secrets Manager の開始方法
- IAM ベストプラクティス
- ID プロバイダーとフェデレーション
- セキュリティパートナーソリューション: アクセスおよびアクセスコントロール
- 一時的な認証情報
- AWS アカウント の認証情報レポートの取得

関連動画:

- シークレットを大規模に管理、取得、変更するためのベストプラクティス
- Managing user permissions at scale with AWS IAM Identity Center
- Mastering identity at every layer of the cake

SEC02-BP06 ユーザーグループと属性を採用する

ユーザーグループと属性に従ってアクセス許可を定義すると、ポリシーの数と複雑度が軽減され、最小特権の原則を簡単に遵守できます。ユーザーグループを使用して、多数のユーザーのアクセス許可をそれぞれが組織内で果たす職務に基づいて 1 か所で管理できます。部門、プロジェクト、または場所などの属性は、ユーザーが同じような職務を果たすが、対象となるリソースのサブセットが異なる場合に、アクセス許可の範囲をさらに限定することができます。

期待される成果: 権限の変更を、職務に基づき、その職務を実行するすべてのユーザーに適用できます。グループのメンバーシップと属性によってユーザーのアクセス許可が管理されるため、個々のユーザーレベルでアクセス許可を管理する必要がなくなります。ID プロバイダー (IdP) で定義したグループと属性が、AWS 環境に自動的に反映されます。

一般的なアンチパターン:

- 個々のユーザーのアクセス許可を管理し、複数のユーザーで重複作業をしている。
- グループの定義が大まか過ぎるため、アクセス許可の付与範囲が広過ぎる。
- グループの定義が細か過ぎるため、メンバーシップに関する重複や混乱が生じている。
- 代わりに属性を使用できる場面でグループを使用し、リソースの複数のサブセットに対してグループが持つアクセス許可が重複している。

- 標準に準拠した ID プロバイダーの AWS 環境への統合によるグループ、属性、メンバーシップの管理を行っていない。
- AWS IAM アイデンティティセンターセッションを使用する際のロールチェーンの使用

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

AWS アクセス許可は、ユーザー、グループ、ロール、リソースなどのプリンシパルに関連付けられている、ポリシーと呼ばれるドキュメントで定義されます。職務、ワークロード、および SDLC 環境に基づいてアクセス許可の割り当て (グループ、アクセス許可、アカウント) を整理することにより、アクセス許可管理をスケールすることができます。従業員に対しては、アクセス対象のリソースではなく、ユーザーが組織で果たす職務に基づいてグループを定義できます。例えば、WebAppDeveloper グループには、開発アカウント内で Amazon CloudFront などのサービスを設定するためのポリシーがアタッチされている場合があります。`AutomationDeveloper` グループには、`WebAppDeveloper` グループと重複するアクセス許可が付与されている場合があります。これらの共通のアクセス許可は、両方の職務のユーザーを `CloudFrontAccess` グループに所属させるのではなく、別々のポリシーで取得して両方のグループに関連付けることができます。

グループに加え、属性を使用してアクセスの範囲をさらに絞り込むことができます。例えば、`WebAppDeveloper` グループ内のユーザーに対して、プロジェクト固有のリソースへのアクセス範囲を限定するためのプロジェクト属性を設定できます。この手法を使用すれば、異なるプロジェクトで作業するアプリケーション開発者に対して、それ以外の点ではアクセス許可が同じ場合、別々のグループを用意する必要がなくなります。アクセス許可ポリシーで属性を参照する方法は、そのソースがフェデレーションプロトコル (SAML、OIDC、SCIM など) の一部として定義されているか、カスタム SAML アサーションとして定義されているか、IAM Identity Center 内で設定されているかに応じて異なります。

実装手順

1. グループと属性を定義する場所を確保します。
    a. SEC02-BP04 一元化された ID プロバイダーを利用するのガイダンスに従って、ID プロバイダー内、IAM アイデンティティセンター内、または特定のアカウント内の IAM ユーザーグループを使用し、グループと属性を定義する必要があるかどうかを判断します。
2. グループの定義:
    a. 必要な職務とアクセス範囲に基づいてグループを決定します。グループを効果的に整理するために、階層構造や命名規則を使用することを検討してください。

b. IAM Identity Center 内で定義する場合は、グループを作成し、アクセス許可セットを使用して、目的とするレベルのアクセス許可を関連付けます。

c. 外部の ID プロバイダー内で定義する場合は、プロバイダーが SCIM プロトコルをサポートしているかどうかを確認し、IAM Identity Center 内で自動プロビジョニングを有効にすることを検討してください。この機能は、プロバイダーと IAM Identity Center との間で、メンバーシップの作成、およびグループの削除を同期します。

3. 属性の定義:

a. 外部の ID プロバイダーを使用する場合、SCIM プロトコルと SAML 2.0 プロトコルは両方とも一部の属性をデフォルトで提供します。追加の属性は、`https://aws.amazon.com/SAML/Attributes/PrincipalTag` 属性名と SAML アサーションを使用して定義し、渡すことができます。カスタム属性の定義と設定に関するガイダンスについては、ID プロバイダーのドキュメントを参照してください。

b. IAM アイデンティティセンター内でロールを定義する場合は、属性ベースのアクセス制御 (ABAC) 機能を有効にし、必要に応じて属性を定義します。組織の構造またはリソースのタグ付け方法に沿った属性を検討してください。

IAM アイデンティティセンターで割り当てられた IAM ロールから IAM ロールチェーンが必要な場合は、`source-identity` や `principal-tags` などの値は反映されません。詳細については、「アクセスコントロールのための属性の有効化と設定」を参照してください。

1. グループと属性に基づいて、アクセス許可の範囲を設定します。

a. プリンシパルの属性と、アクセス対象のリソースの属性を比較する条件をアクセス許可ポリシーに含めることを検討してください。例えば、`PrincipalTag` 条件キーの値が同じ名前の `ResourceTag` キーの値と一致する場合にのみ、リソースへのアクセスを許可する条件を定義できます。

b. ABAC ポリシーを定義する場合は、ABAC 認可のベストプラクティスと例のガイダンスに従ってください。

c. 組織のニーズの変化に応じてグループと属性の構造を定期的に確認し、更新して、最適な権限管理を確保します。

リソース

関連するベストプラクティス:

- SEC02-BP04 一元化された ID プロバイダーを利用する

- SEC03-BP02 最小特権のアクセスを付与する
- COST02-BP04 グループとロールを実装する

関連ドキュメント:

- IAM ベストプラクティス
- IAM Identity Center で ID を管理する
- AWS の ABAC とは
- IAM Identity Center - ABAC
- ABAC ポリシーの例

関連動画:

- Managing user permissions at scale with AWS IAM Identity Center
- Mastering identity at every layer of the cake

権限の管理

アクセス許可を管理して、AWS とワークロードへのアクセスを必要とするユーザー ID やマシン ID へのアクセスを制御します。権限を使用すると、誰が何に、どのような条件でアクセスできるかを制御できます。特定のユーザー ID およびマシン ID にアクセス権限を設定し、必要とするリソースに対するサービスアクションへのアクセスのみを許可します。さらに、アクセスを取得するために満たすべき条件を指定できます。

さまざまなタイプのリソースにアクセスを付与する方法は多数あります。その 1 つは、異なるポリシータイプを使用する方法です。

IAM のアイデンティティベースのポリシーは管理またはインラインで、ユーザー、グループ、ロールなどの IAM ID にアタッチされます。これらのポリシーを使用すると、そのアイデンティティが実行できる内容 (そのアクセス許可) を指定できます。アイデンティティベースのポリシーはさらに分類できます。

管理ポリシー – AWS アカウント内の複数のユーザー、グループ、およびロールにアタッチできるスタンドアロンのアイデンティティベースのポリシーです。管理ポリシーには 2 種類あります。

- AWS 管理ポリシー – AWS が作成および管理する管理ポリシー。

- カスタマー管理ポリシー – AWS アカウントで作成および管理する管理ポリシー。カスタマー管理ポリシーでは、AWS マネージドポリシーよりも正確にポリシー管理できます。

アクセス許可を付与するには、マネージドポリシーのほうが好ましい方法です。ただし、単一のユーザー、グループ、ロールに直接追加するインラインポリシーを使用することもできます。インラインポリシーは、ポリシーと ID の間の厳密な 1 対 1 の関係を維持します。アイデンティティを削除すると、インラインポリシーは削除されます。

ほとんどの場合、最小特権の原則に従って独自のカスタマー管理ポリシーを作成する必要があります。

リソースベースのポリシーをリソースにアタッチします。例えば、Amazon S3 バケットポリシーはリソースベースのポリシーです。これらのポリシーでは、リソースと同じアカウントまたは別のアカウントにあるプリンシパルにアクセス許可を付与します。リソースベースのアクセス許可をサポートするサービスのリストについては、「IAM と連携する AWS のサービス」を参照してください。

アクセス許可の境界は、マネージドポリシーを使用して、管理者が設定できるアクセス許可の上限を設定できます。これによって、IAM ロール作成などのアクセス許可の作成および管理の権限を開発者に委任しながらも、付与できるアクセス許可を制限して、自分でそのアクセス許可の範囲を拡大できないように制限できます。

AWS の属性ベースのアクセス制御 (ABAC) を使用すると、タグと呼ばれる属性に基づいて権限を付与できます。これらのタグは、IAM プリンシパル (ユーザーまたはロール) と AWS リソースにアタッチできます。管理者は、IAM プリンシパルの属性に基づいて権限を適用する再利用可能な IAM ポリシーを作成できます。例えば、管理者は 1 つの IAM ポリシーを使用して、プロジェクトタグに一致する AWS リソースへのアクセス権を組織内のデベロッパーに付与できます。デベロッパーチームがプロジェクトにリソースを追加すると、属性に基づいて権限が自動的に適用されるため、新しいリソースごとにポリシーを更新する必要がなくなります。

組織 のサービスコントロールポリシー (SCP) を使用して、組織または組織単位 (OU) のメンバーアカウントのアクセス許可の上限を定義します。SCP では、アイデンティティベースのポリシーまたはリソースベースのポリシーで、アカウント内のエンティティ (ユーザーまたはロール) に付与するアクセス許可が制限されますが、アクセス許可は付与されません。

セッションポリシーは、ロールまたはフェデレーションユーザーを引き受けます。AWS CLI または AWS API セッションポリシーを使ってロールまたはユーザーのアイデンティティベースのポリシーがセッションに付与する許可を制限する際、セッションポリシーを渡します。セッションポリシーでは、作成したセッションのアクセス許可が制限されますが、アクセス許可は付与されません。詳細については、「セッションポリシー」を参照してください。

ベストプラクティス

- SEC03-BP01 アクセス要件を定義する
- SEC03-BP02 最小特権のアクセスを付与します
- SEC03-BP03 緊急アクセスのプロセスを確立する
- SEC03-BP04 アクセス許可を継続的に削減する
- SEC03-BP05 組織のアクセス許可ガードレールを定義する
- SEC03-BP06 ライフサイクルに基づいてアクセスを管理する
- SEC03-BP07 パブリックおよびクロスアカウントアクセスの分析
- SEC03-BP08 組織内でリソースを安全に共有する
- SEC03-BP09 サードパーティーとリソースを安全に共有する

SEC03-BP01 アクセス要件を定義する

ワークロードの各コンポーネントやリソースには、管理者、エンドユーザー、またはその他のコンポーネントによるアクセスが必要です。各コンポーネントへのアクセス権のある人とマシンを明確に定義し、適切な ID のタイプと、認証および認可の方法を選択します。

一般的なアンチパターン:

- シークレットをハードコーディングする、またはアプリケーション内に格納する
- 各ユーザーにカスタムのアクセス許可を付与する
- 永続的な認証情報を使用する

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

ワークロードの各コンポーネントやリソースには、管理者、エンドユーザー、またはその他のコンポーネントによるアクセスが必要です。各コンポーネントへのアクセス権のある人とマシンを明確に定義し、適切な ID のタイプと、認証および認可の方法を選択します。

組織内の AWS アカウントへの定期的なアクセスは、フェデレーションアクセスか一元化された ID プロバイダーを使用して提供する必要があります。また、アイデンティティ管理を一元化し、AWS へのアクセスを従業員のアクセスライフサイクルに統合するための確立されたプラクティスを整備する必要があります。例えば、従業員がアクセスレベルの異なる職種に異動するときは、そのグループメンバーシップも新しいアクセス要件を反映するように変更される必要があります。

非人間アイデンティティのアクセス要件を定義するときは、どのアプリケーションとコンポーネントがアクセスを必要としているか、またアクセス許可をどのように付与するかを決定します。推奨されるアプローチは、最小特権アクセスモデルで構築されたロールを使用する方法です。AWSマネージドポリシーは、最も一般的なユースケースをカバーする、事前定義済みの IAM ポリシーを提供します。

AWS Secrets Manager や AWS Systems Manager Parameter Store などの AWS のサービスは、アプリケーションまたはワークロードからシークレットを安全に分離するのに役立ちます。Secrets Manager では、認証情報の自動ローテーションを確立できます。Secrets Manager でパラメータの作成時に指定した一意の名前を使用することで、スクリプト、コマンド、SSM ドキュメント、設定、自動化ワークフロー内のパラメータを参照できます。

AWS IAM Roles Anywhere を使用すると、AWS の外部で実行されるワークロード IAM における一時的セキュリティ認証情報を取得できます。ワークロードでは、AWS アプリケーションが AWS リソースにアクセスする際に使用するのと同じ IAM ポリシーと IAM ロールを使用できます。

可能な場合は、長期の静的な認証情報よりも、短期の一時的な認証情報を優先します。プログラムによるアクセスと長期的認証情報を持つユーザーが必要なシナリオでは、アクセスキーが最後に使用した情報を使用して、アクセスキーをローテーションおよび削除します。

AWS マネジメントコンソール の外部で AWS を操作するには、ユーザーはプログラムによるアクセスが必要です。プログラマチックアクセス権を付与する方法は、AWS にアクセスしているユーザーのタイプによって異なります。

ユーザーにプログラマチックアクセス権を付与するには、以下のいずれかのオプションを選択します。

プログラマチックアクセス権を必要とするユーザー	目的	方法
IAM	(推奨) 一時的な認証情報としてコンソール認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラマチックリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、「AWS Command Line Interface ユーザーガイド」の「AWS ローカル開発用のログイン」を参照してください。

プログラマチックアクセス権を必要とするユーザー	目的	方法
		• AWS SDK については、「AWS SDK およびツールリファレンスガイド」の「AWS ローカル開発用のログイン」を参照してください。
ワークフォースアイデンティティ (IAM アイデンティティセンターで管理されているユーザー)	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラマチックリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、AWS Command Line Interface ユーザーガイドの「AWS IAM アイデンティティセンター を使用するための AWS CLI の設定」を参照してください。 • AWS SDK、ツール、および AWS API については、AWS SDK とツールリファレンスガイドの「IAM Identity Center 認証」を参照してください。
IAM	一時的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	IAM ユーザーガイドの「AWS リソースでの一時的な認証情報の使用」の指示に従ってください。

プログラマチックアクセス権を必要とするユーザー	目的	方法
IAM	(非推奨) 長期的な認証情報を使用して、AWS CLI、AWS SDK、または AWS API へのプログラムによるリクエストに署名します。	使用するインターフェイスの指示に従ってください。 • AWS CLI については、AWS Command Line Interface ユーザーガイドの「IAM ユーザー認証情報を使用した認証」を参照してください。 • AWS SDK とツールについては、AWS SDK とツールリファレンスガイドの「長期認証情報を使用して認証する」を参照してください。 • AWS API については、IAM ユーザーガイドの「IAM ユーザーのアクセスキーの管理」を参照してください。

リソース

関連ドキュメント:

- 属性ベースのアクセスコントロール (ABAC)
- AWS IAM アイデンティティセンター
- IAM Roles Anywhere
- AWS Managed policies for IAM Identity Center
- AWS IAM ポリシー条件
- IAM のユースケース
- 不要な認証情報の削除

- IAM ポリシーを管理する
- How to control access to AWS resources based on AWS アカウント, OU, or organization
- Identify, arrange, and manage secrets easily using enhanced search in AWS Secrets Manager

関連動画:

- Become an IAM Policy Master in 60 Minutes or Less
- Separation of Duties, Least Privilege, Delegation, and CI/CD
- Streamlining identity and access management for innovation

SEC03-BP02 最小特権のアクセスを付与します

ユーザーが特定の条件下で特定のリソースに対する特定のアクションを実行するために、必要なアクセス許可のみを付与します。グループと ID 属性を使用して、個々のユーザーのアクセス許可を定義するのではなく、規模に応じてアクセス許可を動的に設定します。例えば、デベロッパーのグループに、プロジェクトのリソースのみを管理するためのアクセスを許可することができます。これにより、デベロッパーがプロジェクトを離れると、基盤となるアクセスポリシーに変更を加えることなく、そのデベロッパーのアクセスは自動的に取り消されます。

期待される成果: ユーザーには、それぞれの職務に必要な最低限のアクセス許可のみが付与されます。デベロッパーを本番環境から分離するには、別個の AWS アカウント を使用します。デベロッパーが特定のタスクの本番環境にアクセスする必要がある場合、それらのタスクの期間中のみ、管理された制限付きのアクセス許可が付与されます。本番環境へのアクセスは、必要な作業が完了するとすぐに取り消されます。アクセス許可の定期的な見直しを行い、ユーザーがロールを変更したり組織を離れたりするなど、不要になったらすぐに取り消します。管理者権限を信頼できる小さなグループに限定することにより、リスクを最小限に抑えます。マシンまたはシステムのアカウントには、意図したタスクの実行に必要な最小限のアクセス許可のみを付与します。

一般的なアンチパターン:

- デフォルトで、ユーザーに管理者のアクセス許可を付与する。
- 通常のアクティビティには、ルートユーザーアカウントを使用する。
- 適切な範囲を指定せずに、過度に許容されたポリシーを作成する。
- アクセス許可のレビューは頻繁に行われないため、アクセス許可のクリープが発生する。
- 環境の分離やアクセス許可の管理には、属性ベースのアクセスコントロールのみに依存している。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

最小特権の原則では、特定のタスクを実行するために必要な、最小限のアクションの実行のみを ID に許可する必要があると規定されています。これは、ユーザビリティ、効率性、セキュリティのバランスを取ります。この原則の下に運用すると、意図しないアクセスを制限し、誰がどのリソースにアクセスできるかを追跡するのに役立ちます。デフォルトでは、IAM ユーザーやロールにアクセス許可はありません。ルートユーザーにはデフォルトでフルアクセスがありますが、厳密に制御、モニタリングされ、ルートアクセスを必要とするタスクにのみ使用する必要があります。

IAM ポリシーは、IAM ロールまたは特定のリソースに明示的にアクセス許可を付与するために使用します。例えば、アイデンティティベースのポリシーは IAM グループにアタッチでき、S3 バケットはリソースベースのポリシーで制御できます。

IAM ポリシーの作成時に、サービスアクション、リソース、および AWS がアクセスを許可または拒否するために true が必須な条件を指定できます。AWS は、アクセスの範囲を絞り込むのに役立つさまざまな条件をサポートしています。例えば、PrincipalOrgID 条件キーを使用すると、リクエスタが AWS Organization の一部でない場合にアクションを拒否できます。

また、CalledVia 条件キーを使用して、AWS Lambda 関数を作成する AWS CloudFormation など、AWS サービスがユーザーに代わって行うリクエストを制御できます。異なるポリシータイプを層にして深層防御を確立し、ユーザーの全体的なアクセス許可を制限できます。どのアクセス許可がどのような条件の下で付与できるかも制限できます。例えば、ワークロードチームに対して、構築するシステムに独自の IAM ポリシーを作成することを許可できますが、その場合、付与できるアクセス許可の上限を制限するアクセス許可の境界を適用する必要があります。

実装手順

- 最小特権ポリシーを実装する: IAM グループおよびロールに最小特権のアクセスポリシーを割り当てて、定義したユーザーのロールまたは機能を反映させます。
- 別個の AWS アカウント を使用して開発環境と本番環境を分離する: 開発環境と本番環境には別個の AWS アカウント を使用し、サービスコントロールポリシー 、リソースポリシー、アイデンティティポリシーを使用して、開発環境と本番環境間のアクセスを制御します。
- API の使用状況に基づくポリシー: 必要なアクセス許可を決定する 1 つの方法は、AWS CloudTrail ログを確認することです。このレビューを使用して、ユーザーが AWS 内で実際に実行するアクションに合わせて、カスタマイズしたアクセス許可を作成できます。IAM Access Analyzer は、アクセスアクティビティに基づいて IAM ポリシーを自動生成できます。組織またはアカウントレベルで IAM Access Advisor を使用し、特定のポリシーの最終アクセス情報を追跡できます。

- ジョブ機能に AWS マネージドポリシーの使用を検討する: きめ細かいアクセス許可ポリシーの作成を開始する場合は、請求、データベース管理者、データサイエンティストなどの一般的なジョブのロールに AWS マネージドポリシーを使用することを推奨します。これらのポリシーは、最小特権ポリシーの実装方法を決定する際に、ユーザーの持つアクセスを絞り込むことができます。
- 不要なアクセス許可を削除する: 未使用の IAM エンティティ、認証情報、アクセス許可を検出して削除し、最小特権の原則を実現します。IAM Access Analyzer を使用すると、外部アクセスと未使用のアクセスを識別でき、IAM Access Analyzer ポリシーの生成によって、アクセス許可ポリシーのファインチューニングを行いやすくなります。
- ユーザーの本番環境へのアクセスが制限されていることを確認する: ユーザーは、有効なユースケースを持つ本番環境にのみアクセスする必要があります。ユーザーが、本番稼働アクセスが必要な特定のタスクを実行した後は、アクセスを取り消す必要があります。本番環境へのアクセスを制限することは、本番に影響する意図しないイベントを回避するのに役立ち、意図しないアクセスの影響範囲を狭めます。
- アクセス許可の境界を考慮する: アクセス許可の境界は、アイデンティティベースのポリシーが IAM エンティティに付与できるアクセス許可の上限を設定する、マネージドポリシーを使用するための機能です。エンティティのアクセス許可の境界により、エンティティは、アイデンティティベースのポリシーとそのアクセス許可の境界の両方で許可されているアクションのみを実行できます。
- 属性ベースのアクセス制御とリソースタグを使用したアクセスの絞り込み サポートされている場合、リソースタグを使用した属性ベースのアクセス制御 (ABAC) を使用して、アクセス許可を絞り込むことができます。ABAC モデルを使用して、プリンシパルタグとリソースタグを比較し、定義したカスタムディメンションに基づいてアクセスを絞り込むことができます。このアプローチにより、組織内のアクセス許可ポリシーを簡素化し、その数を削減できます。
  - ABAC は、プリンシパルとリソースの両方が AWS Organization によって所有されている場合にのみ、アクセス制御に使用することを推奨します。外部関係者は、自分のプリンシパルとリソースに対して、組織と同じタグ名と値を使用できます。外部パーティのプリンシパルやリソースへのアクセス許可を付与する際に、これらの名前と値のペアのみに依存していると、意図しないアクセス許可を付与してしまう可能性があります。
- AWS Organizations のサービスコントロールポリシーを使用する: サービスコントロールポリシーは、組織のメンバーアカウントで利用できる最大のアクセス許可を一元管理します。重要なのは、サービスコントロールポリシーを使用すると、メンバーアカウントでルートユーザーのアクセス許可を制限できることです。AWS Organizations を強化する規範的マネージドコントロールを提供する、AWS Control Tower の使用も検討してください。Control Tower 内で独自のコントロールを定義できます。

- 組織のユーザーライフサイクルポリシーを確立する: ユーザーライフサイクルポリシーは、ユーザーが AWS にオンボーディングされたとき、ジョブロールまたはスコープを変更したとき、または AWS へのアクセスが不要になったときに実行するタスクを定義します。ユーザーのライフサイクルの各段階でアクセス許可のレビューを行い、アクセス許可が適切に制限されていることを検証して、アクセス許可のクリープを回避する必要があります。
- アクセス許可を見直して不要なアクセス許可を削除する: ユーザーアクセスを定期的に見直して、ユーザーに過剰なアクセス許可がないことを確認する必要があります。AWS Config および IAM Access Analyzer は、ユーザーのアクセス許可を監査する際に役立ちます。
- ジョブロールマトリックスを確立する: ジョブロールマトリックスは、AWS フットプリント内に必要なさまざまなロールとアクセスレベルを視覚化します。ジョブロールマトリックスにより、組織内でのユーザーの責任に基づいてアクセス許可を定義し、分離できます。個々のユーザーまたはロールにアクセス許可を直接適用するのではなく、グループを使用します。

リソース

関連ドキュメント:

- 最小特権アクセス許可を適用する
- IAM エンティティのアクセス許可境界
- Techniques for writing least privilege IAM policies
- IAM Access Analyzer は、アクセスアクティビティに基づいて IAM ポリシーを生成することにより、最小特権のアクセス許可の実装を容易にする
- Delegate permission management to developers by using IAM permissions boundaries
- 最終アクセス情報を使用した AWS のアクセス許可の調整
- IAM ポリシーのタイプと使用するタイミング
- IAM ポリシーシミュレーターを使用した IAM ポリシーのテスト
- AWS Control Tower のガードレール
- ゼロトラストアーキテクチャ: AWS の視点
- How to implement the principle of least privilege with CloudFormation StackSets
- 属性ベースのアクセスコントロール (ABAC)
- ユーザーアクティビティを表示してポリシーの範囲を狭める
- ロールへのアクセスの表示
- Use Tagging to Organize Your Environment and Drive Accountability

- AWS タグ付け戦略
- AWS リソースのタグ付け

関連動画:

- Next-generation permissions management
- Zero Trust: An AWS perspective

SEC03-BP03 緊急アクセスのプロセスを確立する

一元化された ID プロバイダーで万一問題が発生した場合に備え、ワークロードへの緊急アクセスを許可するプロセスを作成します。

緊急事態につながる可能性のあるさまざまな障害モードに対応するプロセスを設計する必要があります。例えば、通常の状況では、従業員ユーザーは一元化された ID プロバイダー (SEC02-BP04) を使用してクラウドにフェデレーションし、ワークロードを管理します。ただし、一元化された ID プロバイダーに障害が発生した場合や、クラウドのフェデレーションの設定が変更された場合、従業員ユーザーはクラウドにフェデレーションできなくなる可能性があります。緊急アクセスのプロセスでは、権限を持つ管理者がフェデレーション設定やワークロードの問題を解決するために、代替手段 (代替のフェデレーションやユーザーの直接アクセスなど) を通じてクラウドリソースにアクセスすることを許可します。緊急アクセスのプロセスは、通常のフェデレーションメカニズムが復旧するまで使用されます。

期待される成果:

- 緊急事態とみなされる障害モードを定義して文書化します。通常の状況と、ユーザーがワークロードの管理に使用するシステムを考慮してください。それぞれの依存関係でどのように障害が発生しうるか、またその障害がどのように緊急事態を引き起こすかを検討します。信頼性の柱の質問とベストプラクティスは、障害モードを特定し、障害の可能性を最小限に抑えるための、より回復力のあるシステムの構築に役立ちます。
- 障害が緊急事態であると確認する際に従うべき手順を文書化します。例えば、ID 管理者がプライマリ ID プロバイダーとスタンバイ ID プロバイダーのステータスを確認すること、両方とも使用できない場合は、ID プロバイダーに障害が発生した場合の緊急事態を宣言することを要求できます。
- 各種の緊急モードまたは障害モードに固有の緊急アクセスプロセスを定義します。具体的に定義することで、ユーザーが緊急事態の種類にかかわらず一般的なプロセスを使いすぎる状況を減らすこ

とができます。緊急アクセスのプロセスで、各プロセスを使用すべき状況、逆にそのプロセスを使用すべきでない状況、適用される可能性のある代替プロセスを説明します。

- 詳細な指示とプレイブックを含めてプロセスが十分に文書化されており、迅速かつ効率的に実行できます。緊急事態はユーザーにとってストレスの多い時間であり、ユーザーは極度の時間的プレッシャーにさらされる可能性があるため、プロセスはできるだけシンプルに設計してください。

一般的なアンチパターン:

- 緊急アクセスプロセスの文書化およびテストが不十分である。ユーザーは緊急事態への備えができておらず、緊急事態が発生しても即席のプロセスに従う。
- 緊急アクセスのプロセスで、通常のアクセスメカニズムと同じシステム (一元化された ID プロバイダーなど) を使用する。これにより、このようなシステムの障害が、通常および緊急の両方のアクセスメカニズムに影響を及ぼし、障害からの回復能力が損なわれる可能性がある。
- 緊急アクセスのプロセスが、緊急ではない状況で使用される。例えば、パイプラインを通じて変更を送信するよりも直接変更を加える方が簡単だと感じるため、ユーザーが頻繁に緊急アクセスプロセスを誤用する。
- 緊急アクセスのプロセスで、プロセスを監査するための十分なログが生成されていない、またはログが監視されておらずプロセスの誤用の可能性についてアラートされない。

このベストプラクティスを活用するメリット:

- 緊急アクセスのプロセスを十分に文書化し、十分にテストすることで、ユーザーが緊急事態に対応して解決するための時間を短縮できます。これにより、ダウンタイムが減少し、顧客に提供するサービスの可用性が高まります。
- 緊急アクセスのリクエストをそれぞれ追跡し、緊急事態以外の場合にプロセスを誤用しようとする不正な試みを検出してアラートすることができます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

このセクションでは、AWS 上にデプロイされたワークロードに関連する複数の障害モードに対する緊急アクセスプロセス作成のためのガイダンスを提供します。すべての障害モードに適用される共通のガイダンスから始め、次に障害モードのタイプに基づいた具体的なガイダンスを示します。

すべての障害モードに共通のガイダンス

障害モードに対する緊急アクセスプロセスを設計する際は、次の点を考慮してください。

- プロセスの前提条件と仮定事項 (プロセスを使用すべき場合と使用すべきでない場合) を文書化します。障害モードを詳しく説明し、他の関連システムの状態などの仮定事項を文書化しておくと役立ちます。例えば、障害モード 2 に対するプロセスでは、ID プロバイダーは使用可能だが、AWS の設定が変更されているか、有効期限が切れていることを仮定しています。
- 緊急アクセスのプロセスで必要となるリソースを事前に作成しておきます (SEC10-BP05)。例えば、IAM ユーザーとロールを持つ緊急アクセス用の AWS アカウントと、すべてのワークロードアカウントでのクロスアカウントの IAM ロールを事前に作成します。これにより、緊急事態の発生時にリソースが準備され使用可能である状態を確保することができます。リソースを事前に作成することで、緊急時に使用できない可能性のある AWS コントロールプレーン API (AWS リソースの作成と変更に使用) への依存関係がなくなります。さらに、IAM リソースを事前に作成することで、結果整合性による潜在的な遅延を考慮する必要がなくなります。
- インシデント管理計画に緊急アクセスのプロセスを含めます (SEC10-BP02)。緊急事態の追跡方法、同僚チームやリーダーシップなど組織内の他のメンバーや、該当する場合は外部の顧客やビジネスパートナーへの伝達方法を文書化します。
- 既存のサービスリクエストワークフローシステム (ある場合) で緊急アクセスリクエストプロセスを定義します。通常、このようなワークフローシステムでは、リクエストに関する情報を収集する受付フォームを作成したり、ワークフローの各段階でリクエストを追跡したり、自動および手動の承認ステップを追加したりできます。各リクエストを、インシデント管理システムで追跡される、対応する緊急イベントに関連付けます。緊急アクセス用の統一されたシステムがあると、こうしたリクエストを単一のシステムで追跡し、使用傾向を分析して、プロセスを改善できます。
- 緊急アクセスプロセスは権限を持つユーザーのみが開始できることと、必要に応じてそのユーザーの同僚または管理層の承認が必要であることを確認します。承認プロセスは、営業時間の内外で効果的に実施される必要があります。承認リクエストについて、一次承認者が不在の場合はどのように二次承認者を許可するのか、承認を受けるまで、どのように一連の管理層にリクエストをエスカレーションするのかを定義します。
- 緊急アクセスのプロセスとメカニズムに対して、堅牢なログ記録、モニタリング、アラートメカニズムを実装します。緊急アクセスに成功した場合と失敗した場合のすべての試行について、詳細な監査ログを生成します。インシデント管理システムから進行中の緊急イベントとアクティビティを関連付け、想定期間外にアクションが発生した場合、または通常のオペレーション中に緊急アクセスアカウントが使用された場合にアラートを開始します。緊急アクセスアカウントは、Break Glass 手順がバックドアと見なされる可能性があるため、緊急時にのみアクセスする必要があります。セキュリティ情報イベント管理 (SIEM) ツールまたは AWS Security Hub CSPM を統合して、緊急アクセス期間中のすべてのアクティビティをレポートおよび監査します。通常のオペレーショ

ンに戻ったら、緊急アクセス認証情報を自動的にローテーションし、関連するチームに通知します。

- 緊急アクセスプロセスを定期的にテストして、手順が明確であること、適切なレベルのアクセス権を迅速かつ効率的に付与できることを確認します。緊急アクセスのプロセスは、インシデント対応シミュレーション (SEC10-BP07) とディザスタリカバリテスト (REL13-BP03) の一部としてテストする必要があります。

障害モード 1: AWS へのフェデレーションに使用する ID プロバイダーが使用できない

「SEC02-BP04 一元化されたプロバイダーを使用する」で説明したとおり、ワークフォースユーザーをフェデレーションして AWS アカウントへのアクセス権を付与するには、一元化された ID プロバイダーを使用することを推奨します。IAM Identity Center を使用して AWS 組織内の複数の AWS アカウントにフェデレーションするか、IAM を使用して個別の AWS アカウントにフェデレーションすることができます。いずれの場合も、ワークフォースユーザーは、シングルサインオンのために AWS へのサインインエンドポイントにリダイレクトされる前に、一元化された ID プロバイダーで認証されます。

万一、一元化された ID プロバイダーが利用できなくなった場合、ワークフォースユーザーは AWS アカウントにフェデレーションすることも、ワークロードを管理することもできなくなります。こうした緊急事態には、AWS アカウントにアクセスするための緊急アクセスプロセスを少数の管理者に提供し、一元化された ID プロバイダーのオンライン復帰を待つ余裕のない重要なタスクを実行できるようにします。例えば、ID プロバイダーを 4 時間利用できない間に、顧客トラフィックの想定外の急増に対応するために、本番稼働用アカウントの Amazon EC2 Auto Scaling グループの上限を変更する必要が生じたとします。その場合、緊急管理者は、緊急アクセスプロセスに従って特定の本番稼働用 AWS アカウントへのアクセス権を取得し、必要な変更を加える必要があります。

緊急アクセスのプロセスでは、事前に作成された緊急アクセス用 AWS アカウントを使用します。このアカウントは緊急アクセスの目的でのみ使用され、緊急アクセスのプロセスに対応するための AWS リソース (IAM ロールや IAM ユーザーなど) が設定されています。通常の操作中は、誰も緊急アクセスアカウントにアクセスしてはならず、このアカウントの誤用については監視してアラートする必要があります (詳細については、前述の「共通のガイダンス」セクションを参照してください)。

緊急アクセスアカウントには、緊急アクセスを必要とする AWS アカウントでクロスアカウントロールを引き受ける権限を持つ、緊急アクセス IAM ロールがあります。これらの IAM ロールは事前に作成され、緊急アカウントの IAM ロールを信頼する信頼ポリシーで設定されます。

緊急アクセスプロセスでは、次のいずれかの方法を使用できます。

- 緊急アクセスアカウントでは、関連する強力なパスワードと MFA トークンを使用して、緊急管理者用の一連の IAM ユーザーを事前に作成します。これらの IAM ユーザーには、IAM ロールを引き受け、緊急アクセスが必要な AWS アカウントへのクロスアカウントアクセスを許可する権限があります。このようなユーザーはできるだけ少人数にし、各ユーザーを 1 人の緊急管理者に割り当てることが推奨されます。緊急時には、緊急管理者ユーザーがパスワードと MFA トークンコードを使用して緊急アクセスアカウントにサインインし、緊急アカウントで緊急アクセス IAM ロールに切り替え、最後にワークロードアカウントで緊急アクセス IAM ロールに切り替えて、緊急の変更アクションを実行します。この方法の利点は、それぞれの IAM ユーザーが 1 人の緊急管理者によって割り当てられるため、CloudTrail イベントを確認することで、どのユーザーがサインインしたかを把握できることです。欠点は、複数の IAM ユーザーと、それぞれに関連付けられた永続的なパスワードおよび MFA トークンを管理する必要があることです。
- 緊急アクセス AWS アカウントルートユーザーを使用して緊急アクセスアカウントにサインインし、緊急アクセスの IAM ロールを引き受け、ワークロードアカウントでクロスアカウントロールを引き受けることができます。ルートユーザーには強力なパスワードと複数の MFA トークンを設定することが推奨されます。また、パスワードと MFA トークンは、強力な認証と承認を実行する安全なエンタープライズ認証情報ボールトに保管することをお勧めします。パスワードと MFA トークンのリセット要因を確保する必要があります。アカウントの E メールアドレスを、クラウドセキュリティ管理者が監視するメール配布リストに設定し、アカウントの電話番号は、同様にセキュリティ管理者が監視する共有電話番号に設定します。この方法の利点は、管理するルートユーザーの認証情報が 1 セットだけであることです。欠点は、これが共有ユーザーであるため、複数の管理者がルートユーザーとしてサインインできてしまうことです。エンタープライズボールトのログイベントを監査して、どの管理者がルートユーザーのパスワードをチェックアウトしたかを特定する必要があります。

障害モード 2: AWS の ID プロバイダー設定が変更された、または有効期限が切れている

従業員ユーザーが AWS アカウントにフェデレーションできるようにするには、外部 ID プロバイダーを使用して IAM Identity Center を設定するか、IAM ID プロバイダーを作成します (SEC02-BP04)。通常、これらを設定するには、ID プロバイダーが提供する SAML メタデータ XML ドキュメントをインポートします。メタデータ XML ドキュメントには、ID プロバイダーが SAML アサーションの署名に使用するプライベートキーに対応する X.509 証明書が含まれています。

AWS 側でのこれらの設定は、管理者が誤って変更または削除する可能性があります。もう 1 つのシナリオとして、AWS にインポートされた X.509 証明書の有効期限が切れ、新しい証明書を含む新しいメタデータ XML が AWS にインポートされていない場合があります。いずれの場合も、ワークフォースユーザーの AWS へのフェデレーションが失敗し、緊急事態が発生する可能性があります。

こうした緊急事態には、フェデレーションの問題を解決するための AWS へのアクセスを ID 管理者に提供できます。例えば、ID 管理者は緊急アクセスプロセスを使用して緊急アクセス用 AWS アカウントにサインインし、アイデンティティセンター管理者アカウントのロールに切り替え、ID プロバイダーから提供された最新の SAML メタデータ XML ドキュメントをインポートして外部 ID プロバイダーの設定を更新することで、フェデレーションを再有効化することができます。フェデレーションが修正されたら、ワークフォースユーザーは引き続き通常の操作プロセスに従ってワークロードアカウントにフェデレーションします。

前述の「障害モード 1」で説明した方法に従って、緊急アクセスプロセスを作成できます。アイデンティティセンター管理者アカウントだけにアクセスし、そのアカウントでアイデンティティセンター上でのアクションを実行するための最小権限のアクセス権を、ID 管理者に付与できます。

障害モード 3: ID センターの中断

万一、IAM Identity Center または AWS リージョンが中断した場合に備えて、AWS マネジメントコンソールへの一時的なアクセスを提供するための構成を設定しておくことをお勧めします。

緊急アクセスのプロセスでは、ID プロバイダーから緊急アカウントの IAM への、直接フェデレーションを使用します。プロセスと設計上の考慮事項の詳細については、「Set up emergency access to the AWS マネジメントコンソール」を参照してください。

実装手順

すべての障害モードで共通の手順

- 緊急アクセスプロセス専用の AWS アカウントを作成します。IAM ロール、IAM ユーザー、IAM ID プロバイダー (オプション) など、アカウントで必要となる IAM リソースを事前に作成しておきます。さらに、緊急アクセスアカウントで対応する IAM ロールとの信頼関係を持つクロスアカウントの IAM ロールを、ワークロード AWS アカウントで事前に作成します。CloudFormation StackSets with AWS Organizations を使用して、組織内のメンバーアカウントでこうしたリソースを作成できます。

- メンバー AWS アカウント内のクロスアカウント IAM ロールの削除と変更を拒否する、AWS Organizations サービスコントロールポリシー (SCP) を作成します。
- 緊急アクセス AWS アカウントの CloudTrail を有効にし、ログ収集 AWS アカウントの中央の S3 バケットに証跡イベントを送信します。AWS Control Tower を使用して AWS マルチアカウント環境を設定・管理している場合は、AWS Control Tower を使用して作成した、または AWS Control Tower に登録したすべてのアカウントで CloudTrail がデフォルトで有効になっており、専用のログアーカイブ AWS アカウントの S3 バケットに送信されます。

- 緊急 IAM ロールごとのコンソールログインと API アクティビティに一致する EventBridge ルールを作成して、緊急アクセスアカウントのアクティビティを監視します。インシデント管理システムで追跡されている進行中の緊急事態以外でアクティビティが発生した場合は、セキュリティオペレーションセンターに通知を送信します。

「障害モード 1 : AWS へのフェデレーションに使用する ID プロバイダーが使用できない」、および「障害モード 2: AWS の ID プロバイダー設定が変更された、または有効期限が切れている」での追加手順

- 緊急アクセス用に選択したメカニズムに応じて、リソースを事前に作成します。
  - IAM ユーザーを使用する: 強力なパスワードおよび関連付けられた MFA デバイスを持つ IAM ユーザーを事前に作成します。
  - 緊急アカウントのルートユーザーを使用する: ルートユーザーに強力なパスワードを設定し、そのパスワードをエンタープライズ認証情報ボールトに保存します。複数の物理 MFA デバイスをルートユーザーに関連付け、緊急管理チームのメンバーがすぐにアクセスできる場所に保管します。

障害モード 3: ID センターの中断」での追加手順

- 「Set up emergency access to the AWS マネジメントコンソール」で説明しているとおり、緊急アクセス用の AWS アカウントで IAM ID プロバイダーを作成し、ID プロバイダーからの直接 SAML フェデレーションを有効にします。
- ID プロバイダーでメンバーのいない緊急オペレーショングループを作成します。
- 緊急アクセスアカウントで、緊急オペレーショングループに対応する IAM ロールを作成します。

リソース

関連する Well-Architected のベストプラクティス:

- SEC02-BP04 一元化された ID プロバイダーを利用する
- SEC03-BP02 最小特権のアクセスを付与する
- SEC10-BP02 インシデント管理計画を作成する
- SEC10-BP07 ゲームデーを実施する

関連ドキュメント:

- Set up emergency access to the AWS マネジメントコンソール
- SAML 2.0 フェデレーティッドユーザーが AWS マネジメントコンソールにアクセス可能にする
- Break glass access

関連動画:

- AWS re:Invent 2022 - Simplify your existing workforce access with IAM Identity Center
- AWS re:Inforce 2022 - AWS Identity and Access Management (IAM) deep dive

関連する例:

- AWS Break Glass Role
- AWS customer playbook framework
- AWS incident response playbook samples

SEC03-BP04 アクセス許可を継続的に削減する

チームと必要とするアクセスを決定したら、不要になったアクセス許可を削除し、最小特権のアクセス許可を達成するためのレビュープロセスを確立します。人間とマシンアクセス両方について使用しないアイデンティティとアクセス許可を継続的にモニタリングして削除します。

期待される成果: アクセス許可ポリシーは、最小特権の原則に従う必要があります。職務やロールの定義がはっきりしてくるにつれ、アクセス許可ポリシーを見直し、必要でないアクセス許可を削除する必要があります。このアプローチにより、不注意による認証情報漏洩や不正アクセスによる影響を軽減することができます。

一般的なアンチパターン:

- デフォルトでユーザーに管理者アクセス許可を付与する
- 過度に寛容でありながら、完全な管理者権限がないポリシーを作成する。
- 不要になった後もアクセス許可ポリシーを保持する。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

チームやプロジェクトが始まったばかりの場合、革新とアジリティを刺激するために、寛容な許可ポリシーが使われる可能性があります。例えば、開発またはテスト環境であれば、開発者にはさまざまな AWS サービスへのアクセスを付与できます。継続的にアクセスを評価し、アクセスを、現在のジョブを完了するために必要なサービスおよびサービスアクションのみに制限することが推奨されます。この評価は、人的およびマシン ID 両方にお薦めします。マシン ID は、システムまたはサービスアカウントと呼ばれることもありますが、AWS にアプリケーションまたはサービスへのアクセスを付与するアイデンティティです。このアクセスは、本稼働環境で特に重要です。ここでは、過剰に寛容なアクセス許可を使うと影響が大きく、顧客データを開示してしまう可能性があるためです。

AWS は、使用されていないユーザー、ロール、アクセス許可、および認証情報を特定するための方法を複数提供しています。AWS は、Amazon S3 バケットのオブジェクトなど AWS リソースへの関連付けられたアクセスキー、およびアクセスを含む、IAM ユーザーとロールのアクセス活動を分析するのにも役立ちます。AWS Identity and Access Management Access Analyzer ポリシー生成により、プリンシパルが実際にやりとりするサービスやアクションに基づいて、限定的な許可ポリシーを作成することができます。属性ベースのアクセスコントロール (ABAC) は、アクセス許可ポリシーを各ユーザーに直接アタッチするのではなく、属性を使用してユーザーにアクセス許可を付与できるため、アクセス許可の管理を簡素化するのに役立ちます。

実装手順

- AWS Identity and Access Management Access Analyzer を使用する: IAM Access Analyzer の機能は、外部エンティティと共有されている Amazon Simple Storage Service (Amazon S3) バケットや IAM ロールなど、組織とアカウントのリソースを識別するのに役立ちます。
- IAM Access Analyzer ポリシー生成を使用する: IAM Access Analyzer ポリシー生成は、IAM ユーザーまたはロールのアクセスアクティビティに基づいてきめ細かなアクセス許可ポリシーを作成するのに役立ちます。
- 本番稼働前に下位環境全体のアクセス許可をテストする: まず、重要度の低いサンドボックス環境と開発環境で、IAM Access Analyzer を使用してさまざまなジョブ機能に必要なアクセス許可をテストします。その後、本番環境に適用する前に、これらのアクセス許可をテスト環境、品質保証環境、ステージング環境で段階的に強化して検証します。サービスコントロールポリシー (SCP) は、付与される最大アクセス許可を制限することによってガードレールを適用するため、環境が低いほど、最初はアクセス許可がより緩和されます。
- IAM ユーザーおよびロールの許容時間枠と使用ポリシーを決定する: 最後にアクセスされたタイムスタンプを使用して、未使用のユーザーとロールを特定し、削除します。サービスとアクションの最終アクセス時間情報を確認して、特定のユーザーとロールのアクセス許可を特定して範囲を設

定します。例えば、最終アクセス時間情報を使用して、アプリケーションロールが必要とする特定の Amazon S3 アクションを特定し、それらのアクションのみにアクセスを制限できます。最終アクセス時間情報は、AWS マネジメントコンソールおよびプログラムで使用でき、インフラストラクチャワークフローや自動化ツールに組み込むことができます。

- AWS CloudTrail でのデータイベントの口グ記録を検討する: デフォルトでは、CloudTrail は Amazon S3 オブジェクトレベルのアクティビティ (`GetObject` や など `DeleteObject`) や Amazon DynamoDB テーブルアクティビティ (`PutItem` や `DeleteItem` など) などのデータイベントを口グに記録しません。これらのイベントの口グ記録を有効にして、特定の Amazon S3 オブジェクトまたは DynamoDB テーブルアイテムにアクティビティする必要があるユーザーとロールを決定します。

リソース

関連ドキュメント:

- 最小特権アクセス許可を適用する
- 不要な認証情報の削除
- What is AWS CloudTrail?
- IAM ポリシーを管理する
- DynamoDB でのモニタリングと口グ記録
- S3 バケットとオブジェクトの CloudTrail イベント口グ記録の有効化
- の認証情報レポートを生成します。AWS アカウント

関連動画:

- Become an IAM Policy Master in 60 Minutes or Less
- Separation of Duties, Least Privilege, Delegation, and CI/CD
- AWS re:Inforce 2022 - AWS Identity and Access Management (IAM) deep dive

SEC03-BP05 組織のアクセス許可ガードレールを定義する

アクセス許可ガードレールを使用して、プリンシパルに付与できるアクセス許可の範囲を縮小します。アクセス許可ポリシーの評価チェーンには、承認の決定を行う際のプリンシパルの有効なアクセス許可を決定するガードレールが含まれています。 ガードレールは階層型のアプローチで定義でき

ます。一部のガードレールは組織全体に広く適用し、他のガードレールはきめ細かく一時的なアクセスセッションに適用します。

期待される成果: 個別の AWS アカウントを使用して環境を明確に分離できます。 サービスコントロールポリシー (SCP) を使用して、組織全体のアクセス許可ガードレールを定義します。比較的広範なガードレールは組織のルートに近い階層に設定し、比較的厳格なガードレールは各アカウントのレベルの近くで設定します。

リソースポリシー (サポートされている場合) で、プリンシパルがリソースにアクセスするために満たす必要がある条件を定義します。リソースポリシーは、許可される一連のアクションも適宜限定します。ワークロードのアクセス許可を管理するプリンシパルにアクセス許可境界を設けたうえで、アクセス許可管理が個々のワークロード所有者に委任されています。

一般的なアンチパターン:

- AWS Organization 内に AWS アカウントメンバーを作成しているが、SCP を使用してルート認証情報で利用できる使用とアクセス許可を制限していない。
- 最小特権に基づいてアクセス許可を割り当てているが、付与できるアクセス許可一式に上限を設けるガードレールが敷かれていない。
- AWS IAM の暗黙的な拒否基盤に依存してアクセス許可を制限し、ポリシーが望ましくない明示的な許可のアクセス許可を付与しないことに頼る。
- 同じ AWS アカウント内で複数のワークロード環境を実行していて、アクセス許可境界の設定は VPC、タグ、リソースポリシーなどのメカニズム頼みである。

このベストプラクティスを活用するメリット: アクセス許可ガードレールは、アクセス許可ポリシーが付与しようとしても、望ましくないアクセス許可を付与できないという確信を持つために役立ちます。 考慮する必要があるアクセス許可の最大範囲が縮小されるため、アクセス許可の定義と管理が簡単になります。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

組織のアクセス許可ガードレールは、階層型のアプローチで定義することを推奨します。このアプローチでは、層を重ねるに従い、付与できるアクセス許可一式の上限が体系的に引き下げられます。最小特権の原則に基づいてアクセス権を付与できるため、ポリシーの設定ミスによる意図しないアクセスが起きるリスクが軽減されます。

アクセス許可ガードレールを敷くには、まず、ワークロードと環境を個別の AWS アカウントに分離します。1 つのアカウントのプリンシパルは、両方のアカウントが同じ AWS 組織または同じ組織単位 (OU) にある場合でも、明示的なアクセス許可なしに別のアカウントのリソースにアクセスすることはできません。OU を使用して、管理対象の複数のアカウントを 1 つのユニットとしてグループ化できます。

次に、組織のメンバーアカウント内のプリンシパルに付与できるアクセス許可一式の上限を引き下げます。これにはサービスコントロールポリシー (SCP) を使用できます。SCP は OU またはアカウントに適用できます。SCP では、特定の AWS リージョンへのアクセスを制限する、リソースが削除されないように防ぐ、リスクが懸念されるサービスアクションを無効にするなど、一般的なアクセス制御を適用できます。組織のルートに適用する SCP は、そのメンバーアカウントにのみ影響し、管理アカウントには影響しません。 SCP は組織内のプリンシパルにのみ適用されます。組織の外部からリソースにアクセスするプリンシパルは対象外です。

AWS Control Tower を使用している場合は、アクセス許可ガードレールとマルチアカウント環境の基盤として、コントロールとランディングゾーンを活用できます。ランディングゾーンは、事前設定済みの安全なベースライン環境と、さまざまなワークロードやアプリケーション向けの個別のアカウントを提供します。ガードレールは、サービスコントロールポリシー (SCP)、AWS Config ルール、およびその他の設定を組み合わせて、セキュリティ、オペレーション、コンプライアンスに関する必須のコントロールを適用します。しかし、Control Tower のガードレールとランディングゾーンを組織独自の SCP と併用する場合は、競合を回避し、適切なガバナンスを確保するため、AWS のドキュメントに記載されているベストプラクティスに従うことが非常に重要です。Control Tower 環境内の SCP、アカウント、組織単位 (OU) の管理に関する詳細な推奨事項については、「AWS Organizations の AWS Control Tower ガイダンス」を参照してください。

これらのガイドラインに従うことにより、Control Tower のガードレール、ランディングゾーン、カスタム SCP を効果的に活用しながら、競合を防止し、マルチアカウント AWS 環境の適切なガバナンスと管理を確保できます。

さらに、IAM リソースポリシーを使用して、管理対象のリソースに対して実行できるアクションと、動作するプリンシパルが満たす必要がある条件の範囲を絞り込むことができます。これは、プリンシパルが組織の一部である限り (PrincipalOrgId 条件キーを使用) すべてのアクションを許可するのと同じくらい広範囲にすることも、特定の IAM ロールによる特定のアクションのみを許可するのと同じくらい詳細にすることもできます。IAM ロールの信頼ポリシーの条件でも、同様のアプローチをとることができます。 リソースまたはロールの信頼ポリシーで、適用対象のロールまたはリソースと同じアカウントのプリンシパルの名前が明示的に指定されている場合、そのプリンシパルには同じアクセス許可を付与する IAM ポリシーをアタッチする必要はありません。 プリンシパルがリソー

スとは異なるアカウントにある場合は、該当するアクセス許可を付与する IAM ポリシーをプリンシパルにアタッチする必要があります。

多くの場合、ワークロードチームが担当ワークロードに必要なアクセス許可を管理することを望みます。 その場合は、そのチームが新しい IAM ロールとアクセス許可のポリシーを適宜作成する必要があります。 チームが IAM アクセス許可の境界で付与できるアクセス許可の最大範囲をキャプチャし、このドキュメントをチームが IAM ロールとアクセス許可の管理に使用できる IAM ロールに関連付けることができます。 このアプローチにより、業務の完遂に必要な柔軟性をチームに与えつつ、IAM の管理権限を持つリスクを軽減できます。

よりきめ細かいステップが、特権アクセス管理 (PAM) と一時的な昇格アクセス管理 (TEAM) を実装する方法です。 PAM の一例としては、特権が必要なアクションの実行前にプリンシパルに多要素認証の実行を要求します。 詳細については、「MFA 保護 API アクセスの設定」を参照してください。TEAM では、プリンシパルへの昇格アクセスの承認とそのアクセス権を持っていられる期間を管理するソリューションが必要です。 1 つの方法は、昇格アクセス権を持つ IAM ロールのロール信頼ポリシーにプリンシパルを一時的に追加することです。 もう 1 つの方法は、通常のオペレーションでは、セッションポリシーを使用して IAM ロールによってプリンシパルに付与されたアクセス許可の範囲を絞り込み、承認された時間枠内にこの制限を一時的に解除することです。AWS と特定のパートナーが検証したソリューションの詳細については、「Temporary elevated access」を参照してください。

実装手順

1. ワークロードと環境を個別の AWS アカウントに分離します。
2. SCP を使用して、組織のメンバーアカウント内のプリンシパルに付与できるアクセス許可一式の上限を引き下げます。
   a. SCP を定義して、組織のメンバーアカウント内のプリンシパルに付与できるアクセス許可の最大セットを削減するには、許可リストまたは拒否リストのアプローチを選択できます。許可リストによる方法では、許可するアクセスを明示的に指定し、それ以外のすべてのアクセスを暗黙的にブロックします。拒否リストによる方法は、許可しないアクセスを明示的に指定し、デフォルトで他のすべてのアクセスを許可します。どちらの方法にも利点とトレードオフがあり、どちらを選択するのが適切かどうかは、具体的な要件とリスクモデルによって異なります。詳細については、「SCP の使用戦略」を参照してください。
   b. さらに、「サービスコントロールポリシーの例」を確認して、SCP を効果的に構築する方法を理解します。

3. IAM リソースポリシーを使用して、リソースに対して許可されるアクションの範囲を限定し、その条件を指定します。 IAM ロールの信頼ポリシーで条件を指定して、ロールを引き受けた場合の制限を設けます。
4. IAM アクセス許可境界を IAM ロールに割り当てます。ワークロードチームがこのロールを使用して、各自のワークロードの IAM ロールとアクセス許可を管理できるようになります。
5. ニーズに基づいて PAM ソリューションや TEAM ソリューションを評価します。

リソース

関連ドキュメント:

- Data perimeters on AWS
- データ境界を使用してアクセス許可のガードレールを確立する
- ポリシーの評価論理

関連する例:

- Service control policy examples

関連ツール:

- AWS Solution: Temporary Elevated Access Management
- TEAM 用の検証済みセキュリティパートナーソリューション

SEC03-BP06 ライフサイクルに基づいてアクセスを管理する

プリンシパル (ユーザー、ロール、グループ) に付与されるアクセス許可を、組織内での各々の全ライフサイクルにわたり監視し、調整します。ユーザーの役割の変更に応じてグループメンバーシップを調整し、ユーザーが組織を離れた時点でアクセス権を取り消します。

期待される成果: 組織内のプリンシパルのライフサイクルを通じてアクセス許可をモニタリングおよび調整し、不要な権限のリスクを減らします。ユーザーの作成時に適切なアクセス権が付与されます。ユーザーの責任が変わった時点でアクセス権を変更し、ユーザーが業務を遂行していないときや組織を離れたときはアクセス権を取り消します。ユーザー、ロール、グループに対する変更を一元管理します。自動化を使用して、AWS 環境に変更を反映します。

一般的なアンチパターン:

- 初期段階で必要以上に過剰または広範なアクセス権限をアイデンティティに事前に付与している。
- アイデンティティの役割と責任が経時的に変化しても、アクセス権限を見直したり調整したりしない。
- 非アクティブまたは終了したアイデンティティに、アクティブなアクセス権限を与えたままにしている。これにより、不正アクセスのリスクが高まります。
- ID のライフサイクル管理に自動化が活用されていない。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

アイデンティティ (ユーザー、ロール、グループなど) に付与するアクセス権限は、それらのライフサイクル全体にわたって慎重に管理および調整してください。このライフサイクルには、初期のオンボーディングフェーズ、役割と責任の継続的な変更、そして最終的なオフボーディングまたは終了が含まれます。ライフサイクルの段階に基づいてアクセス権をプロアクティブに管理し、適切なアクセスレベルを維持します。最小特権の原則に従い、過剰または不必要なアクセス権限が付与されるリスクを軽減してください。

IAM ユーザーのライフサイクルは AWS アカウント 内で直接管理することも、ワークフォース ID プロバイダーと AWS IAM アイデンティティセンターとのフェデレーションを通じて管理することもできます。IAM ユーザーの場合は、ユーザーと関連するアクセス許可を AWS アカウント内で作成、変更、削除できます。フェデレーションユーザーについては、IAM アイデンティティセンターを使用してライフサイクルを管理できます。その場合は、System for Cross-domain Identity Management (SCIM) プロトコルを使用して、組織の ID プロバイダーからのユーザーとグループの情報を同期します。

SCIM は、さまざまなシステム間でユーザー ID のプロビジョニングとプロビジョニング解除を自動化するためのオープンスタンダードのプロトコルです。SCIM を使用して ID プロバイダーを IAM Identity Center に統合することで、ユーザーとグループの情報を自動的に同期し、組織の信頼できるアイデンティティソースにおける変更に基づいてアクセス権限が付与、変更、または取り消されているか検証できます。

組織内での従業員の役割と責任が変化したら、その従業員のアクセス権限を適宜調整してください。IAM Identity Center のアクセス許可セットを使用して、さまざまな職務または責任を定義し、それらに適切な IAM ポリシーやアクセス許可を関連付けることができます。従業員の役割が変更されたら、割り当てられているアクセス許可セットを更新して、その従業員の新しい責任を反映させることができます。最小特権の原則に従いながら、従業員に必要なアクセス権が与えられていることを確認してください。

実装手順

1. 初期アクセス権の付与、定期的なレビュー、オフボーディングの手順を含む、アクセス管理ライフサイクルのプロセスを定義し、文書化します。
2. IAM ロール、グループ、アクセス許可の境界を実装して、アクセス許可を一元管理し、最大許容アクセスレベルを適用します。
3. フェデレーティッド ID プロバイダー (Microsoft Active Directory、Okta、Ping Identity など) を、ユーザーおよびグループ情報の信頼できるソースとして IAM アイデンティティセンターを使用して統合します。
4. SCIM プロトコルを使用して、ID プロバイダーからのユーザー情報やグループ情報を IAM アイデンティティセンターの ID ストアに同期します。
5. IAM アイデンティティセンターで、組織内のさまざまな職務や責任を表すアクセス許可セットを作成します。アクセス許可セットごとに適切な IAM ポリシーとアクセス許可を定義します。
6. 定期的なアクセスレビュー、迅速なアクセス取り消し、アクセス管理ライフサイクルプロセスの継続的な改善を実施します。
7. アクセス管理のベストプラクティスについて、従業員にトレーニングを行い、周知徹底させます。

リソース

関連するベストプラクティス:

- SEC02-BP04 一元化された ID プロバイダーを利用する

関連ドキュメント:

- ID ソースを管理する
- IAM Identity Center で ID を管理する
- AWS Identity and Access Management Access Analyzer の使用
- IAM Access Analyzer ポリシーの生成

関連動画:

- AWS re:Inforce 2023 - Manage temporary elevated access with AWS IAM Identity Center
- AWS re:Invent 2022 - Simplify your existing workforce access with IAM Identity Center

- AWS re:Invent 2022 - Harness power of IAM policies & rein in permissions w/Access Analyzer

SEC03-BP07 パブリックおよびクロスアカウントアクセスの分析

パブリックおよびクロスアカウントアクセスに焦点を当てた結果を継続的にモニタリングします。パブリックアクセスとクロスアカウントアクセスを減らして、このアクセスを必要とする特定のリソースのみへのアクセスに限定します。

期待される成果: どの AWS リソースが誰と共有されているかを把握します。共有されたリソースを継続的にモニタリングおよび監査し、認証されたプリンシパルとのみ共有されていることを確認します。

一般的なアンチパターン:

- 共有されたリソースのインベントリを保持しない。
- リソースへのクロスアカウントまたはパブリックアクセスの承認のためのプロセスを遵守しない。

このベストプラクティスを活用しない場合のリスクレベル: 低

実装のガイダンス

アカウントが AWS Organizations にある場合、リソースへのアクセスを、組織全体、特定の組織単位、または個別のアカウントに付与することができます。アカウントが組織のメンバーでない場合、個別のアカウントとリソースを共有することができます。リソースベースのポリシー (例: Amazon Simple Storage Service (Amazon S3) バケットポリシー) を使用して、または別のアカウントのプリンシパルがアカウントの IAM ロールを引き受けることを許可することで、クロスアカウントアクセスを直接付与できます。リソースポリシーを使用している場合、アクセスが認証済みのプリンシパルにのみ付与されていることを確認してください。パブリックアクセス可能にする必要があるすべてのリソースを承認するプロセスを定義します。

AWS Identity and Access Management Access Analyzer は証明可能セキュリティを使用して、アカウントの外部からリソースへのすべてのアクセスパスを識別します。また、リソースポリシーの継続的な確認と、パブリックおよびクロスアカウントアクセスの結果の報告により、広範囲なアクセス権の分析を単純化します。すべてのアカウントが表示可能であることを確認するために、IAM Access Analyzer で AWS Organizations を設定することを検討します。IAM Access Analyzer では、リソースのアクセス許可をデプロイする前に検出結果をプレビューすることもできます。これにより、ポリシー変更によって、意図されたパブリックアクセスおよびクロスアカウントアクセス

のみがリソースに付与されていることを検証できます。マルチアカウントアクセス用に設計する場合、信頼ポリシーを使用して、ロールを引き受けることができるケースを制御できます。例えば、PrincipalOrgId 条件キーを使用して、AWS Organizations の外部からロールを割り当てる試みを拒否できます。

AWS Config は、誤って設定されたリソースをレポートでき、AWS Config ポリシーチェックを通じて、パブリックアクセスが設定されているリソースを検出できます。AWS Control Tower および AWS Security Hub CSPM などのサービスでは、AWS Organizations 全体でチェックとガードレールのデプロイが簡素化され、パブリックに公開されているリソースを特定、修復できます。例えば、AWS Control Tower には、Amazon EBS スナップショットが AWS アカウントによって復元可能かどうかを検出できるマネージドガードレールがあります。

実装手順

- AWS Config に AWS Organizations を使用することを検討する: AWS Config では、AWS Organizations 内の複数のアカウントから委任された管理者アカウントに検出結果を集約できます。これにより、包括的なビューが提供され、アカウント間で AWS Config ルール をデプロイして、パブリックにアクセス可能なリソースを検出できます。
- AWS Identity and Access Management Access Analyzer を設定する: IAM Access Analyzer は、外部エンティティと共有されている Amazon S3 バケットや IAM ロールなど、組織とアカウントのリソースを識別するのに役立ちます。
- AWS Config で自動修復を使用して、Amazon S3 バケットのパブリックアクセス設定の変更に対応する: Amazon S3 バケットのパブリックアクセスブロック設定を自動的にオンにできます。
- モニタリングとアラートを実装して、Amazon S3 バケットがパブリックになったかどうかを確認する: Amazon S3 パブリックアクセスブロックがオフになって、Amazon S3 バケットがパブリックになったかどうかを特定するモニタリングとアラートを設定する必要があります。さらに、AWS Organizations を使用している場合は、Amazon S3 パブリックアクセスポリシーへの変更を防ぐサービスコントロールポリシーを作成できます。AWS Trusted Advisor は、オープンアクセス許可を持つ Amazon S3 バケットをチェックします。誰にでもアクセスを付与、アップロード、削除するバケット権限は、バケットのアイテムを誰でも追加、変更、または削除できるようにすることで、セキュリティ関連の問題の原因となることがあります。Trusted Advisor のチェックでは、バケットの明示的なアクセス許可を検証します。また、バケットに関連付けられたポリシーで、バケットのアクセス許可を上書きする可能性があるものについても検証します。また、AWS Config を使って、Amazon S3 バケットにパブリックアクセスがないかモニタリングできます。詳細については、「パブリックアクセスを許可する Amazon S3 バケットを AWS Config でモニタリングおよび応答する方法」を参照してください。

Amazon S3 バケットのアクセスコントロールを確認する場合は、バケット内に保存されているデータの性質を考慮することが重要です。Amazon Macie は、個人を特定できる情報 (PII)、保護対象医療情報 (PHI)、プライベートキーや AWS アクセスキーなどの認証情報などの機密データを検出し、保護するのに役立つように設計されています。

リソース

関連ドキュメント:

- AWS Identity and Access Management Access Analyzer の使用
- AWS Control Tower controls library
- AWS Foundational Security Best Practices standard
- AWS Config マネージドルール
- AWS Trusted Advisor check reference
- Monitoring AWS Trusted Advisor check results with Amazon EventBridge
- Managing AWS Config Rules Across All Accounts in Your Organization
- AWS Config および AWS Organizations
- Amazon EC2 で使用するために AMI を公開する

関連動画:

- Best Practices for securing your multi-account environment
- Dive Deep into IAM Access Analyzer

SEC03-BP08 組織内でリソースを安全に共有する

ワークロードの数が増えるにつれて、それらのワークロードのリソースへのアクセスを共有したり、複数のアカウントでリソースを複数回プロビジョニングしたりする必要が生じます。開発環境、テスト環境、本番環境などの環境を区分けするための構造があるかもしれません。ただし、分離構造があっても、安全に共有する能力は制限できません。重複するコンポーネントを共有することにより、運用諸経費を削減し、同一リソースを複数回作成する間に見逃したものを推測しなくても、一貫したエクスペリエンスを実現できます。

期待される成果: 安全な方法を使用して組織内でリソースを共有し、データ損失防止イニシアチブを支援することで、意図しないアクセスを最小限に抑えます。個々のコンポーネントを管理するのと比較して、運用諸経費を削減し、同じコンポーネントを何度も手動で作成することによるエラーを減ら

し、ワークロードのスケーラビリティを向上させることができます。削減できた時間を活用して、マルチポイント障害シナリオを解決し、自信を持ってコンポーネントが不要になるときを判断できるようになります。外部共有リソースの分析に関する規範ガイダンスについては、「SEC03-BP07 パブリックおよびクロスアカウントアクセスの分析」を参照してください。

一般的なアンチパターン:

- 継続的にモニタリングして、予定外の外部共有が生じたときに自動的にアラートを発動するプロセスがない。
- 共有すべき/すべきでない内容に関する基準がない。
- 必要な時点で明示的に共有するのではなく、広く開かれたポリシーをデフォルトとしている。
- 必要に応じて重複する基本的リソースを手動で作成する。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

アクセスコントロールとパターンを構築し、信頼できるエンティティとのみ共有リソースの消費を安全に管理します。共有リソースをモニタリングして、継続的に共有リソースアクセスをレビューし、不適切なまたは予想外の共有があればアラートを発動します。「パブリックおよびクロスアカウントアクセスの分析」を確認し、ガバナンスを確立して、外部アクセスを必要なリソースのみに減らします。また、継続的かつ自動的にアラートをモニタリングするプロセスを確立します。

AWS Organizations 内のクロスアカウント共有は、AWS Security Hub CSPM、Amazon GuardDuty、AWS Backup など、多数の AWS サービスでサポートされています。これらのサービスを使用すると、中央アカウントでデータを共有し、中央アカウントからアクセス可能、あるいは中央アカウントからリソースとデータを管理できます。例えば、AWS Security Hub CSPM は個別アカウントから中央アカウントに検出結果を送信するため、すべての検出結果を確認することができます。AWS Backup は、リソースのバックアップを取り、アカウント全体で共有します。AWS Resource Access Manager (AWS RAM) を使用することで、VPC サブネットや Transit Gateway アタッチメント、AWS Network Firewall、Amazon SageMaker AI Pipelines など、他の一般的なリソースを共有することができます。

アカウントが組織内のリソースのみを共有するように制限するには、サービスコントロールポリシー (SCP) を使用して、外部プリンシパルへのアクセスを防止します。リソースを共有するときは、アイデンティティベースのコントロールとネットワークコントロールを組み合わせて組織のデータ境界を作成し、意図しないアクセスから保護します。データ境界とは、信頼できるアイデンティティのみが、期待されるネットワークから信頼できるリソースにアクセスするよう徹底するのに役立つ予防的

な一連のガードレールです。これらのコントロールは、どのリソースが共有可能かについて適切な制限を設け、共有や公開が許可されるべきでないリソースについてはそれを禁止する必要があります。例えば、データ境界の一部として、VPC エンドポイントポリシーと `AWS:PrincipalOrgId` 条件を使用して、Amazon S3 バケットにアクセスする ID が組織に属していることを確認できます。SCP はサービスにリンクされたロールまたは AWS サービスプリンシパルには適用されないことにご注意ください。

Amazon S3 を使用する場合は、Amazon S3 バケット ACL をオフにし、IAM ポリシーを使用してアクセスコントロールを定義します。Amazon CloudFront から Amazon S3 オリジンにアクセスされることを制限するには、オリジンアクセスアイデンティティ (OAI) からオリジンアクセスコントロール (OAC) に移行します。OAC では AWS Key Management Service によるサーバー側暗号化などの追加機能をサポートします。

場合によっては、組織外のリソースを共有したり、リソースにサードパーティーのアクセスを付与したりするかもしれません。外部でリソースを共有するアクセス許可を管理するための規範ガイダンスについては、「アクセス許可の管理」を参照してください。

実装手順

1. AWS Organizations を使用する: AWS Organizations は、ユーザーが作成する組織に、複数の AWS アカウント を統合し、一元管理できるアカウント管理サービスです。アカウントを組織単位 (OU) にグループ化し、OU ごとに異なるポリシーをアタッチすることにより、予算、セキュリティ、コンプライアンスのニーズに対応できます。また、AWS 人工知能 (AI) と機械学習 (ML) サービスがどのようにデータを収集して保管するかをコントロールし、Organizations と統合された AWS サービスのマルチアカウント管理を使用できます。
2. AWS Organizations と AWS サービスの統合: 組織のメンバーアカウントで自動的にタスクを実行するために AWS サービスを使用すると、AWS Organizations はそのサービス用のサービスにリンクされた IAM ロール (SLR) を各メンバーアカウントに作成します。AWS マネジメントコンソール、AWS API、または AWS CLI を使用して、信頼できるアクセスを管理する必要があります。信頼されたアクセスを有効にするための規範ガイダンスについては、「AWS Organizations を AWS の他のサービスと併用する」および「Organizations と併用できる AWS サービス」を参照してください。
3. データ境界を確立する: データ境界は、信頼と所有権の明確な境界を提供します。AWS では、通常、AWS リソースにアクセスするオンプレミスネットワークまたはシステムと共に、AWS Organizations によって管理される AWS Organization として表されます。このデータ境界の目標は、アイデンティティが信頼され、リソースが信頼され、さらにネットワークが予想されている場合に、そのアクセスが許可されていることを検証することです。ただし、データ境界を確立することは、万能なアプローチではありません。AWS ホワイトペーパーの「境界の構築」に記載さ

れているコントロール目標を、特定のセキュリティリスクモデルと要件に基づいて評価し、採用します。リスクに対する企業固有の姿勢を慎重に検討し、セキュリティニーズに沿った境界コントロールを実装する必要があります。

4. AWS サービスでリソース共有を使用し、必要に応じて制限する: 多くの AWS サービスでは、リソースを別のアカウントと共有できます。また、Amazon マシンイメージ (AMI) および AWS Resource Access Manager (AWS RAM) など別のアカウントのリソースをターゲットにできます。`ModifyImageAttribute` API を制限して、AMI を共有する信頼されたアカウントを指定します。AWS RAM を使用して組織への共有のみを制限する場合は、信頼できない ID からのアクセスを防ぐために `ram:RequestedAllowsExternalPrincipals` 条件を指定します。規範ガイダンスと考慮事項については、「リソース共有と外部ターゲット」を参照してください。
5. AWS RAM を使用して、アカウントまたは他の AWS アカウント と安全に共有する: AWS RAM は、作成したリソースをアカウント内のロールやユーザー、他の AWS アカウント とともに安全に共有できます。マルチアカウント環境の場合、AWS RAM ではリソースを作成したら、それを他のアカウントと共有できます。このアプローチにより、運用諸経費を削減し、Amazon CloudWatch および AWS CloudTrail との統合を通じて、一貫性、可視性、監査可能性を提供することができます。これは、クロスアカウントアクセスを使用している場合は享受できません。

   リソースベースのポリシーを使用して過去に共有したリソースが存在する場合、`PromoteResourceShareCreatedFromPolicy` API または同等機能を使用して、リソース共有を完全な AWS RAM リソース共有に昇格できます。

   場合によっては、リソースを共有するための追加ステップが必要かもしれません。例えば、暗号化されたスナップショットを共有するには、AWS KMS キーを共有する必要があります。

リソース

関連するベストプラクティス:

- SEC03-BP07 パブリックおよびクロスアカウントアクセスの分析
- SEC03-BP09 サードパーティーとリソースを安全に共有する
- SEC05-BP01 ネットワークレイヤーを作成する

関連ドキュメント:

- 例 4 - バケット所有者が所有権のないオブジェクトへのクロスアカウントアクセス許可を付与する
- How to use Trust Policies with IAM

- Building Data Perimeter on AWS
- AWS リソースへのアクセス権を第三者に付与するときに外部 ID を使用する方法
- AWS services you can use with AWS Organizations
- Establishing a data perimeter on AWS: Allow only trusted identities to access company data

関連動画:

- Granular Access with AWS Resource Access Manager
- Securing your data perimeter with VPC endpoints
- Establishing a data perimeter on AWS

関連ツール:

- Data Perimeter Policy Examples

SEC03-BP09 サードパーティーとリソースを安全に共有する

クラウド環境のセキュリティは、組織内にとどまりません。組織が、データの一部を管理するのにサードパーティーに依存することもあります。サードパーティー管理システムの権限管理は、一時的な認証情報を使用する最小特権の原則を用いたジャストインタイムアクセスの実践に従う必要があります。サードパーティーと密に連携することにより、意図しないアクセスの影響が及ぶ範囲とリスクをともに縮小することができます。

期待される成果: アクセスキーやシークレットキーなどの長期 AWS Identity and Access Management (IAM) 認証情報は、悪用された場合にセキュリティリスクをもたらすため、使用を避けます。代わりに、IAM ロールと一時的な認証情報を使用してセキュリティ体制を強化し、長期的な認証情報を管理する運用上のオーバーヘッドを最小限に抑えます。サードパーティーにアクセス権を付与する場合は、IAM 信頼ポリシーの外部 ID として共通な一意の識別子 (UUID) を使用し、IAM ポリシーをコントロール下にあるロールにアタッチして、最小特権アクセスを確保します。外部で共有されているリソースの分析に関する規範ガイダンスについては、「SEC03-BP07 パブリックおよびクロスアカウントアクセスの分析」を参照してください。

一般的なアンチパターン:

- 条件なしでデフォルトの IAM 信頼ポリシーを使用する。
- 長期的 IAM 認証情報とアクセスキーを使う。

- 外部 ID を再使用する。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

AWS Organizations 外のリソースを共有したり、アカウントにサードパーティーのアクセスを付与したりする場合があります。例えば、サードパーティーが提供する監視ソリューションが、貴社のアカウント内のリソースにアクセスする必要があるかもしれません。そのような場合、サードパーティーにとって必要な権限のみを含む IAM クロスアカウントロールを作成します。さらに、外部の ID 条件を使用して信頼ポリシーを定義します。外部 ID を使用すると、自分またはサードパーティーが各顧客、サードパーティー、またはテナンシーに対して一意の ID を生成できます。一意の ID を作成後は、自分以外の人物によってコントロールできなくなります。サードパーティーは、外部 ID を安全に、監査可能かつ再現可能な方法で顧客に関連付けるプロセスを実装する必要があります。

また、IAM Roles Anywhere を使用すると、AWS API を使用している AWS の外部のアプリケーションの IAM ロールを管理できます。

サードパーティーが貴社の環境にアクセスする必要がなくなった場合は、ロールを削除します。サードパーティーに長期的な認証情報を提供することは避けてください。他の AWS アカウント とワークロードを共有できる AWS Well-Architected Tool や、所有する AWS リソースを他のアカウントと安全に共有できる AWS Resource Access Manager など、共有をサポートする他の AWS サービスについて把握しておきます。

実装手順

1. クロスアカウントロールを使用して外部アカウントにアクセスできるようにします。クロスアカウントロールを使用すると、顧客にサービスを提供する目的で外部アカウントやサードパーティーが保存している機密情報の量を、減らすことができます。クロスアカウントロールがあると、アクセスを管理および監査する能力を維持しながら、AWS パートナーまたは組織内の他のアカウントなど、アカウントの AWS リソースへのアクセスをサードパーティーに安全に付与できます。サードパーティーが、ハイブリッドインフラストラクチャからサービスを提供したり、オフサイトロケーションにデータをプルしたりする場合があります。IAM Roles Anywhere を使用すると、サードパーティーのワークロードは AWS ワークロードと安全に通信でき、長期的な認証情報の必要性も削減できます。

   外部アカウントアクセスを提供するために、ユーザーと関連付けられた長期的認証情報、またはアクセスキーを使用しないでください。かわりに、クロスアカウントロールを使ってクロスアカウントアクセスを提供します。

2. デューデリジェンスを実施して、サードパーティーの SaaS プロバイダーの安全なアクセスを確保します。サードパーティーの SaaS プロバイダーとリソースを共有する場合は、そのプロバイダーが AWS リソースにアクセスする際に安全で責任あるアプローチを取っていることを確認するために、徹底的なデューデリジェンスを実施します。責任共有モデルを評価し、どのようなセキュリティ対策が提供され、何が自社の責任に該当するかを理解します。SaaS プロバイダーについて、外部 ID の使用や最小特権アクセス原則など、リソースにアクセスするために安全で監査可能なプロセスがあることを確認します。外部 ID を使用すると、混乱した代理問題 に対処できます。

   サードパーティーの SaaS プロバイダーへのアクセスを許可する場合は、安全なアクセスと最小特権の原則の遵守を確保するため、セキュリティ対策を実施します。これには、外部 ID、共通の一意の識別子 (UUID)、およびアクセスを必要なもののみに厳格に制限する IAM 信頼ポリシーの使用が含まれます。SaaS プロバイダーと密接に連携して、安全なアクセスメカニズムを確立し、AWS リソースへのアクセスを定期的に見直し、監査を実施してセキュリティ要件へのコンプライアンスを確保します。

3. 顧客が提供する長期的認証情報を廃止します。長期的認証情報の使用を廃止して、クロスアカウントロールまたは IAM Roles Anywhere を使用します。長期的認証情報を使用する必要がある場合、ロールベースのアクセスに移行する計画を立ててください。キー管理の詳細については、「ID 管理」を参照してください。また、AWS アカウントチームおよびサードパーティーと連携して、リスク軽減ランブックを確立します。セキュリティインシデントの潜在的影響への対応とその軽減に関する規範的ガイダンスについては、「インシデント対応」を参照してください。

4. セットアップに規範的ガイダンスがある、または規範的ガイダンスが自動化されていることを確認します。外部 ID はシークレットとして取り扱われませんが、外部 ID は電話番号、氏名、アカウント ID など推測しやすい値であってはなりません。外部 ID を読み取り専用にすることで、設定のなりすましを目的として外部 ID が変更されないようにします。

   ご自身またはサードパーティーが外部 ID を生成できます。ID 生成に責任がある担当者を決定するプロセスを定義します。外部 ID を作成するエンティティにかかわらず、サードパーティーは顧客間で一貫した一意性とフォーマットを適用します。

   アカウントのクロスアカウントアクセス向けに作成されたポリシーは、最小特権の原則に従う必要があります。サードパーティーは、ロールポリシードキュメント、または AWS CloudFormation テンプレートまたは同等のものを使用する自動化されたセットアップメカニズムを提供する必要があります。これにより、手動ポリシー作成に関連してエラーが発生する可能性が減り、監査可能証跡を提供します。AWS CloudFormation テンプレートを使用してクロスアカウントロールを作成する方法の詳細については、「Cross-Account Roles」を参照してください。

サードパーティーは、自動化され、監査可能なセットアップメカニズムを提供する必要があります。ただし、必要なアクセスを概説したロールポリシードキュメントを使用することにより、ロールのセットアップを自動化する必要があります。AWS CloudFormation テンプレートまたは同等のものを使用して、監査業務の一環として、ドリフト検出で変化をモニタリングする必要があります。

5. 変更点を説明します。アカウント構成、サードパーティーのニーズ、または提供されるサービスオファリングは変わる可能性があります。変更と障害を予想し、それに応じて適切な人材、プロセス、テクノロジーを計画する必要があります。定期的に提供するアクセスのレベルを監査し、予想外の変更があった場合にアラートを出す検出方法を実装します。外部 ID のロールとデータストアをモニタリングおよび監査します。予想外の変更やアクセスパターンの結果、サードパーティーのアクセスを一時的または恒久的に取り消す準備をしておく必要があります。また、実行にかかる時間、関与する人材、コスト、および他のリソースの影響など、取り消し操作の影響を測定します。

   検知方法に関する規範的なガイダンスについては、「検知」内のベストプラクティスを参照してください。

リソース

関連するベストプラクティス:

- SEC02-BP02 一時的な認証情報を使用する
- SEC03-BP05 組織のアクセス許可ガードレールを定義する
- SEC03-BP06 ライフサイクルに基づいてアクセスを管理する
- SEC03-BP07 パブリックおよびクロスアカウントアクセスの分析
- SEC04 検知

関連ドキュメント:

- 例 4 - バケット所有者が所有権のないオブジェクトへのクロスアカウントアクセス許可を付与する
- How to use trust policies with IAM roles
- IAM チュートリアル: IAM ロールを使用した AWS アカウント 間でのアクセスの委任
- IAM を使用して別の AWS アカウント のリソースにアクセスするにはどうすればよいですか?
- IAM でのセキュリティのベストプラクティス

- クロスアカウントポリシーの評価論理
- AWS リソースへのアクセス権を第三者に付与するときに外部 ID を使用する方法
- Collecting Information from AWS CloudFormation Resources Created in External Accounts with Custom Resources
- Securely Using External ID for Accessing AWS Accounts Owned by Others
- Extend IAM roles to workloads outside of IAM with IAM Roles Anywhere

関連動画:

- How do I allow users or roles in a separate AWS アカウント access to my AWS アカウント?
- AWS re:Invent 2018: Become an IAM Policy Master in 60 Minutes or Less
- AWS Knowledge Center Live: IAM Best Practices and Design Decisions

関連する例:

- Amazon DynamoDB へのクロスアカウントアクセスを設定する
- AWS STS Network Query Tool

検出

検知は、予期しない、望ましくない設定の変更の検知、および想定外の動作の検知という 2 つの部分に分かれています。1 番目は、アプリケーション配信ライフサイクルのあらゆる段階で発生する可能性があります。Infrastructure as Code (例えば CloudFormation テンプレート) を使用し、CI/CD パイプラインやソース管理でチェックを実施することにより、ワークロードをデプロイする前に不要な設定をチェックできます。次に、ワークロードを非本番環境と本番環境にデプロイする際、ネイティブの AWS、オープンソース、または AWS パートナーツールを使って設定をチェックできます。これらのチェックは、セキュリティプリンシパルまたはベストプラクティスに合致しない設定や、テストされた設定とデプロイされた設定の間で行われた変更に対して実行できます。実行中のアプリケーションの場合、既知のデプロイ以外や自動化されたスケーリングイベントなど、設定が予期しないやり方で変更されたかどうかをチェックできます。

2 番目の予期しない動作の検知については、ツールを使うか、または特定タイプの API コール増加アラートを送信することができます。Amazon GuardDuty を使用すると、予期しない、さらに潜在的に不正または悪意のあるアクティビティが AWS アカウントで発生した場合、アラートを受け取ることができます。また、ワークロードでの使用が予想されていない変異型 API コールや、セキュリティ体制を変更する API コールを明示的にモニタリングする必要があります。

検出により、潜在的なセキュリティ設定の誤り、脅威、予期しない動作を特定できます。検出はセキュリティライフサイクルの最重要部分であり、品質管理プロセス、法的義務またはコンプライアンス義務、脅威の特定とその対応をサポートします。検出メカニズムにはさまざまなタイプがあります。例えば、ワークロードのログは、使用された脆弱性を分析できます。ワークロードに関連する検出メカニズムを定期的に見直し、内部および外部のポリシーと要件を満たしていることを確認する必要があります。自動化されたアラートと通知は、チームやツールが調査できるように、定義された条件に基づいて行う必要があります。これらのメカニズムは、組織が異常なアクティビティの範囲を特定し把握するのに役立つ重要な対応機能です。

AWS には、検出メカニズムに対処する際に使用できる多くのアプローチがあります。以下の各セクションでは、こうしたアプローチの使用方法を説明します。

ベストプラクティス

- SEC04-BP01 サービスとアプリケーションのログ記録を設定する
- SEC04-BP02 標準化した場所にログ、検出結果、メトリクスを取り込む
- SEC04-BP03 セキュリティアラートを相関付けて充実させる
- SEC04-BP04 非準拠リソースの修復を開始する

SEC04-BP01 サービスとアプリケーションのログ記録を設定する

サービスとアプリケーションのセキュリティイベントログを保持します。これは、監査、調査、運用のユースケースにおけるセキュリティの基本原則であり、ガバナンス、リスク、コンプライアンス (GRC) の標準、ポリシー、手順によって推進される共通のセキュリティ要件です。

期待される成果: 組織は、セキュリティインシデント対応など内部のプロセスまたは義務を遂行する必要があるときは、AWS サービスやアプリケーションからのセキュリティイベントログを確実にかつ一貫性をもって、タイムリーに取得できるようにしておく必要があります。運用側の成果を改善するためにログの一元化を検討してください。

一般的なアンチパターン:

- ログが永久に保存される、またはすぐに削除される。
- 誰でもログにアクセスできる。
- ログガバナンスと使用について、手動プロセスのみに依存する。
- 必要な場合に備えて、あらゆるタイプのログを保存する。
- 必要な場合にのみログ整合性をチェックする。

このベストプラクティスを活用するメリット: セキュリティインシデントの根本原因分析 (RCA) のメカニズムを導入し、ガバナンス、リスク、コンプライアンスの義務における証拠の源とします。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

セキュリティ調査または要件に基づいた他のユースケース中、インシデントの全容とタイムラインを記録して理解するために関連ログをレビューできる必要があります。ログはまた、関心のある特定のアクションが発生したことを示すアラート生成にも必須です。クエリと取得のメカニズムを選択、有効化、保存、設定してアラートを発するのに不可欠です。

実装手順

- ログのソースを選択して使用します。セキュリティ調査の前に、関連するログを取得し、過去にさかのぼって AWS アカウントでアクティビティを再構築する必要があります。ワークロードに関連するログソースを選択します。

ログソース選択条件は、ビジネスで必要なユースケースに基づいたものである必要があります。各 AWS アカウントに AWS CloudTrail または AWS Organizations 証跡を使って証跡を作成し、それらの Amazon S3 バケットを設定します。

AWS CloudTrail は、AWS のサービスアクティビティをキャプチャする AWS アカウントに対して API コールをトラッキングするログサービスです。これはデフォルトで有効になっており、管理イベントは 90 日間保持され、AWS マネジメントコンソール、AWS CLI、AWS のいずれかを使用して CloudTrail イベント履歴から検索することが可能です。データイベントをより長く保持し、確認できるようにするには、CloudTrail 証跡を作成して、これを Amazon S3 バケットと Amazon CloudWatch ロググループ (任意) に関連付けます。あるいは、CloudTrail Lake を作成する方法もあります。この方法では、CloudTrail ログを最長 7 年間保持でき、SQL ベースのクエリ機能を利用できます。

AWS では、VPC を使用しているお客様には、VPC フローログと Amazon Route 53 Resolver のクエリログをそれぞれ使用してネットワークトラフィックと DNS ログを有効にし、それらを Amazon S3 バケットまたは CloudWatch ロググループにストリーミングすることを推奨しています。VPC、サブネット、またはネットワークインターフェイス向けに VPC フローログを作成できます。VPC フローログについては、コストを削減するためにどこでどのようにフローログを使用するかを選択できます。

AWS CloudTrail ログ、VPC フローログ、Route 53 Resolver のクエリログは、AWS でのセキュリティ調査をサポートする基本的なログ記録ソースです。Amazon Security Lake を使用して、このログデータを収集、正規化し、Apache Parquet フォーマットと Open Cybersecurity Schema Framework (OCSF) で保存することもできます。これらはクエリに使用できます。Security Lake は、他の AWS ログ、およびサードパーティーソースからのログもサポートします。

AWS のサービスは、Elastic Load Balancing ログ、AWS WAF ログ、AWS Config レコーダーログ、Amazon GuardDuty の検出結果、Amazon Elastic Kubernetes Service (Amazon EKS) 監査ログ、Amazon EC2 インスタンスのオペレーティングシステムとアプリケーションログなど、基本のログソースではキャプチャされないログを生成できます。ログ記録とモニタリングオプションの一覧については、「AWS セキュリティインシデント対応ガイド」の「付録 A: クラウド機能の定義 – ログ記録とイベント」を参照してください。

- 各 AWS サービスとアプリケーションの調査ログ機能: 各 AWS サービスとアプリケーションにはログストレージのさまざまなオプションが用意されており、それぞれ独自の保持機能とライフサイクル機能を備えています。最も一般的な 2 つのログストレージサービスは、Amazon Simple Storage Service (Amazon S3) と Amazon CloudWatch です。保持期間が長い場合、費用対効果と柔軟なライフサイクル機能のために Amazon S3 を使用することが推奨されます。ログ記録の主要

な方法が Amazon CloudWatch Logs である場合、オプションとして、アクセス頻度の低いログを Amazon S3 にアーカイブすることを検討する必要があります。

- ログストレージを選ぶ: どのログストレージを選ぶかは、使用しているクエリツール、保持機能、使いやすさ、コストなどが関わってきます。ログストレージの一般的な選択肢は、Amazon S3 バケットまたは CloudWatch Log ロググループです。

  Amazon S3 バケットは、ライフサイクルポリシーがオプションで備わっている、費用対効果に優れ、耐久性の高いストレージを提供します。Amazon S3 バケットに保存されているログは、Amazon Athena などのサービスを使ってクエリすることができます。

  CloudWatch ロググループは、CloudWatch Logs Insights により、耐久性の高いストレージとビルトインクエリ施設を提供します。

- 適切なログ保持を特定する: Amazon S3 バケットまたは CloudWatch ロググループを使ってログを保存するときは、各ログソースに対して適切なライフサイクルを選び、ストレージと取得コストを最適化する必要があります。顧客のログは通常 3 か月～1 年間で、すぐにクエリでき、最長 7 年間保持されます。可用性と保持の選択は、セキュリティ要件と、法令、規制、およびビジネス上の義務の組み合わせに合わせるべきです。
- 適切な保持とライフサイクルポリシーを使って各 AWS サービスとアプリケーションのログを使用する: 組織の各 AWS サービスまたはアプリケーションには、特定のログ記録の設定ガイダンスを確認します。
  - AWS CloudTrail 証跡を設定する
  - VPC フローログを設定する
  - Amazon GuardDuty Finding Export を設定する
  - AWS Config 記録を設定する
  - AWS WAF Web ACL トラフィックを設定する
  - AWS Network Firewall ネットワークトラフィックログを設定する
  - Elastic Load Balancing アクセスログを設定する
  - Amazon Route 53 リゾルバーのクエリログを設定する
  - Amazon RDS ログを設定する
  - Amazon EKS コントロールプレーンログを設定する
  - Amazon EC2 インスタンスとオンプレミスサーバーに Amazon CloudWatch エージェントを設定する
- ログのクエリメカニズムを選択して実装する: ログのクエリについては、CloudWatch Logs Insights を CloudWatch ロググループに保存されたデータに、Amazon Athena と Amazon

OpenSearch Service を Amazon S3 に保存されたデータに使用できます。また、セキュリティ情報とイベント管理 (SIEM) サービスなど、サードパーティーのクエリツールを使用することもできます。

ログクエリツールを選択するためのプロセスは、セキュリティオペレーションの人材、プロセス、およびテクノロジー側面を考慮する必要があります。オペレーション、ビジネス、セキュリティの要件を満たし、長期的にアクセスとメンテナンスが可能なツールを選択します。ログクエリツールは、スキャンするログの数がツールの制限内に収まっている場合、動作が最適であることに注意してください。コストや技術的な制約から、複数のクエリツールを所有することも珍しくありません。

例えば、過去 90 日間のデータにはサードパーティーのセキュリティ情報とイベント管理 (SIEM) ツールを使用しながらも、SIEM のログインジェストコストが原因で 90 日以前のデータをクエリする際は Athena を使用するとった場合です。どのような実装であっても、必要なツールの数を最小限に抑えることで、特にセキュリティイベントの調査時に、運用効率が最大となるアプローチであることを確認してください。

- アラートにログを使用する: AWS は複数のセキュリティサービスでアラート機能を提供しています。
  - AWS Config は、AWS リソース構成のモニタリングと記録が行われ、目標の構成に対する評価と修復が自動化できます。
  - Amazon GuardDuty は、悪意のあるアクティビティや不正な動作を継続的にモニタリングして AWS アカウントとワークロードを保護する脅威検知サービスです。GuardDuty は、AWS CloudTrail 管理およびデータイベント、DNS ログ、VPC フローログ、Amazon EKS 監査ログなどのソースから情報を取得して、集計、分析します。GuardDuty は、CloudTrail、VPC フローログ、DNS クエリログ、Amazon EKS から、独立したデータストリームを直接プルします。Amazon S3 バケットポリシーを管理したり、ログの収集と保存方法を変更したりする必要はありません。独自の調査やコンプライアンス目的で、これらのログを保持することは引き続き推奨されています。
  - AWS Security Hub CSPM では、複数の AWS のサービスや任意のサードパーティー製品からのセキュリティアラートまたは検出結果の集約、整理、優先順位付けが一元的に行われ、セキュリティアラートとコンプライアンスステータスを包括的に把握できます。

また、これらのサービスの対象外となるセキュリティアラートや、自分の環境に関連する特定なアラートについては、カスタムアラート生成エンジンを使用することもできます。これらのアラートや検知の設定に関する詳細は、「AWS セキュリティインシデント対応ガイド」の「検知」を参照してください。

リソース

関連するベストプラクティス:

- SEC04-BP02 標準化した場所にログ、検出結果、メトリクスを取り込む
- SEC07-BP04 スケーラブルなデータのライフサイクル管理を定義する
- SEC10-BP06 ツールを事前デプロイする

関連ドキュメント:

- AWS Security Incident Response Guide
- Amazon Security Lake の開始方法
- Getting started: Amazon CloudWatch Logs

関連動画:

- AWS re:Invent 2022 - Introducing Amazon Security Lake

関連する例:

- Assisted Log Enabler for AWS
- AWS Security Hub CSPM Findings Historical Export

SEC04-BP02 標準化した場所にログ、検出結果、メトリクスを取り込む

セキュリティチームは、ログと検出結果に基づいて、不正なアクティビティや意図しない変更を示唆する可能性があるイベントを分析します。この分析の効率を高めるため、標準化した場所にセキュリティログや検出結果を集めてください。 重要なデータポイントを相関のために使えるようになり、ツールの統合を簡素化できます。

期待される成果: ログデータ、検出結果、メトリクスの収集、分析、視覚化に、標準化された方法を使用できます。セキュリティチームは、さまざまなシステムから集めたセキュリティデータを効率的に相関付けて分析し、視覚化して、潜在的なセキュリティイベントを発見し、異常を検知できます。セキュリティ情報とイベント管理 (SIEM) システムやその他のメカニズムを統合してログデータを照会および分析し、セキュリティイベントに対し適時の対応、追跡、エスカレーションを行います。

一般的なアンチパターン:

- チームがそれぞれ独自にログやメトリクスのコレクションを所有および管理していて、それが組織のログ記録の戦略と矛盾している。
- チームが収集したデータの表示と変更を制限するための十分なアクセス制御を実施していない。
- チームがセキュリティログ、検出結果、メトリクスをデータ分類ポリシーに則って管理していない。
- チームがデータ収集の設定時に、データ主権とローカリゼーションの要件を無視している。

このベストプラクティスを活用するメリット: 標準化されたログ記録ソリューションを使用してログデータやイベントを収集しクエリすることで、ログに含まれる情報からより良いインサイトを引き出すことができます。収集したログデータに対して自動ライフサイクルを設定することで、ログの保管にかかるコストを削減できます。収集されたログ情報に対して、データの機密性とチームが求めるアクセスパターンに応じて、きめ細かくアクセスを制御できます。ツールを統合して、データを相関付け、視覚化し、データからインサイトを引き出すことができます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

組織内で AWS の使用量が増えれば、分散したワークロードと環境の数が増えます。これらのワークロードと環境が各々、その中のアクティビティに関するデータを生成するため、そのデータを取り込んでローカルに保存することが、セキュリティ運用にとって課題となります。セキュリティチームは、セキュリティ情報とイベント管理 (SIEM) システムなどのツールを使用して、分散したソースからデータを収集し、相関付け、分析、対応のワークフローを実行します。そのためには、さまざまなデータソースにアクセスするための複雑な許可一式を管理する必要があり、抽出、変換、ロード (ETL) プロセスの運用における追加のオーバーヘッドも生じます。

こうした課題を解消するには、セキュリティログデータの関連ソースをすべて、ログアーカイブアカウントに集約することを検討します。詳細については「複数のアカウントで AWS 環境を構成する」を参照してください。これには、ワークロードのすべてのセキュリティ関連データと、AWS CloudTrail、AWS WAF、Elastic Load Balancing、Amazon Route 53 などの AWS サービスによって生成されるログが含まれます。このデータを個別の AWS アカウント の標準化した場所に保存し、適切なクロスアカウントアクセス許可を設定しておくことには、利点がいくつかあります。ワークロードや環境が侵害された場合にその内部でのログの改ざん防止に役立ち、追加のツールの統合先を一元化できるだけでなく、データ保持とライフサイクルの設定モデルを簡素化できます。 データ主

権、コンプライアンス範囲、その他の規制の影響を評価して、セキュリティデータの保管場所と保持期間を複数設ける必要があるかどうかを判断します。

ログと検出結果をすぐに取り込んで標準化できるようにするため、ログアーカイブアカウントの Amazon Security Lake を評価します。Security Lake で、CloudTrail、Route 53、Amazon EKS、VPC フローログなどの一般的なソースから自動的にデータを取り込むように設定できます。AWS Security Hub CSPM を Security Lake のデータソースとして設定することで、Amazon GuardDuty や Amazon Inspector など他の AWS サービスの検出結果をログデータに関連付けることもできます。 サードパーティーのデータソース統合を利用したり、カスタムのデータソースを設定することもできます。すべての統合により、データが Open Cybersecurity Schema Framework (OCSF) フォーマットに標準化され、Amazon S3 バケットに Parquet ファイルとして保存されるため、ETL 処理は不要になります。

セキュリティデータを標準化された場所に保存することで、高度な分析機能を使用できるようになります。AWS は、AWS 環境で動作するセキュリティ分析ツールを、ログアーカイブのアカウントとは別のセキュリティツール用アカウントにデプロイすることを推奨しています。このアプローチなら、細かい統制を実装し、ログとログ管理プロセスの整合性と可用性を、ログにアクセスするツールとは別個に保護できます。 Amazon Athena などのサービスを使用して、複数のデータソースを関連付けるオンデマンドクエリを実行することを検討します。Quick などの視覚化ツールを組み込むこともできます。AI を活用したソリューションがますます普及し、検出結果を人間が読める要約や自然言語での対話に変換するなどの機能を提供しています。これらのソリューションは多くの場合、標準化したデータ保存場所をクエリ用に用意することで統合しやすくなります。

実装手順

1. ログアーカイブアカウントとセキュリティツール用アカウントを作成する
    a. AWS Organizations を使用して、セキュリティ組織単位の下にログアーカイブアカウントとセキュリティツール用アカウントを作成します。AWS Control Tower を使用して組織を管理している場合、ログアーカイブアカウントとセキュリティツール用アカウントが自動的に作成されます。必要に応じて、これらのアカウントにアクセスして管理するためのロールとアクセス許可を設定します。
2. セキュリティデータの標準化した保存場所を設定する
    a. セキュリティデータの標準化した保存場所の作成戦略を決定します。 これは、一般的なデータレイクアーキテクチャのアプローチ、サードパーティーのデータ製品、Amazon Security Lake などのオプションを使用して実行できます。AWS では、アカウント用にオプトインしている AWS リージョンからも、積極的に使用していない場合でもセキュリティデータを取得することを推奨しています。

3. 標準化した保存場所へのデータソースの公開を設定する
    a. セキュリティデータのソースを特定し、標準化された場所に公開するように設定します。ETL プロセスの開発が必要な形式ではなく、希望する形式でデータを自動的にエクスポートする方法を検討します。Amazon Security Lake を使用すると、サポートされている AWS ソースや統合されたサードパーティーシステムからデータを収集できます。
4. 標準化した場所にアクセスするためのツールを設定する
    a. Amazon Athena、Quick、サードパーティーソリューションなどのツールを設定して、標準化された場所にアクセスできるようにします。 これらのツールをセキュリティツール用アカウント外で動作するように設定し、ログアーカイブアカウントへのクロスアカウントの読み取りアクセス権を適宜設定します。Amazon Security Lake にサブスクライバーを作成し、これらのツールからデータにアクセスできるようにします。

リソース

関連するベストプラクティス:

- SEC01-BP01 アカウントを使用してワークロードを分ける
- SEC07-BP04 データのライフサイクル管理を定義する
- SEC08-BP04 アクセスコントロールを適用する
- OPS08-BP02 ワークロードログを分析する

関連ドキュメント:

- AWS ホワイトペーパー: 複数のアカウントで AWS 環境を構成する
- AWS Prescriptive Guidance: AWS Security Reference Architecture (AWS SRA)
- AWS Prescriptive Guidance: Logging and monitoring guide for application owners

関連する例:

- Amazon Athena と Quick を使用して分散ソースからのログデータを集約、検索、視覚化する
- Amazon Security Lake の調査結果を Quick で視覚化する方法
- Generate AI powered insights for Amazon Security Lake using Amazon SageMaker AI Studio and Amazon Bedrock
- Identify cybersecurity anomalies in your Amazon Security Lake data using Amazon SageMaker AI

- Ingest, transform, and deliver events published by Amazon Security Lake to Amazon OpenSearch Service
- CloudTrail Lake での自然言語クエリ生成による AWS CloudTrail ログ分析の簡素化

関連ツール:

- Amazon Security Lake
- Amazon Security Lake のパートナー
- Open Cybersecurity Schema Framework (OCSF)
- Amazon Athena
- Quick
- Amazon Bedrock

SEC04-BP03 セキュリティアラートを相関付けて充実させる

予期しないアクティビティが検知されると、さまざまなソースがそれぞれのセキュリティアラートを発します。状況を完全に理解するには、それらをさらに関連付けて情報を強化 (エンリッチメント) しなくてはなりません。セキュリティアラートの関連付けとエンリッチメントを自動化すると、インシデントの特定と対応をより正確に行えるようになります。

期待される成果: アクティビティによってワークロードや環境内でさまざまなアラートが生成されると、自動メカニズムがデータを関連付け、補足情報によってそのデータを強化します。この前処理のおかげで、イベントについて詳しく把握できるようになり、調査員はイベントの重要度や、正式な対応を必要とするインシデントであるかどうかを判断できます。これにより、監視チームと調査チームの負担が軽減します。

一般的なアンチパターン:

- 職務分掌の要件で別途義務付けられているわけでもないのに、さまざまなグループの人員がさまざまなシステムによって生成された検出結果やアラートを調査している。
- 組織はセキュリティ検出結果と警告データをすべて標準の保存先に集めているが、相関付けやエンリッチメントは調査員が手作業で行う必要がある。
- 検出結果の報告と重要度の決定は、脅威検出システムのインテリジェンスのみに頼っている。

このベストプラクティスを活用するメリット: アラートの関連付けや強化を自動化すると、調査担当者に求められる認知的な負担や手作業によるデータ準備を、全体的に減らすことができます。これにより、イベントがインシデントを示唆しているかどうかを判断し、正式な対応に着手するまでにかかる時間を短縮できます。また、文脈が補足されることで、イベントの実際の重大度を正確に評価できます。実際の重大度は、1 つのアラートが示す重大度よりも高い場合や低い場合が考えられます。

このベストプラクティスを活用しない場合のリスクレベル: 低

実装のガイダンス

セキュリティアラートは、次のような AWS 内のさまざまなソースが生成する可能性があります。

- Amazon GuardDuty、AWS Security Hub CSPM、Amazon Macie、Amazon Inspector、AWS Config、AWS Identity and Access Management Access Analyzer、Network Access Analyzer などのサービス。
- Amazon OpenSearch Service のセキュリティ分析など、AWS サービス、インフラストラクチャ、アプリケーションログの自動分析から発せられたアラート。
- Amazon CloudWatch、Amazon EventBridge、AWS Budgets などのソースからの、請求アクティビティの変化に反応したアラート。
- 脅威インテリジェンスのフィードや AWS Partner Networkのセキュリティパートナーソリューションなど、サードパーティーのソース。
- AWS Trust & Safety、または顧客や従業員などその他のソースからの連絡。
- Threat Technique Catalog by AWS (TTC) を使用して、侵害インジケータ (IoC) 識別による脅威アクターの行動の特定と相関を支援します。TTC は、MITRE ATT&CK フレームワークの拡張であり、AWS リソースを対象とするすべての既知の脅威アクターの動作と手法を分類します。

アラートには、最も基本的な形式で、誰が (プリンシパルまたはアイデンティティ)、何を (実行されたアクション)、何に (影響を受けるリソース) 対して行っているか、という情報が含まれます。これらのソースごとに、相関付けの土台として、アイデンティティ、アクション、リソースの識別子間のマッピングを作成する方法があるかどうかを確認してください。例えば、アラートソースをセキュリティ情報とイベント管理 (SIEM) ツールと統合して相関付けを自動で行う、独自のデータパイプラインを構築して処理する、またはその両方を組み合わせるという形が考えられます。

ユーザーに代わって関連付けを行うサービスの代表的な例が Amazon Detective です。Detective は、AWS やサードパーティーのさまざまなソースからのアラートを継続的に取り込み、さまざまな形式のインテリジェンスを使用してそれらの関係を視覚的にグラフ化して調査に役立てます。

アラートの初期の重要度は優先順位付けに役立ちますが、実際の重要度はアラートが発生した文脈によって決まります。例えば、Amazon GuardDuty が、ワークロード内の Amazon EC2 インスタンスが想定されていないドメイン名をクエリしたとしてアラートを発したとします。GuardDuty は、このアラートそれ自体には、重要度を「低」と判定しています。ただし、アラートの発生前後の他のアクティビティと自動で相関付けた結果、数百の EC2 インスタンスが同じアイデンティティによってデプロイされていて、全体的な運用コストが増加していることが判明しました。この場合、これの関連付けられたイベントのコンテキストを新しいセキュリティアラートとして公開し、重要度を「高」に調整する可能性があります。これを受けて、その後のアクションが迅速化します。

実装手順

1. セキュリティアラート情報のソースを特定します。これらのシステムからのアラートがアイデンティティ、アクション、リソースをどのように表すかを理解して、どこで相関付けることができるかを判断してください。
2. さまざまなソースからのアラートを取りまとめるメカニズムを確立します。この用途では、Security Hub CSPM、EventBridge、CloudWatch などのサービスを利用することを検討します。
3. データの相関付けとエンリッチメントのためのソースを特定します。ソースの例には、AWS CloudTrail、VPC フローログ、Route 53 Resolver ログ、インフラストラクチャ、およびアプリケーションログなどがあります。Amazon Security Lake との統合により、これらのログのすべてまたは一部が消費される可能性があります。
4. アラートをデータの相関付けやエンリッチメントのソースと統合して、セキュリティイベントのより詳細な文脈を作成し、重要度を設定します。
    a. Amazon Detective、SIEM ツール、その他のサードパーティーソリューションでは、一定レベルの取り込み、相関付け、エンリッチメントを自動的に実行できます。
    b. AWS サービスを使用して独自に構築することもできます。例えば、AWS Lambda 関数を呼び出して Amazon Athena クエリを AWS CloudTrail や Amazon Security Lake に対して実行し、結果を EventBridge にパブリッシュできます。

リソース

関連するベストプラクティス:

- SEC10-BP03 フォレンジック機能を備える
- OPS08-BP04 実践的なアラートを作成する
- REL06-BP03 通知を送信する (リアルタイム処理とアラーム)

関連ドキュメント:

- AWS セキュリティインシデント対応ガイド

関連する例:

- How to enrich AWS Security Hub CSPM findings with account metadata

関連ツール:

- Amazon Detective
- Amazon EventBridge
- AWS Lambda
- Amazon Athena

SEC04-BP04 非準拠リソースの修復を開始する

発見的統制は、構成要件に準拠していないリソースについて警告する場合があります。プログラムで定義された修復を手動または自動で開始して、該当するリソースを修正し、潜在的な影響を最小限に抑えることができます。プログラムで修復手順を定義しておくと、迅速に一貫した対応をすることができます。

自動化によってセキュリティの運用を強化できますが、自動化の実装と管理は慎重に行う必要があります。 適切な監視と統制のメカニズムを導入して、自動対応が効果的かつ正確であり、組織の方針やリスクアペタイトに合致していることを検証します。

期待される成果: リソース構成の標準を定義し、リソースが非準拠であることが検出された場合の修復手順も定義します。可能な場合は、修復手順をプログラムで定義して、手動または自動で開始できるようにしておきます。検知システムが導入されていて、このシステムが非準拠リソースを検知して、セキュリティ担当者が監視している一元管理ツールにアラートを発行します。これらのツールは、プログラムによる修復の手動実行または自動実行に対応しています。自動修復については、適切な監視と統制のメカニズムが導入され、その使用が管理されています。

一般的なアンチパターン:

- 自動化を実装しているが、修復アクションを徹底的にテストおよび検証できていない。正当な事業運営が中断されたり、システムが不安定になったりといった、意図しない結果が生じる可能性があります。

- 自動化によって応答時間と手続きは改善されたが、適切な監視や、必要に応じて人間が介入して判断できるメカニズムが欠如している。
- 修復だけに頼り、インシデント対応および復旧プログラムという広い枠組みの中に修復を組み込んでいない。

このベストプラクティスを活用するメリット: 自動修復は、手動プロセスよりも迅速に構成ミスに対応できるため、潜在的なビジネスへの影響が最小限に抑えられ、意図しない使用の可能性が低くなります。修復をプログラムで定義しておけば、一貫して適用されるため、人為的ミスのリスクが軽減されます。また、自動化により大量のアラートを同時に処理することもできます。これは、大規模な運用環境では特に重要です。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

「SEC01-BP03 管理目標を特定および検証する:」で説明したように、 AWS Config や AWS Security Hub CSPM などのサービスは、アカウント内のリソースの構成が要件に準拠しているかどうかを監視するのに役立ちます。非準拠のリソースが検出された場合、AWS Security Hub CSPM などのサービスは、アラートの適切なルーティングと修復に役立ちます。これらのソリューションは、セキュリティ調査員が問題を監視して是正措置を講じるための中心的な場所となります。

AWS Security Hub CSPM に加えて、AWS は Security Hub Advanced を導入しました。re:Invent 2025 で発表されたこのサービスは、組織が最も重要なセキュリティ問題を優先し、クラウド環境を保護するために大規模に対応する方法を変革します。拡張 Security Hub は、高度な分析を使用して、クラウド環境全体のセキュリティシグナルを自動的に関連付け、強化、優先順位付けするようになりました。Security Hub は、Amazon GuardDuty、Amazon Inspector、Amazon Macie、および AWS Security Hub CSPM とシームレスに統合されます。Security Hub の相関的な検出結果は、各リソースで見つかった脆弱性に基づいて想定される攻撃パスを含む、露出検出結果と呼ばれるまったく新しい検出結果になる可能性があります。

非準拠リソースの中には、状況が独特で修復には人間の判断が必要となる場合がありますが、プログラムで定義できる標準的な対応で間に合う状況もあります。例えば、VPC セキュリティグループが誤って設定されている場合、標準的な対応として、許可されていないルールを削除して所有者に通知することができます。応答は、AWS Lambda 関数や AWS Systems Manager Automation ドキュメントで定義するか、任意の他のコード環境で定義できます。是正措置を行うために必要最低限のアクセス許可しか持たない IAM ロールを使用して、その環境を AWS に対して認証できるようにしてください。

必要な修復手順を定義したら、その修復を開始する希望の方法を決定できます。AWS Config は修復を自動で開始できます。Security Hub CSPM を使用している場合は、この目的でカスタムアクションを使用し、検出結果の情報を Amazon EventBridge に公開できます。それを受けて、EventBridge ルールで修復を開始できます。Security Hub CSPM で修正を設定し、自動または手動で実行できます。

プログラムによる修復では、実行されたアクションとその結果について包括的なログ記録と監査を行うことをお勧めします。 これらのログを確認および分析して、自動化プロセスの有効性を評価し、改善すべき部分を特定します。ログは Amazon CloudWatch Logs に記録し、修復結果は Security Hub CSPM に検出結果メモとして記録します。

手始めに、AWS での自動化されたセキュリティ対応を検討してください。よくあるセキュリティの構成ミスを解決する修復機能が事前に組み込まれています。

実装手順

1. アラートを分析して優先順位を付けます。
   a. さまざまな AWS サービスから届くセキュリティアラートを Security Hub CSPM に統合して、可視化、優先順位付け、修復を一元的に行います。
2. 修復手順を考案します。
   a. Systems Manager や AWS Lambda などのサービスを使用して、プログラムによる修復を実行します。
3. 修復の開始方法を設定します。
   a. Systems Manager を使用して、検出結果を EventBridge に公開するカスタムアクションを定義します。これらのアクションを手動または自動で開始するように設定します。
   b. また、Amazon Simple Notification Service (SNS) を使用して、関連するステークホルダー (セキュリティチームやインシデント対応チームなど) に通知やアラートを送信し、必要に応じて手動による介入やエスカレーションを行うこともできます。
4. 修復ログを確認して分析し、有効性と改善点を検討します。
   a. ログ出力を CloudWatch Logs に送信します。結果を Security Hub CSPM に検出結果メモとして記録します。

リソース

関連するベストプラクティス:

- SEC06-BP03 手動管理とインタラクティブアクセスを削減する

関連ドキュメント:

- AWS Security Incident Response Guide - Detection

関連する例:

- での自動化されたセキュリティ対応AWS
- Monitor EC2 instance key pairs using AWS Config
- Create AWS Config custom rules by using AWS CloudFormation Guard policies
- Automatically remediate unencrypted Amazon RDS DB instances and clusters

関連ツール:

- AWS Systems Manager Automation
- での自動化されたセキュリティ対応AWS

インフラストラクチャの保護

インフラストラクチャ保護には、ベストプラクティスと組織の義務または規制上の義務に準拠するために必要な、多層防御などの制御手法が含まれています。クラウドでの継続的な運用を成功させるには、このような手法を使用することが非常に重要です。

インフラストラクチャ保護は、情報セキュリティプログラムの重要な部分です。意図しない不正アクセスや潜在的な脆弱性から、ワークロード内のシステムとサービスを保護できます。例えば、信頼境界 (ネットワークとアカウントの境界など)、システムセキュリティの設定とメンテナンス (強化、最小化、パッチ適用など)、オペレーティングシステムの認証と承認 (ユーザー、キー、アクセスレベルなど)、その他の適切なポリシー適用ポイント (ウェブアプリケーションファイアウォールや API ゲートウェイなど) を定義します。

リージョン、アベイラビリティーゾーン、AWS Local Zones、および AWS Outposts

AWS の安全なグローバルインフラストラクチャのコンポーネントであるリージョン、アベイラビリティーゾーン、AWS Local Zones、および AWS Outposts を理解する必要があります。

AWS にはリージョンという概念が存在します。これは、データセンターが集積されている世界中の物理的ロケーションのことです。論理的データセンターの各グループは、アベイラビリティーゾーン (AZ) と呼ばれます。各 AWS リージョンは、1 つの地理的領域内にある、複数の、隔離され、物理的にも分かれている AZ で成り立っています。データレジデンシー要件がある場合は、目的の場所の近くにある AWS リージョンを選択できます。データが物理的に配置されているリージョンに対する完全な制御と所有権を保持することで、地域のコンプライアンス要件やデータレジデンシー要件を満たすのに役立ちます。各 AZ には、独立した電源、冷却、および物理的セキュリティが備わっています。アプリケーションを AZ 間でパーティショニングすると、停電、落雷、竜巻、地震などの問題から、よりよく隔離され保護されます。各 AZ はそれぞれ他の AZ から物理的に意味のある距離、つまり数キロメートル離れていますが、互いにすべて 100 km (60 マイル) 以内に配置されています。AWS リージョン内のすべての AZ は、AZ 間に高スループットかつ低レイテンシーのネットワーキングを提供する、完全に冗長性を持つ専用メトロファイバー上に構築された、高帯域幅、低レイテンシーのネットワーキングで相互接続されています。AZ 間のすべてのトラフィックは暗号化されています。高度な可用性の実現にフォーカスしている AWS のお客様は、複数の AZ で実行するようにアプリケーションの設計をすることで、より強力な障害耐性を実現できます。AWSリージョンは、最高レベルのセキュリティ、コンプライアンス、データ保護を実現します。

AWS Local Zones では、コンピューティング、ストレージ、データベース、およびその他の特定の AWS サービスをエンドユーザーから近い場所に配置します。AWS Local Zones では、メディア工

ンターテインメントのコンテンツ制作、リアルタイムゲーミング、貯水池のシミュレーション、電子自動設計、機械学習など、エンドユーザーに対するレイテンシーが 10 ミリ秒未満であることが要求される高性能なアプリケーションを簡単に実行できます。各 AWS Local Zone ロケーションは AWS リージョンを拡張したものであり、Amazon EC2、Amazon VPC、Amazon EBS、Amazon File Storage、および Elastic Load Balancing などの AWS のサービスを使用して、地理的にエンドユーザーと近い場所で、レイテンシーの影響を受けやすいアプリケーションを実行できます。AWSLocal Zones は、ローカルと AWS リージョンでそれぞれ実行中のワークロード間で高帯域幅かつ安全な接続が利用できます。同じ API とツールセットを介してすべてのリージョン内サービスにシームレスに接続します。

AWS Outposts では、ネイティブの AWS サービス、インフラストラクチャ、運用モデルをほぼすべてのデータセンター、コロケーションスペース、オンプレミスの施設で利用できます。同じ AWS の API、ツール、インフラストラクチャをオンプレミス全体と AWS クラウドで使用できるため、真に一貫したハイブリッドエクスペリエンスが提供されます。AWSOutposts はコネクテッド環境向けに設計されたものです。低レイテンシー、もしくはローカルでデータを処理する必要があるためにオンプレミスに残されているワークロードをサポートできます。

AWS では、いくつかの方法でインフラストラクチャを保護できます。以下の各セクションでは、こうしたアプローチの使用方法を説明します。

トピック

ネットワークの保護

従業員も顧客も、ユーザーはどこにでも存在する可能性があります。ネットワークにアクセスできる人なら誰でも、何でも信用するという従来のモデルから脱却する必要があります。すべてのレイヤーでセキュリティを適用するという原則はつまり、ゼロトラストアプローチを適用することです。ゼロトラストセキュリティは、アプリケーションのコンポーネントやマイクロサービスを互いに分離して考え、どのコンポーネントやマイクロサービスも他のものを信用しないというモデルです。

ネットワーク設計を慎重に計画し管理することで、ワークロード内のリソースを分離し境界を作るための基礎が形成されます。ワークロードのリソースの多くは VPC 内で動作し、セキュリティのプロパティを継承するため、自動化によって支えられた検査および保護メカニズムで設計をサポートすることが重要です。同様に、純粋なエッジサービスやサーバーレスを使用して VPC の外部で動作するワークロードの場合は、よりシンプルなアプローチでベストプラクティスを適用します。サーバーレ

スセキュリティに関する具体的なガイダンスについては、「AWS Well-Architected サーバーレスアプリケーションレンズ」を参照してください。

ベストプラクティス

- SEC05-BP01 ネットワークレイヤーを作成する
- SEC05-BP02 ネットワークレイヤー内のトラフィックフローを制御する
- SEC05-BP03 検査に基づく保護を実装する
- SEC05-BP04 ネットワーク保護を自動化する

SEC05-BP01 ネットワークレイヤーを作成する

ワークロードコンポーネントをそのデータの機密度とアクセス要件に応じて論理的にグループ化し、そのグループに基づいてネットワークトポロジをさまざまなレイヤーに分割します。インターネットからのインバウンドアクセスを必要とするコンポーネント (公開ウェブエンドポイントなど) と、内部アクセスのみを必要とするコンポーネント (データベースなど) は区別します。

期待される成果: ネットワークのレイヤーが多層防御のセキュリティ対策全体の一翼を担い、ワークロードのアイデンティティ認証や承認戦略を補完しています。データの機密度とアクセス要件に応じてレイヤーが配置され、トラフィックフローと統制のメカニズムが適切に機能しています。

一般的なアンチパターン:

- 単一の VPC またはサブネットですべてのリソースを作成する。
- データの機密度の要件、コンポーネントの動作、機能を考慮せずにネットワークレイヤーを構築する。
- ネットワークレイヤーのすべての考慮事項について、デフォルトとして VPC とサブネットを使用し、AWS マネージドサービスがトポロジに与える影響を考慮していない。

このベストプラクティスを活用するメリット: ネットワークレイヤーを確立することは、ネットワーク内の不要な経路、特に重要なシステムやデータにつながる経路を制限するための第一歩です。これにより、権限のない攻撃者がネットワークにアクセスしたり、ネットワーク内の他のリソースを操作したりしづらくなります。ネットワークレイヤーが分離されていると、侵入検知やマルウェア防止などの検査システムの分析範囲が狭くなるという利点があります。そのおかげで、誤検出や不要な処理に伴うオーバーヘッドの発生確率が下がります。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

ワークロードのアーキテクチャを設計する場合、一般的には、コンポーネントをそれぞれの役割に応じて異なるレイヤーに分けます。例えば、ウェブアプリケーションは、プレゼンテーションレイヤー、アプリケーションレイヤー、データレイヤーで構成できます。ネットワークトポロジを設計する際も似たようなアプローチを採用できます。基盤にあるネットワーク統制が、ワークロードのデータアクセス要件を適用するのに役立つことがあります。例えば、3 層のウェブアプリケーションのアーキテクチャでは、プレゼンテーションレイヤーの静的ファイルを Amazon S3 に保存し、それらのファイルを Amazon CloudFront などのコンテンツ配信ネットワーク (CDN) から提供できます。アプリケーションレイヤーには、Application Load Balancer (ALB) が Amazon VPC のパブリックサブネット (非武装地帯 (DMZ) と類似) で提供するパブリックエンドポイントを設け、バックエンドのサービスをプライベートサブネットにデプロイできます。データレイヤーは、データベースや共有ファイルシステムなどのリソースをホストします。アプリケーションレイヤーのリソースとは異なるプライベートサブネットに配置できます。これらのレイヤー境界 (CDN、パブリックサブネット、プライベートサブネット) のそれぞれで統制を効かせて、承認されたトラフィックしかそれらの境界を超えられないようにすることができます。

ワークロードのコンポーネントの機能面の目的に基づいてネットワークレイヤーをモデル化するのと同様に、処理対象のデータの機密度も考慮してください。ウェブアプリケーションの例で考えてみると、ワークロードサービスはすべてアプリケーションレイヤー内にある一方で、それぞれのサービスが処理するデータの機密度は異なる場合があります。この場合、統制の要件によっては、複数のプライベートサブネット、同じ AWS アカウントの異なる VPC、または異なる AWS アカウントの異なる VPC を使用して、データの機密度ごとにアプリケーションレイヤーを分割した方が適切なこともあります。

ネットワークレイヤーについてさらに考慮すべき点は、ワークロードのコンポーネントの動作の一貫性です。引き続き同じ例で言えば、アプリケーションレイヤーにエンドユーザーや統合先の外部システムからの入力を受け入れるサービスがあり、その入力のリスクが他のサービスへの入力と比べて本質的に高いという場合が考えられます。例えば、ファイルのアップロード、実行するコードスクリプト、メールのスキャンなどが該当します。こうしたサービスを独自のネットワークレイヤーに配置すれば、より強固な分離境界を張り巡らせることができ、それらの固有の動作のせいで検査システムで誤検知アラートが発生するのを防止できます。

設計の過程で、AWS マネージドサービスを使用することで、ネットワークトポロジにどのような影響があるかを検討してください。Amazon VPC Lattice などのサービスが、ネットワークレイヤー間のワークロードコンポーネントの相互運用性の向上にいかに役立つかをご確認ください。AWS Lambda を使用する場合は、特に理由がない限り、VPC サブネットにデプロイしてください。イン

ターネットゲートウェイへのアクセスを制限するセキュリティポリシーの遵守を VPC エンドポイントや AWS PrivateLink で簡素化できるのはどこか、判断してください。

実装手順

1. ワークロードのアーキテクチャを見直します。コンポーネントやサービスを、それらが提供する機能、処理対象のデータの機密度、動作に基づいて論理的にグループ化します。
2. インターネットからのリクエストに応答するコンポーネントについては、ロードバランサーやその他のプロキシを使用してパブリックエンドポイントを設けることを検討します。CloudFront、Amazon API Gateway、Elastic Load Balancing、AWS Amplify などのマネージドサービスを利用してパブリックエンドポイントをホストすることで、セキュリティ統制を移行することを検討してください。
3. Amazon EC2 インスタンス、AWS Fargate コンテナ、Lambda 関数など、コンピューティング環境で実行されるコンポーネントの場合、手順 1 で決めたグループに基づいて、これらをプライベートサブネットにデプロイします。
4. Amazon DynamoDB、Amazon Kinesis、Amazon SQS などのフルマネージド型の AWS サービスについては、プライベート IP アドレス経由のアクセスにはデフォルトで VPC エンドポイントを使用することを検討します。

リソース

関連するベストプラクティス:

- REL02 ネットワークトポロジを計画する
- PERF04-BP01 ネットワークがパフォーマンスに与える影響を理解する

関連動画:

- AWS re:Invent 2023 - AWS networking foundations

関連する例:

- VPC の例
- Access container applications privately on Amazon ECS by using AWS Fargate, AWS PrivateLink, and a Network Load Balancer
- Serve static content in an Amazon S3 bucket through a VPC by using Amazon CloudFront

SEC05-BP02 ネットワークレイヤー内のトラフィックフローを制御する

ネットワークのレイヤー内でさらにセグメンテーションを行い、各ワークロードに必要なフローのみにトラフィックを制限します。まず、インターネットや他の外部システムからワークロードや環境へのトラフィック (North-South トラフィック) の制御に着目します。その後、さまざまなコンポーネントとシステム間のフロー (East-West トラフィック) を確認します。

期待される成果: ワークロードのコンポーネントが相互通信や、クライアントや依存先のその他のサービスとの通信に必要とするネットワークフローのみを許可します。設計では、パブリックとプライベートの送受信、データ分類、地域の規制、プロトコル要件などが考慮されます。可能な限り、最小特権の原則に基づく設計の一環として、ネットワークピアリングよりもポイントツーポイントフローを優先します。

一般的なアンチパターン:

- ネットワークセキュリティに境界ベースのアプローチを採用し、ネットワークレイヤーの境界でのみトラフィックフローを制御する。
- ネットワークレイヤー内のすべてのトラフィックが認証済み、承認済みだと仮定している。
- 受信トラフィックと送信トラフィックのいずれかに制御を適用し、両方には適用していない。
- トラフィックの認証と承認をワークロードコンポーネントとネットワーク統制のみに頼っている。

このベストプラクティスを活用するメリット: このプラクティスを実践することで、ネットワーク内の不正な移動のリスクを軽減し、ワークロードに承認のレイヤーを追加できます。トラフィックフロー制御を行うことで、セキュリティインシデントによる影響の範囲を制限し、検出と対応をスピードアップできます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

ネットワークレイヤーによって、機能、データの機密レベル、動作が似ているワークロードのコンポーネント群の周りに境界を確立できますが、最小特権の原則に従ってこれらのレイヤー内のコンポーネントをさらにセグメント化する手法を用いて、よりきめ細かくトラフィックを制御できます。AWS 内では、ネットワークレイヤーは主に、Amazon VPC 内の IP アドレス範囲に応じたサブネットを使用して定義されます。また、マイクロサービス環境をビジネスドメインごとにグループ化するなど、さまざまな VPC を使用してレイヤーを定義することもできます。複数の VPC を使用する場合は、AWS Transit Gateway を使用してルーティングを行います。この場合、セキュリティ

グループとルートテーブルを使用してレイヤー 4 レベル (IP アドレスとポートの範囲) でトラフィックを制御できますが、AWS PrivateLink、Amazon Route 53 Resolver DNS Firewall、AWS Network Firewall、AWS WAF などの追加サービスを使用して制御を強化することもできます。

ワークロードのデータフローと通信の要件を接続の開始側、ポート、プロトコル、ネットワークレイヤーの観点から把握し、インベントリを作成します。接続の確立とデータ転送に使用できるプロトコルを評価して、保護要件を満たすプロトコル (例えば、HTTP ではなく HTTPS) を選択します。ネットワークの境界と各レイヤー内の両方で、これらの要件を把握してください。これらの要件を特定できたら、オプションを検討し、必要なトラフィックのみが各接続ポイントを流れるようにします。まず、VPC 内でセキュリティグループを使用することをお勧めします。セキュリティグループは、Amazon EC2 インスタンス、Amazon ECS タスク、Amazon EKS ポッド、Amazon RDS データベースなど、Elastic Network Interface (ENI) を使用するリソースにアタッチできます。レイヤー 4 のファイアウォールとは異なり、セキュリティグループには別のセキュリティグループからのトラフィックを識別子ごとに許可するルールを設定できるため、グループ内のリソースが時間の経過とともに変化しても更新を最小限に抑えることができます。セキュリティグループを使用し、インバウンドルールとアウトバウンドルールの両方を用いて、トラフィックをフィルタリングすることもできます。

トラフィックが VPC 間を移動する場合、シンプルルーティングには VPC ピアリングを使用し、複雑なルーティングには AWS Transit Gateway を使用するのが一般的です。これらのアプローチにより、送信元ネットワークと宛先ネットワークの両方の IP アドレス範囲間のトラフィックフローが円滑になります。ただし、異なる VPC にある特定のコンポーネント間のトラフィックフローのみをワークロードが必要とする場合は、AWS PrivateLink を使用してポイントツーポイント接続を確立することを検討してください。その場合は、プロデューサーとして機能するサービスと、コンシューマーとして機能するサービスを特定します。プロデューサーには互換性のあるロードバランサーをデプロイし、PrivateLink を適宜有効にしてから、コンシューマーからの接続リクエストを受け入れます。その後、プロデューサーサービスには、コンシューマーの VPC からプライベート IP アドレスが割り当てられます。コンシューマーはこれを使用して以降のリクエストを行うことができます。この方法だと、ネットワークのピアリングはほとんど必要なくなります。PrivateLink の評価の中で、データ処理と負荷分散のコストも考慮してください。

セキュリティグループや PrivateLink は、ワークロードのコンポーネント間のフロー制御に役立ちますが、もう 1 つの重要な考慮事項は、リソースがアクセスできる DNS ドメイン (存在する場合) を制御する方法です。VPC の DHCP 構成に応じて、この目的で 2 つの異なる AWS サービスを検討できます。大半のお客様は、VPC で CIDR 範囲 +2 のアドレスを利用できるデフォルトの Route 53 Resolver DNS サービス (別称 Amazon DNS サーバーまたは AmazonProvidedDNS) を使用します。この方法では、DNS ファイアウォールルールを作成して VPC に関連付け、指定したドメインリストに対して実行するアクションを決定できます。

Route 53 Resolver を使用していない場合や、ドメインフィルタリング以外の詳細な検査やフロー制御の機能で Resolver を補完したい場合は、AWS Network Firewall の デプロイを検討してください。このサービスは、ステートレスルールまたはステートフルルールを使用して個々のパケットを検査し、トラフィックを拒否するか許可するかを決定します。AWS WAF を使用してパブリックエンドポイントへのインバウンドウェブトラフィックをフィルタリングする場合も、同様のアプローチを採用できます。これらのサービスの詳細については、「SEC05-BP03 検査に基づく保護を実装する」を参照してください。

実装手順

1. ワークロードのコンポーネント間で必要なデータフローを特定します。
2. セキュリティグループやルートテーブルの使用など、インバウンドトラフィックとアウトバウンドトラフィックの両方に、多層防御のアプローチで複数の統制を適用します。
3. Route 53 Resolver DNS Firewall、AWS Network Firewall、AWS WAF など、ファイアウォールを使用して、VPC に出入りするネットワークトラフィックに対する制御をきめ細かく定義します。組織全体でファイアウォールルールを一元的に設定および管理するには、AWS Firewall Manager の使用を検討してください。

リソース

関連するベストプラクティス:

- REL03-BP01 ワークロードをセグメント化する方法を選択する
- SEC09-BP02 伝送中に暗号化を適用する

関連ドキュメント:

- VPC のセキュリティのベストプラクティス
- AWS Network Optimization Tips
- Guidance for Network Security on AWS
- Secure your VPC's outbound network traffic in the AWS クラウド

関連ツール:

- AWS Firewall Manager

関連動画:

- AWS Transit Gateway reference architectures for many VPCs
- Application Acceleration and Protection with Amazon CloudFront, AWS WAF, and AWS Shield
- AWS re:Inforce 2023: Firewalls and where to put them

SEC05-BP03 検査に基づく保護を実装する

ネットワークレイヤー間にトラフィックの検査ポイントを設定して、転送中のデータが、予想されるカテゴリやパターンと一致していることを確認します。 トラフィックフロー、メタデータ、パターンを分析して、イベントの識別、検出、対応をより効果的に行えるようにします。

期待される成果: ネットワークレイヤー間を通過するトラフィックが検査され、承認されます。 許可と拒否の決定は、明示的なルールや脅威インテリジェンス、ベースラインの動作から逸脱しているかどうかに基づいて行われます。 トラフィックが機密データに近づくにつれて、保護は厳格化されます。

一般的なアンチパターン:

- ポートとプロトコルに基づくファイアウォールルールのみに依存している。インテリジェントシステムを利用していない。
- 特定の最新の脅威パターンに基づいてファイアウォールルールを作成しているが、このパターンは変更される可能性がある。
- トラフィックがプライベートサブネットからパブリックサブネットに、またはパブリックサブネットからインターネットに転送される箇所のみを検査している。
- 動作の異常の比較基準となるネットワークトラフィックのベースラインビューがない。

このベストプラクティスを活用するメリット: 検査システムでは、トラフィックデータが特定の条件に該当する場合にのみトラフィックを許可または拒否するなど、インテリジェントなルールを作成できます。脅威の状況は時間とともに変化するので、AWS やパートナーが最新の脅威インテリジェンスに基づいて提供するマネージドルールセットが役立ちます。 これにより、ルールの維持や侵害の兆候の調査にかかるオーバーヘッドが減り、誤検出率が下がります。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

AWS Network Firewall や、Gateway Load Balancer (GWLB) の背後にデプロイできる AWS Marketplace の他のファイアウォールおよび侵入防止システム (IPS) を使用して、ステートフルネットワークトラフィックとステートレスネットワークトラフィックの両方をきめ細かく制御します。AWS Network Firewall は、Suricata 互換のオープンソースの IPS 仕様に対応し、ワークロードの保護に役立ちます。

AWS Network Firewall と、GWLB を使用するベンダーソリューションはどちらも、さまざまなインライン検査デプロイモデルをサポートしています。 例えば、VPC 単位で検査を実行することや、検査用 VPC に一元化することができます。また、ハイブリッドモデルでデプロイし、East-West トラフィックは検査用 VPC に流し、インターネットの受信トラフィックは VPC 単位で検査することもできます。 もう 1 つ考慮すべき点は、そのソリューションが Transport Layer Security (TLS) のラップ解除に対応しているかどうかです。これにより、どちらの方向から開始されたトラフィックフローでもディープパケット検査が可能になります。これらの構成の詳細については、AWS Network Firewall のベストプラクティスガイドを参照してください。

プロミスキャスモードで動作しているネットワークインターフェイスからのパケットデータの pcap 分析など、アウトオブバンド検査を実行するソリューションを使用している場合は、VPC トラフィックミラーリングを設定できます。ミラーリングされたトラフィックは、インターフェイスで使用可能な帯域幅にカウントされ、ミラーリングされていないトラフィックと同じデータ転送料金が課されます。これらのアプライアンスの仮想バージョンが AWS Marketplace で提供されているかどうかを確認できます。GWLB の背後のインラインデプロイに対応している場合があります。

HTTP ベースのプロトコルを介してトランザクションを行うコンポーネントの場合は、ウェブアプリケーションファイアウォール (WAF) で一般的な脅威からアプリケーションを保護します。AWS WAF は、HTTP(S) リクエストを監視し、設定可能なルールに一致するものを Amazon API Gateway、Amazon CloudFront、AWS AppSync、または Application Load Balancer に送信する前にブロックできるウェブアプリケーションファイアウォールです。ウェブアプリケーションファイアウォールのデプロイを評価するときは、ディープパケット検査を検討してください。一部、トラフィック検査の前に TLS を終了しなければならないものがあるためです。AWS WAF の使用を開始するには、AWS マネージドルールを独自のルールと組み合わせて使用するか、既存のパートナー統合を使用できます。

AWS Firewall Manager を使用して、AWS Organization 全体で AWS WAF、AWS Shield Advanced、AWS Network Firewall、Amazon VPC セキュリティグループを一元管理できます。

実装手順

1. 検査ルールの範囲を広く設定できるか (検査用 VPC を使用するなど)、または VPC 単位のよりきめ細かいアプローチが必要かどうかを判断します。
2. インライン検査ソリューションの場合:
    a. AWS Network Firewall を使用する場合は、ルール、ファイアウォールポリシー、ファイアウォール自体を作成します。これらを設定したら、トラフィックをファイアウォールエンドポイントにルーティングして検査を有効にすることができます。
    b. Gateway Load Balancer (GWLB) とサードパーティー製アプライアンスを使用する場合は、アプライアンスを 1 つ以上のアベイラビリティーゾーンにデプロイして構成します。次に、GWLB、エンドポイントサービス、エンドポイントを作成し、トラフィックのルーティングを設定します。
3. アウトオブバンド検査ソリューションの場合:
    1. インバウンドトラフィックとアウトバウンドトラフィックをミラーリングする必要があるインターフェイスで VPC トラフィックミラーリングを有効にします。Amazon EventBridge ルールを使用して、新しいリソースが作成されたときに AWS Lambda 関数を呼び出してインターフェイスでトラフィックミラーリングを有効にすることができます。トラフィックミラーリングセッションを、トラフィックを処理するアプライアンスの前にある Network Load Balancer に送ります。
4. インバウンドのウェブトラフィックソリューションの場合:
    a. AWS WAF を設定するには、まずウェブアクセスコントロールリスト (ウェブ ACL) を設定します。ウェブ ACL は、WAF がトラフィックを処理する方法を定義する、逐次処理されるデフォルトアクション (ALLOW または DENY) を指定したルールのコレクションです。独自のルールとグループを作成することも、ウェブ ACL で AWS マネージドルールグループを使用することもできます。
    b. ウェブ ACL を設定したら、ウェブ ACL を AWS リソース (Application Load Balancer、API Gateway REST API、CloudFront ディストリビューションなど) に関連付けて、ウェブトラフィックの保護を開始します。

リソース

関連ドキュメント:

- What is Traffic Mirroring?
- Implementing inline traffic inspection using third-party security appliances

- AWS Network Firewall example architectures with routing
- Centralized inspection architecture with AWS Gateway Load Balancer and AWS Transit Gateway

関連する例:

- ゲートウェイロードバランサーをデプロイするためのベストプラクティス
- TLS inspection configuration for encrypted egress traffic and AWS Network Firewall

関連ツール:

- AWS Marketplace IDS/IPS

SEC05-BP04 ネットワーク保護を自動化する

Infrastructure as Code (IaC) や CI/CD パイプラインなどの DevOps のプラクティスを活用して、ネットワーク保護のデプロイを自動化します。 これらのプラクティスに則って、ネットワーク保護の変更をバージョン管理システムを通じて追跡し、変更のデプロイにかかる時間を短縮できます。また、ネットワーク保護が目的の設定から逸脱していないかどうかを検知できます。

期待される成果: ネットワーク保護をテンプレートに定義して、バージョン管理システムにコミットします。 新しい変更が加えられると、自動パイプラインが開始して、そのテストとデプロイを調整します。 変更をデプロイ前に検証するための、ポリシーチェックやその他の静的テストが整備されています。 変更をステージング環境にデプロイして、統制が予期したとおりに機能していることを検証します。 統制が承認されると、本番環境へのデプロイも自動的に実行されます。

一般的なアンチパターン:

- 個々のワークロードチームに、ネットワークスタック、保護、自動化の完全な定義を任せている。 ネットワークスタックと保護の標準的な側面が、ワークロードチームが利用できるように中央で公開されていない。
- ネットワーク、保護、自動化のあらゆる側面の定義を中央のネットワークチームに一任している。 ネットワークスタックと保護のワークロード固有の側面を、そのワークロードのチームに委任していない。
- ネットワークチームとワークロードチームの間で一元管理と委任の適切なバランスを取っているが、一貫したテストとデプロイの標準がすべての IaC テンプレートと CI/CD パイプラインにわたっては適用されていない。 テンプレートの準拠状況をチェックするために必要な設定が、ツールに取り込まれていない。

このベストプラクティスを活用するメリット: テンプレートにネットワーク保護を定義しておくと、経時的な変更をバージョン管理システムで追跡し、比較できます。 変更のテストとデプロイを自動化することで、プロセスが標準化されて予測可能性が高まり、デプロイの成功率が上がり、繰り返しの手動設定を省くことができます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

「SEC05-BP02 ネットワークレイヤー内のトラフィックフローを制御する」や「SEC05-BP03 検査に基づく保護を実装する」で説明されている多くのネットワーク保護統制では、最新の脅威インテリジェンスを反映して自動更新できるマネージドルールシステムを採用します。 ウェブエンドポイントを保護する例としては、AWS WAF マネージドルールや AWS Shield Advanced アプリケーションレイヤー DDoS の自動緩和などがあります。AWS Network Firewall マネージドルールグループを使用すれば、レピュテーションの低いドメインリストや脅威シグネチャの最新情報が随時反映されます。

マネージドルール以外にも、DevOps のプラクティスを使用して、指定したネットワークリソース、保護、ルールのデプロイを自動化することをお勧めします。 これらを AWS CloudFormation や任意の Infrastructure as Code (IaC) ツールに定義してバージョン管理システムにコミットし、CI/CD パイプラインを使用してデプロイできます。 この方法なら、リリースの予測可能性の向上、AWS CloudFormation Guard などのツールを使用した自動テスト、デプロイした環境が目的の構成とずれている場合の検知など、ネットワーク統制管理に関する DevOps の従来のメリットが得られます。

「SEC05-BP01 ネットワークレイヤーを作成する」の過程で行った決断に伴い、一元管理のアプローチで受信フロー、送信フロー、検査フロー専用の VPC を作成する場合があります。 「AWS Security Reference Architecture (AWS SRA)」で説明されているとおり、これらの VPC は専用のネットワークインフラストラクチャアカウントで定義できます。 同様の手法を用いて、他のアカウントのワークロード、そのセキュリティグループ、AWS Network Firewall デプロイ、Route 53 Resolver ルールと DNS ファイアウォール構成、その他のネットワークリソースが使用する VPC を一元的に定義できます。 これらのリソースは、AWS Resource Access Manager を使用して他のアカウントと共有できます。 このアプローチなら、管理先が 1 つになり、ネットワーク統制の自動テストとネットワークアカウントへの自動デプロイを簡素化できます。 これは、ハイブリッドモデルで実現できます。つまり、特定の統制を一元的にデプロイして共有し、他の統制を個々のワークロードチームとそれぞれ該当するアカウントに委任します。

実装手順

1. ネットワークと保護のどの側面を一元的に定義し、どの側面をワークロードチームで管理できるのか、所有権を明確にします。
2. ネットワークとその保護に対する変更をテストし、デプロイする環境を作成します。 例えば、ネットワークテストアカウントとネットワーク本稼働アカウントを用意します。
3. テンプレートをバージョン管理システムに保存し、管理する方法を決定します。 中央テンプレートはワークロードリポジトリとは別のリポジトリに保存し、ワークロードテンプレートはそのワークロード固有のリポジトリに保存できます。
4. テンプレートをテストしてデプロイするための CI/CD パイプラインを作成します。 設定ミスがないかチェックし、テンプレートが会社の標準に準拠していることを確認するテストを定義します。

リソース

関連するベストプラクティス:

- SEC01-BP06 標準的なセキュリティ統制のデプロイを自動化する

関連ドキュメント:

- AWS セキュリティリファレンスアーキテクチャ - Network account

関連する例:

- AWS Deployment Pipeline Reference Architecture
- NetDevSecOps to modernize AWS networking deployments
- Integrating AWS CloudFormation security tests with AWS Security Hub CSPM and AWS CodeBuild reports

関連ツール:

- AWS CloudFormation
- AWS CloudFormation Guard
- cfn_nag

コンピューティングの保護

コンピューティングリソースには、EC2 インスタンス、コンテナ、AWS Lambda 関数、データベースサービス、IoT デバイスなどがあります。これらのコンピューティングリソースタイプには、それぞれ異なるアプローチでセキュリティを確保する必要があります。しかし、深層防御、脆弱性管理、アタックサーフェスの縮小、設定と運用の自動化、遠隔操作など、検討すべき戦略は共通しています。このセクションでは、主要なサービスのためのコンピューティングリソースを保護するための一般的なガイダンスを紹介します。使用される各 AWS サービスについて、サービス文書に記載されている具体的なセキュリティ推奨事項を確認することが重要です。

ベストプラクティス

- SEC06-BP01 脆弱性管理を実行する
- SEC06-BP02 強化されたイメージからコンピューティングをプロビジョニングする
- SEC06-BP03 手動管理とインタラクティブアクセスを削減する
- SEC06-BP04 ソフトウェアの整合性を検証する
- SEC06-BP05 コンピューティング保護を自動化する

SEC06-BP01 脆弱性管理を実行する

コード、依存関係、インフラストラクチャ内の脆弱性のスキャンとパッチ適用を頻繁に実施し、新しい脅威から保護します。

期待される成果: ソフトウェアの脆弱性、潜在的な欠陥、意図しないネットワークへの露出がないかワークロードを継続的にスキャンするソリューションがあります。リスク評価基準に基づいて、これらの脆弱性を特定、優先順位付け、および修正するためのプロセスと手順を確立しました。さらに、コンピューティングインスタンスに自動パッチ管理を実装しました。脆弱性管理プログラムは、CI/CD パイプライン中にソースコードをスキャンするソリューションを使用して、ソフトウェア開発ライフサイクルに統合されています。

一般的なアンチパターン:

- 脆弱性管理プログラムがない。
- 重大度またはリスク回避を考慮せずに、システムパッチ適用を実施する。
- ベンダーが提供する耐用年数 (EOL) を過ぎたソフトウェアを使用する。
- セキュリティの問題を分析する前に、本番環境にコードをデプロイする。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

脆弱性管理は、安全で堅牢なクラウド環境を維持するうえで重要な要素です。これには、セキュリティスキャン、問題の特定と優先順位付け、特定された脆弱性を解決するためのパッチの適用など、包括的なプロセスが含まれます。このプロセスでは、潜在的な問題や意図しないネットワークへの露出、および修復作業に対するワークロードの継続的なスキャンを容易にするため、自動化はこのプロセスで重要な役割を果たします。

AWS 責任共有モデルは、脆弱性管理を支える基本的な概念です。このモデルによると、AWS は、AWS のサービスを実行するハードウェア、ソフトウェア、ネットワーク、施設など、基盤となるインフラストラクチャを保護する責任があります。一方で、ユーザーは Amazon EC2 インスタンスや Amazon S3 オブジェクトなどのサービスに関連するデータ、セキュリティ設定、管理タスクを保護する責任があります。

AWS は、脆弱性管理プログラムをサポートするさまざまなサービスを提供します。Amazon Inspector は、ソフトウェアの脆弱性や意図しないネットワークアクセスがないか、AWS ワークロードを継続的にスキャンし、AWS Systems Manager Patch Manager は Amazon EC2 インスタンス間のパッチ適用の管理に役立ちます。これらのサービスは、AWS セキュリティチェックの自動化、セキュリティアラートの一元化、組織のセキュリティ体制の包括的なビューを提供するクラウドセキュリティ体制管理サービスである AWS Security Hub CSPM と統合できます。さらに、Amazon CodeGuru Security は静的コード分析を使用して、開発フェーズ中の Java および Python アプリケーションの潜在的な問題を特定します。

ソフトウェア開発ライフサイクルに脆弱性管理プラクティスを組み込むことにより、本番環境に導入される前に、脆弱性をプロアクティブに解決できるため、セキュリティイベントのリスクが軽減され、脆弱性の潜在的な影響を最小限に抑えることができます。

実装手順

1. 責任共有モデルを理解する: AWS 責任共有モデルを確認して、クラウド内のワークロードとデータを保護する責任を理解します。AWS は基盤となるクラウドインフラストラクチャを保護する責任があり、ユーザーは使用するアプリケーション、データ、サービスを保護する責任があります。
2. 脆弱性スキャンの実装: Amazon Inspector などの脆弱性スキャンサービスを設定すると、ソフトウェアの脆弱性、潜在的な欠陥、意図しないネットワークへの流出がないか、コンピューティングインスタンス (仮想マシン、コンテナ、サーバーレス関数など) を自動的にスキャンします。

3. 脆弱性管理プロセスを確立する:脆弱性の特定、優先順位付け、修正のためのプロセスと手順を定義します。これには、定期的な脆弱性スキャンスケジュールの設定、リスク評価基準の確立、脆弱性の重大度に基づく修復スケジュールの定義などが含まれます。
4. パッチ管理の設定: パッチ管理サービスを使用して、オペレーティングシステムとアプリケーションの両方でコンピューティングインスタンスへのパッチ適用プロセスを自動化します。サービスを設定して、パッチが適用されていないインスタンスをスキャンし、スケジュールに従って自動的にインストールできます。この機能を導入するため、AWS Systems Manager Patch Manager を検討してください。
5. マルウェア保護の設定: 環境内の悪意のあるソフトウェアを検出するメカニズムを実装します。例えば、Amazon GuardDuty などのツールを使用して、EC2 ボリュームや EBS ボリューム内のマルウェアを分析、検出、アラートを行うことができます。また、GuardDuty は、Amazon S3 に新たにアップロードされたオブジェクトをスキャンし、潜在的なマルウェアやウイルスを検出できます。また、それらがダウンストリームプロセスに取り込まれる前に分離するアクションを実行できます。
6. CI/CD パイプラインに脆弱性スキャンを統合する: アプリケーションのデプロイに CI/CD パイプラインを使用している場合は、脆弱性スキャンツールをパイプラインに統合します。Amazon CodeGuru Security やオープンソースオプションなどのツールは、ソースコード、依存関係、アーティファクトをスキャンして、潜在的なセキュリティ上の問題を検出できます。
7. セキュリティモニタリングサービスの設定: AWS Security Hub CSPM などのセキュリティモニタリングサービスを設定して、複数のクラウドサービスのセキュリティ体制を包括的に把握します。このサービスは、さまざまなソースからセキュリティ検出結果を収集し、優先順位付けや修復を容易にするために標準化された形式で提示する必要があります。
8. ウェブアプリケーションのペネトレーションテストの実装: アプリケーションがウェブアプリケーションであり、組織に必要なスキルがあるか、外部からの支援を利用できる場合は、ウェブアプリケーションのペネトレーションテストの実装を検討して、アプリケーションの潜在的な脆弱性を特定してください。
9. Infrastructure as Code で自動化する: AWS CloudFormation などの Infrastructure as Code (IaC) ツールを使用して、前述のセキュリティサービスを含むリソースのデプロイと設定を自動化します。この手法は、複数のアカウントや環境にわたって、より一貫性があり標準化されたリソースアーキテクチャを作成するのに役立ちます。
10. モニタリングと継続的な改善: 脆弱性管理プログラムの有効性を継続的にモニタリングし、必要に応じて改善してください。セキュリティの検出結果を検討し、是正措置の有効性を評価し、それに応じてプロセスとツールを調整します。

リソース

関連ドキュメント:

- AWS Systems Manager
- AWS Lambda のセキュリティの概要
- Amazon CodeGuru
- Improved, Automated Vulnerability Management for Cloud Workloads with a New Amazon Inspector
- Automate vulnerability management and remediation in AWS using Amazon Inspector and AWS Systems Manager – Part 1

関連動画:

- サーバーレスおよびコンテナサービスを保護する
- Security best practices for the Amazon EC2 instance metadata service

SEC06-BP02 強化されたイメージからコンピューティングをプロビジョニングする

強化されたイメージからランタイム環境をデプロイすることで、ランタイム環境への意図しないアクセスの機会を減らします。コンテナイメージやアプリケーションライブラリなどのランタイム依存関係は、信頼できるレジストリからのみ取得し、その署名を検証します。独自のプライベートレジストリを作成して、ビルドおよびデプロイのプロセスで使用する、信頼できるイメージとライブラリを保存します。

期待される成果: コンピューティングリソースは、強化されたベースラインイメージからプロビジョニングされます。コンテナイメージやアプリケーションライブラリなどの外部依存関係は、信頼できるレジストリからのみ取得し、その署名を検証します。これらはプライベートレジストリに保存され、ビルドおよびデプロイのプロセスで参照できます。新たに発見された脆弱性に対応できるように、イメージと依存関係を定期的にスキャンして更新します。

一般的なアンチパターン:

- 信頼できるレジストリからイメージとライブラリを取得しているが、使用前に署名の検証や脆弱性スキャンを行っていない。

- イメージを強化しているが、新たな脆弱性がないか定期的にテストしたり、最新バージョンに更新したりしていない。
- イメージの想定されるライフサイクル中に、不要なソフトウェアパッケージをインストールしている、または削除していない。
- 本番環境のコンピューティングリソースを最新の状態に保つために、パッチの適用しか行っていない。パッチを適用するだけでは、経時的に、コンピューティングリソースが強化された標準に準拠しなくなる可能性があります。また、パッチを適用しても、セキュリティイベントの発生時に脅威アクターによってインストールされた可能性のあるマルウェアを削除できない場合があります。

このベストプラクティスを活用するメリット: イメージを強化すると、権限のないユーザーやサービスへの意図しないアクセスを許可する可能性がある、ランタイム環境で利用可能なパスの数を減らすことができます。また、万一、不正アクセスが発生した場合でも、影響の範囲を低減することができます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

システムを強化するには、まず、オペレーティングシステム、コンテナイメージ、アプリケーションライブラリを最新バージョンにします。既知の問題にはパッチを適用します。不要なアプリケーション、サービス、デバイスドライバー、デフォルトユーザー、その他の認証情報を削除し、システムを最小限に抑えます。ポートを無効にしてワークロードに必要なリソースと機能のみを備えた環境を作成するなど、その他の必要な対策を行ってください。その状態をベースラインとして、ワークロードの監視や脆弱性管理などの目的に必要なソフトウェア、エージェント、その他のプロセスをインストールします。

Center for Internet Security (CIS) や Defense Information Systems Agency (DISA) の Security Technical Implementation Guide (STIG) など、信頼できるソースが提供するガイダンスを利用することで、システム強化の負担を軽減できます。AWS または APN パートナーによって公開された Amazon マシンイメージ (AMI) から開始して、AWS EC2 Image Builder を使用し、CIS コントロールと STIG コントロールの適切な組み合わせに従って構成を自動化することをお勧めします。

CIS または DISA STIG のレコメンデーションを適用する強化済みのイメージや EC2 Image Builder レシピが用意されていますが、それらの構成によって、ご利用のソフトウェアの正常な実行が妨げられる場合があります。このような状況では、まず、強化されていないベースイメージを用意してソフトウェアをインストールし、CIS の統制を段階的に適用してその影響をテストします。ソフトウェアの実行を妨げている CIS 統制がある場合は、代わりに DISA でよりきめ細かな強化のレコメンデー

ションを実装できるかをテストしてください。正常に適用できる各種の CIS 統制とDISA STIG 構成を記録しておきましょう。これらを使用して、イメージ強化のレシピを EC2 Image Builder で適宜、定義します。

コンテナ化されたワークロードの場合、Docker の強化されたイメージは、Amazon Elastic Container Registry (ECR) パブリックリポジトリで利用できます。EC2 Image Builder を使用して、AMI と共にコンテナイメージを強化できます。

オペレーティングシステムやコンテナイメージと同様に、pip、npm、Maven、NuGet などのツールを通じて、パブリックリポジトリからコードパッケージ (またはライブラリ) を取得できます。AWS CodeArtifact 内などのプライベートリポジトリを信頼できるパブリックリポジトリと統合して、コードパッケージを管理することをお勧めします。この統合により、パッケージを取得、保存し、最新の状態に保つことができます。その後、アプリケーションビルドプロセスで、Software Composition Analysis (SCA)、Static Application Security Testing (SAST)、および Dynamic Application Security Testing (DAST) などの手法を使用して、これらのパッケージの最新バージョンをアプリケーションと一緒に取得し,テストできます。

AWS Lambda を使用するサーバーレスワークロードの場合、Lambda レイヤーを使用してパッケージの依存関係を簡単に管理できます。Lambda レイヤーを使用して、さまざまな関数間で共有される一連の標準依存関係をスタンドアロンアーカイブに構成します。独自のビルドプロセスでレイヤーを作成して管理できるため、関数を一元的に最新の状態に保つことができます。

実装手順

- オペレーティングシステムを強化する。信頼できるソースから取得したベースイメージを、強化した AMI を構築するための基盤として使用します。EC2 Image Builder を使用すると、イメージにインストールされたソフトウェアをカスタマイズできます。
- コンテナ化されたリソースを強化する。セキュリティのベストプラクティスを満たすように、コンテナ化されたリソースを設定します。コンテナを使用する場合は、ビルドパイプラインに ECR イメージスキャンを実装し、イメージリポジトリに対して定期的にコンテナ内の CVE を探します。
- AWS Lambda でサーバーレス実装を使用する場合は、Lambda レイヤーを使用して、アプリケーション関数コードと共有依存ライブラリを分離します。Lambda でコード署名を設定すると、信頼できるコードのみを Lambda 関数で実行することができます。

リソース

関連するベストプラクティス:

- OPS05-BP05 パッチ管理を実行する

関連動画:

- Deep dive into AWS Lambda security

関連する例:

- Quickly build STIG-compliant AMI using EC2 Image Builder
- Building better container images
- Using Lambda layers to simplify your development process
- Develop & Deploy AWS Lambda Layers using Serverless Framework
- オープンソースの SCA、SAST、DAST ツールを使用してエンドツーエンドの AWS DevSecOps CI/CD パイプラインを構築する

SEC06-BP03 手動管理とインタラクティブアクセスを削減する

デプロイ、構成、保守、調査のタスクは、可能な限り自動化します。自動化を利用できない場合は、緊急対応が必要な事態や安全な (サンドボックス) 環境でのコンピューティングリソースへの手動アクセスを検討してください。

期待される成果: プログラムスクリプトと自動化ドキュメント (ランブック) は、コンピューティングリソースに対する承認されたアクションをキャプチャします。これらのランブックは、変更検出システムによって自動的に開始されるか、人間による判断が必要な場合は手動で開始されます。コンピューティングリソースへの直接アクセスは、自動化を利用できない緊急時にのみ可能です。手動のアクティビティはすべてログに記録され、自動化機能を継続的に改善していくため、レビュープロセスに組み込まれます。

一般的なアンチパターン:

- SSH や RDP などのプロトコルを使用して、Amazon EC2 インスタンスにインタラクティブにアクセスする。
- `/etc/passwd` や Windows ローカルユーザーなどの個々のユーザーログインを維持する。
- インスタンスにアクセスするためのパスワードまたはプライベートキーを複数のユーザー間で共有している。
- 手動でソフトウェアをインストールし、構成ファイルを作成または更新している。

- 手動でソフトウェアを更新するかパッチを適用する。
- インスタンスにログインして問題をトラブルシューティングする。

このベストプラクティスを活用するメリット: 自動化を使用してアクションを実行すると、意図しない変更や設定ミスの運用リスクを軽減できます。インタラクティブアクセスに Secure Shell (SSH) や Remote Desktop Protocol (RDP) を使用しなくなれば、コンピューティングリソースへのアクセスの範囲が限定されます。これにより、不正行為に対して一般的な経路を遮断できます。コンピューティングリソースの管理タスクを自動化ドキュメントやプログラムスクリプトに定義しておくことで、承認されるアクティビティの全範囲をきめ細かく定義および監査するメカニズムを確立できます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

インスタンスへのログインは、システム管理における従来のアプローチです。サーバーのオペレーティングシステムをインストールした後、ユーザーは通常手動でログインしてシステムを設定し、必要なソフトウェアをインストールします。サーバーの存続期間中、ユーザーはログインしてソフトウェアの更新、パッチの適用、構成の変更、問題のトラブルシューティングを行うことができます。

ただし、手動アクセスは多くのリスクを伴います。サーバーは SSH や RDP サービスなどのリクエストをリスニング (待ち受け) しなければならず、それが不正アクセスに対して経路を開くことになりかねません。また、手作業による人為的ミスのリスクも高まります。その結果、ワークロードインシデント、データの破損や破壊、その他のセキュリティ問題が発生するおそれがあります。また、人為的アクセスには認証情報の共有に対する保護も必須となり、管理オーバーヘッドが増えます。

これらのリスクを軽減するために、AWS Systems Manager などのエージェントベースのリモートアクセスソリューションを実装できます。AWS Systems Managerエージェント (SSM エージェント) は、暗号化されたチャネルを自ら確立するため、外部発信のリクエストをリッスンする必要がありません。VPC エンドポイントを介してこのチャネルを確立するように、SSM エージェントを設定することを検討してください。

Systems Manager では、マネージドインスタンスとの対話方法をきめ細かく制御できます。実行する自動化、実行できるユーザー、実行できるタイミングを定義します。Systems Manager は、インスタンスへのインタラクティブなアクセスなしで、パッチの適用、ソフトウェアのインストール、および設定変更を行うことができます。Systems Manager は、リモートシェルへのアクセスを提供し、セッション中に呼び出されたすべてのコマンドとその出力を、ログと Amazon S3に記録することもできます。AWS CloudTrail は、検査のために Systems Manager API の呼び出しを記録します。

実装手順

1. Amazon EC2 インスタンスに、AWS Systems Manager Agent (SSM エージェント) をインストールします。SSM Agent が基本の AMI 構成に組み込まれていて、自動的に起動されるかどうかを確認してください。
2. EC2 インスタンスプロファイルに関連付けられた IAM ロールに、`AmazonSSMManagedInstanceCore` マネージド IAM ポリシーが含まれていることを確認します。
3. インスタンスで実行されている SSH、RDP、その他のリモートアクセスサービスを無効にします。このためには、起動テンプレートのユーザーデータセクションで設定したスクリプトを実行するか、EC2 Image Builder などのツールを使用してカスタムの AMI を構築します。
4. EC2 インスタンスに適用されるセキュリティグループの受信 (イングレス) ルールで、ポート 22/tcp (SSH) またはポート 3389/tcp (RDP) へのアクセスが許可されていないことを確認します。AWS Config などのサービスを使用して、セキュリティグループの構成ミスに対する検出とアラートを実装します。
5. Systems Manager で適切な自動化、ランブックを定義し、コマンドを実行します。IAM ポリシーを使用して、これらのアクションを実行できるユーザーと、アクションが許可される条件を定義します。これらの自動化を本稼働環境以外で徹底的にテストしてください。インスタンスにインタラクティブにアクセスする代わりに、必要に応じてこれらの自動化を呼び出します。
6. AWS Systems Manager Session Manager を使用し、必要に応じてインスタンスへのインタラクティブなアクセスを提供します。セッションアクティビティのログ記録を有効にして、Amazon CloudWatch Logs または Amazon S3 で監査証跡を維持します。

リソース

関連するベストプラクティス:

- REL08-BP04 イミュータブルなインフラストラクチャを使用してデプロイする

関連する例:

- Replacing SSH access to reduce management and security overhead with AWS Systems Manager

関連ツール:

- AWS Systems Manager

関連動画:

- Controlling User Session Access to Instances in AWS Systems Manager Session Manager

SEC06-BP04 ソフトウェアの整合性を検証する

暗号化技術による検証を使用して、ワークロードが使用するソフトウェアアーティファクト (イメージを含む) の整合性を検証します。 コンピューティング環境内で行われる不正変更への対策として、暗号化技術によりソフトウェアに署名します。

期待される成果: すべてのアーティファクトが信頼できるソースから取得されます。ベンダーのウェブサイトの証明書が検証されます。 ダウンロードしたアーティファクトは、署名により暗号化技術を使用して検証されます。独自のソフトウェアは暗号化技術を使用して署名され、コンピューティング環境によって検証されます。

一般的なアンチパターン:

- 定評あるベンダーのウェブサイトを信頼してソフトウェアアーティファクトを入手しているが、証明書の有効期限の通知を無視している。 証明書が有効であることを確認せずにダウンロードを続行する。
- ベンダーウェブサイトの証明書を検証するが、これらのウェブサイトからダウンロードしたアーティファクトについては、暗号化技術による検証を行わない。
- ソフトウェアの整合性の検証をダイジェストまたはハッシュのみに頼っている。 ハッシュでは、アーティファクトが元のバージョンから変更されていないことは確認できますが、ソースは検証されません。
- 独自のソフトウェア、コード、またはライブラリに署名しない (独自のデプロイでしか使用しない場でも、署名は必要です)。

このベストプラクティスを活用するメリット: ワークロードが依存するアーティファクトの整合性を検証すると、マルウェアがコンピューティング環境に侵入するのを防ぐことができます。 ソフトウェアに署名することで、コンピューティング環境での不正実行を防ぐことができます。 コードに署名して検証することで、ソフトウェアサプライチェーンが保護されます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

オペレーティングシステムイメージ、コンテナイメージ、コードアーティファクトは、多くの場合、ダイジェストやハッシュなどによる整合性チェックが可能な状態で配布されます。 その場合、クライアントはペイロードのハッシュを独自に計算し、それが公開されたものと同じであることを検証することで、整合性を検証できます。 これらのチェックはペイロードが改ざんされていないことを確認するのに役立ちますが、ペイロードが元のソース (データの来歴) から送信されたかどうかは検証しません。 データの来歴を確認するには、信頼できる機関がアーティファクトにデジタル署名するために発行した証明書が必要です。

ダウンロードしたソフトウェアまたはアーティファクトをワークロードで使用している場合は、プロバイダーがデジタル署名検証用のパブリックキーを提供しているかどうかを確認してください。 AWS では、公開しているソフトウェアのパブリックキーと検証手順を提供しています。以下に例を示します。

- EC2 Image Builder: AWS TOE インストールダウンロードの署名を検証します
- AWS Systems Manager: SSM エージェントの署名を検証します
- Amazon CloudWatch: CloudWatch エージェントパッケージの署名を検証します

「SEC06-BP02 強化されたイメージからコンピューティングをプロビジョニングする」で説明しているように、イメージの取得と強化に使用するプロセスにデジタル署名検証を組み込みます。

AWS Signer を使用して、署名の検証と、独自のソフトウェアとアーティファクトの独自のコード署名ライフサイクルを管理できます。 AWS Lambda と Amazon Elastic Container Registry はどちらも Signer との統合が可能で、コードとイメージの署名を検証します。 「リソース」セクションの例を参考にして、継続的インテグレーションと継続的デリバリー (CI/CD) パイプラインに Signer を組み込み、署名の検証と独自のコードやイメージへの署名を自動化できます。

リソース

関連ドキュメント:

- Cryptographic Signing for Containers
- Best Practices to help secure your container image build pipeline by using AWS Signer
- Announcing Container Image Signing with AWS Signer and Amazon EKS
- AWS Lambda でのコード署名の設定
- Best practices and advanced patterns for Lambda code signing

- Code signing using AWS Certificate Manager Private CA and AWS Key Management Service asymmetric keys

関連する例:

- Automate Lambda code signing with Amazon CodeCatalyst and AWS Signer
- Signing and Validating OCI Artifacts with AWS Signer

関連ツール:

- AWS Lambda
- AWS Signer
- AWS Certificate Manager
- AWS Key Management Service
- AWS CodeArtifact

SEC06-BP05 コンピューティング保護を自動化する

コンピューティング保護操作を自動化して、人的介入の必要性を減らします。自動スキャンを使用してコンピューティングリソース内の潜在的な問題を検知し、プログラムによる自動応答またはフリート管理操作で修正します。 CI/CD プロセスに自動化を組み込むことで、最新の依存関係を反映した信頼できるワークロードをデプロイできます。

期待される成果: 自動化システムは、コンピューティングリソースのすべてのスキャンとパッチ適用を実行します。自動検証を使用して、ソフトウェアイメージと依存関係が信頼できるソースから取得され、改ざんされていないことを確認します。ワークロードは自動的に最新の依存関係をチェックし、AWS コンピューティング環境での信頼度を確立するために署名されます。 非準拠リソースが検出されると、自動修復が開始します。

一般的なアンチパターン:

- イミュータブルインフラストラクチャの慣習に従っているが、本稼働システムの緊急時のパッチ適用や交換に備えたソリューションが不在である。
- 誤った構成のリソースを自動修正しているが、手動によるオーバーライドメカニズムが導入されていない。 要件の調整が必要となる事態が発生し、そうした変更を行うまで自動化を中断しなければならない場合が考えられます。

このベストプラクティスを活用するメリット: 自動化は、コンピューティングリソースへの不正アクセスと不正使用のリスクを軽減します。 本番環境に構成ミスが波及しないよう防ぎ、構成ミスが生じた場合は検知して修正できます。 コンピューティングリソースへの不正アクセスや不正使用を検知し、対応にかかる時間を短縮するうえでも、自動化が役に立ちます。 その結果、問題による全体的な影響範囲を縮小できます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

コンピューティングリソースを保護するために、セキュリティの柱のプラクティスで説明されている自動化を適用できます。「SEC06-BP01 脆弱性管理を実行する」では、CI/CD パイプラインの両方で Amazon Inspector を使用し、ランタイム環境を継続的にスキャンして既知の共通脆弱性識別子 (CVE) を見つける方法を説明しています。 AWS Systems Manager を使用してパッチを適用したり、自動ランブックを通じて新しいイメージから再デプロイしたりして、コンピューティングフリートを最新のソフトウェアやライブラリで更新できます。 これらの手法を使用すれば、手作業によるプロセスやコンピューティングリソースへのインタラクティブアクセスの必要性を低減できます。 詳細については、「SEC06-BP03 手動管理とインタラクティブアクセスを削減する」を参照してください。

自動化は、「SEC06-BP02 強化されたイメージからコンピューティングをプロビジョニングする」と「SEC06-BP04 ソフトウェアの整合性を検証する」で説明している、信頼できるワークロードのデプロイでも役割を果たします。 EC2 Image Builder、AWS Signer、AWS CodeArtifact、Amazon Elastic Container Registry (ECR) などのサービスを使用することで、強化され、承認されたイメージとコードの依存関係をダウンロード、検証、コンストラクト、保存できます。 Inspector と同様、これらのサービスがそれぞれ CI/CD プロセスで役割を果たし、依存関係が最新であり、出所が信頼できるソースであることが確認された場合にのみ、ワークロードが本番環境に投入されるようにします。 ワークロードも署名されるため、AWS Lambda や Amazon Elastic Kubernetes Service (EKS) などの AWS コンピューティング環境は、実行を許可する前に改ざんされていないことを確認できます。

このような予防的統制のほかに、コンピューティングリソースの発見的統制でも自動化を活用できます。 一例として、AWS Security Hub CSPM は、[EC2.8] EC2 インスタンスはインスタンスメタデータサービスバージョン 2 (IMDSv2) を使用する必要があるか、などのチェックを含む、NIST 800-53 Rev.5標準を提供しています IMDSv2 はセッション認証の手法を使用して、X-Forwarded-For HTTP ヘッダーを含むリクエストをブロックし、ネットワーク TTL を 1 に設定して、EC2 インスタンスに関する情報を取得する外部ソース発信のトラフィックを停止します。Security Hub CSPM のこのチェックでは、EC2 インスタンスが IMDSv1 を使用しているタイミングを検出し、自動修復を開始

できます。自動検出と修復の詳細については、「SEC04-BP04 非準拠リソースの修復を開始する」を参照してください。

実装手順

1. EC2 Image Builder を使用して、安全で規制に準拠し、強化された AMI の作成を自動化します。 基本の AWS イメージや APN パートナーイメージから、Center for Internet Security (CIS) ベンチマークまたは Security Technical Implementation Guide (STIG) 標準の統制を組み込んだイメージを作成できます。
2. 構成管理を自動化します。構成管理のサービスやツールを使うことで、コンピューティングリソースで安全性の高い構成を自動的に適用および検証します。
   a. AWS Config を使用した自動構成管理
   b. AWS Security Hub CSPM を使用したセキュリティとコンプライアンス体制の自動管理
3. Amazon Elastic Compute Cloud (Amazon EC2) インスタンスのパッチ適用または置き換えを自動化する。AWSSystems Manager Patch Manager は、セキュリティ関連および他の種類の更新の両方を使用して、マネージドインスタンスにパッチを適用するプロセスを自動化します。Patch Manager を使用して、オペレーティングシステムとアプリケーションの両方にパッチを適用することができます。
   a. AWS Systems Manager Patch Manager
4. 共通脆弱性識別子 (CVE) を検知するためのコンピューティングリソースのスキャンを自動化し、セキュリティスキャンソリューションをビルドパイプラインに埋め込みます。
   a. Amazon Inspector –
   b. ECR イメージスキャン
5. コンピューティングリソースを保護するために、マルウェアと脅威を自動検出する Amazon GuardDuty を検討してください。GuardDuty は、AWS 環境で AWS Lambda 関数が呼び出されたときに発生する可能性のある問題を特定することもできます。
   a. Amazon GuardDuty
6. AWS パートナーソリューションを検討する。AWSパートナーは、オンプレミス環境の既存の統制に匹敵するか同等の、またはそれらと統合できる、業界をリードする製品を提供しています。これらの製品で AWS の既存のサービスを補完して、包括的なセキュリティアーキテクチャをデプロイし、クラウド環境とオンプレミス環境の全体でよりシームレスなエクスペリエンスを実現できます。
   a. インフラストラクチャセキュリティ

リソース

関連するベストプラクティス:

- SEC01-BP06 標準的なセキュリティ統制のデプロイを自動化する

関連ドキュメント:

- Get the full benefits of IMDSv2 and disable IMDSv1 across your AWS infrastructure

関連動画:

- Security best practices for the Amazon EC2 instance metadata service

データ保護

ワークロードを設計する前に、セキュリティに影響を与える基本的なプラクティスを用意しておく必要があります。例えば、データ分類は機密性のレベルに基づいてデータを分類する方法を提供し、暗号化では、不正なアクセスに対し、データを判読できなくすることでデータを保護します。これらの方法は、誤操作の防止や規制義務の遵守などの目的に役立つため、重要です。

AWS にはデータ保護対策として使用できるさまざまな方法があります。以下のセクションでは、こうしたアプローチの使用方法を説明します。

トピック

データ分類

データ分類方法を確立すると、重要度と機密性に基づいて組織データをカテゴリ別に分類して、各カテゴリに適した保護と保持方法でデータを管理できるようになります。

ベストプラクティス

SEC07-BP01 データ分類スキームを理解する

ワークロードが処理するデータの分類、その取り扱い要件、関連するビジネスプロセス、データの保存場所、データの所有者について理解します。 データの分類および取り扱いのスキームでは、ワークロードに適用される法的要件とコンプライアンス要件、必要なデータ統制を考慮する必要があります。データを理解することが、データ分類作業の第一歩です。

期待される成果: ワークロードに存在するデータの種類が十分に理解され、文書化されます。 適切な統制が効き、機密データがその分類に基づいて保護されます。 こうした統制では、データへのアク

セス許可を持つユーザーとアクセスの目的、データの保存場所、データの暗号化ポリシーと暗号化キーの管理方法、データのライフサイクルとその保持要件、適切な破棄プロセス、実施されているバックアップと復旧のプロセス、アクセスの監査などの考慮事項が規定されます。

一般的なアンチパターン:

- データの機密レベルと取り扱い要件を定義する正式なデータ分類ポリシーが導入されていない。
- ワークロード内のデータの機密レベルが十分に理解されておらず、その情報がアーキテクチャや運用のドキュメントに記録されていない。
- データに対し、その機密性と要件に基づいた適切な統制を、データの分類と取り扱いに関するポリシーに従って適用できていない。
- データの分類と取り扱い要件に関するフィードバックを、ポリシーの所有者と共有できていない。

このベストプラクティスを活用するメリット: このプラクティスは、ワークロード内のデータの適切な処理に関するあいまいさを排除します。 組織内のデータの機密レベルと各レベルで求められる保護を定義した正式なポリシーを適用することで、法的規制やサイバーセキュリティに関する証明書、認証に準拠しやすくなります。 ワークロードの所有者は、機密データがどこに保存されているか、どのような保護統制が実施されているかを確実に把握できます。 こうした情報を文書に記録することで、新しいチームメンバーが職務の早い段階でそれらを理解し、統制を維持できるようになります。これらを実践すると、データの種類ごとに統制の適切なサイジングを行い、コストも削減できます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

ワークロードを設計する際、機密データを保護する方法を直感的に考慮しているかもしれません。 例えば、マルチテナントアプリケーションの場合、各テナントのデータを機密とみなし、任意のテナントが別のテナントのデータにはアクセスできないように保護するのは、直感的な対応です。 同様に、データを変更できるのは管理者だけで、他のユーザーは読み取り専用のアクセス権しかないか、一切アクセス権がないというように、直感的にアクセス制御を設計する場合もあるでしょう。

こうしたデータ機密レベルをデータ保護要件と併せて定義し、ポリシーに取り込むことで、ワークロードに含まれるデータを正式に特定できます。その上で、適切な統制が実施されているか、統制を監査できるか、データの取り扱いミスが判明した場合はどのような対応が適切かを判断できます。

機密データがワークロード内のどこに存在するかを特定するには、データカタログの使用を検討してください。データカタログは、組織内のデータ、場所、機密性レベル、データを保護するための⊐

ントロールをマッピングするデータベースです。また、利用可能な場合はリソースタグの使用を検討してください。 例えば、保護されたヘルス情報 (PHI) PHI のタグキー `Classification` とタグ値を持つタグ、およびタグキー `Sensitivity` とタグ値 `High` を持つ別のタグを適用できます。 その後、AWS Config などのサービスを使用して、これらのリソースの変更をモニタリングし、保護要件に準拠していない方法で変更された場合 (暗号化設定を変更するなど) に警告できます。 AWS Organizations の機能であるタグポリシーを使用して、タグキーと許容値の標準定義をキャプチャできます。タグのキーや値にプライベートデータや機密データを含めることはお勧めできません。

実装手順

1. 組織のデータ分類スキームと保護要件を理解します。
2. ワークロードによって処理される機密データの種類を特定します。
3. データカタログにデータを保存します。これにより、データが存在する場所とそのデータの機密性を 1 つのビューで確認できます。
4. 可能であれば、リソースレベルとデータレベルのタグ付けを行い、監視やインシデント対応に役立つ可能性のある機密レベルやその他の運用メタデータをデータにタグ付けすることを検討してください。
   a. AWS Organizations タグポリシーを使用して、タグ付けの標準を適用できます。

リソース

関連するベストプラクティス:

- SUS04-BP01 データ分類ポリシーを実装する

関連ドキュメント:

- データ分類に関するホワイトペーパー
- AWS リソースのタグ付けのベストプラクティス

関連する例:

- AWS Organizations タグポリシーの構文と例

関連ツール

- AWS タグエディタ

SEC07-BP02 データの機密性に基づいてデータ保護統制を適用する

データ保護統制を適用し、分類ポリシーで定義されている各クラスのデータに適切なレベルの統制を行います。 これにより、データの可用性と使用を確保しながら、機密データを不正アクセスや不正使用から守ることができます。

期待される成果: 組織内のデータのさまざまな機密性レベルを定義する分類ポリシーがあります。 機密レベルごとに、承認された保管や取り扱いのサービスと場所、それらに必要な構成に関する明確なガイドラインが公開されています。 必要とされる保護のレベルと付随するコストに応じて、レベルごとに統制を実施します。 データが許可されていない場所に存在する場合、許可されていない環境で処理されている場合、権限のないアクターによってアクセスされている場合、または関連サービスの構成がコンプライアンス違反になった場合に検知し、警告する監視体制が整っています。

一般的なアンチパターン:

- すべてのデータに同じレベルの保護統制が適用されている。機密性の低いデータに対してセキュリティ統制が行き過ぎたり、機密性の高いデータの保護が不十分になったりするおそれがあります。
- データ保護統制を定義する際に、セキュリティチーム、コンプライアンスチーム、ビジネスチームの関係者が関与していない。
- データ保護統制の導入と維持に関連する運用上のオーバーヘッドとコストを見落としている。
- 分類ポリシーとの整合性を維持するための、データ保護統制の定期的なレビューを実施していない。
- 保存中および転送中のデータレジデンシーの完全なインベントリがない。

このベストプラクティスを活用するメリット: コントロールをデータの分類レベルに合わせることで、必要に応じてより高いレベルのコントロールに投資できます。例えば、保護、監視、測定、修復、報告に関するリソースの増強が該当します。 統制を緩めた方が適切な場面では、従業員、顧客、または関係者にとってデータのアクセシビリティと完全性を向上させることができます。 このアプローチにより、組織はデータ保護要件を順守しながら、データを最大限柔軟に活用できるようになります。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

データの機密レベルに基づいてデータ保護統制を実施するには、いくつかの重要なステップが必要です。まず、ワークロードアーキテクチャ内のさまざまなデータ機密レベル (公開、内部、機密、制限など) を特定し、該当データを保存して処理する場所を評価します。次に、機密レベルに基づいて

データの周囲の分離境界を定義します。サービスコントロールポリシー (SCP) を使用して、データ機密性レベルごとに許可されるサービスとアクションを制限し、データを異なる AWS アカウントに分けることをお勧めします。これにより、強固な分離境界を確立し、最小特権の原則を適用できます。

分離境界を定義したら、データの機密レベルに基づいて適切な保護統制を実装します。暗号化、アクセスコントロール、監査などの関連するコントロールを実装するには、保管中のデータを保護するベストプラクティスと転送中のデータを保護するベストプラクティスを参照してください。データの機密レベルを下げるには、トークン化や匿名化などの手法を検討してください。トークン化とトークン化解除の一元化システムにより、ビジネス全体に一貫したデータポリシーを簡単に適用できます。

実装した統制の有効性を継続的に監視し、テストします。組織のデータ環境や脅威が進化するにつれて、データ分類スキーム、リスク評価、保護統制を定期的に見直し、更新してください。実装されているデータ保護統制を、関連する業界規制、基準、法的要件に適応させてください。さらに、従業員がデータ分類スキームと、機密データの取り扱いと保護における各自の責任を把握できるように、セキュリティについて周知徹底し、トレーニングを実施してください。

実装手順

1. ワークロード内のデータの分類と機密レベルを特定します。
2. 各レベルの分離境界を定義し、実施戦略を決定します。
3. データ分類ポリシーで必要とされるアクセス、暗号化、監査、保持などについて定義した統制を評価します。
4. 必要に応じて、トークン化や匿名化の使用など、データの機密レベルを下げるオプションを評価してください。
5. 構成されたリソースのテストと監視を自動化し、統制を検証します。

リソース

関連するベストプラクティス:

- PERF03-BP01 データアクセスとストレージ要件に最適な専用データストアを使用する
- COST04-BP05 データ保持ポリシーを適用する

関連ドキュメント:

- データ分類に関するホワイトペーパー

- セキュリティ、アイデンティティ、コンプライアンスに関するベストプラクティス
- AWS KMS のベストプラクティス
- Encryption best practices and features for AWS services

関連する例:

- Building a serverless tokenization solution to mask sensitive data
- How to use tokenization to improve data security and reduce audit scope

関連ツール:

- AWS Key Management Service (AWS KMS)
- AWS CloudHSM
- AWS Organizations

SEC07-BP03 識別および分類を自動化する

データの識別と分類を自動化すると、適切な統制を実装するのに役立ちます。手動での判断を自動化によって補強することで、人為的ミスやエクスポージャのリスクが軽減されます。

期待される成果: 分類と処理ポリシーに基づいて、適切なコントロールが設定されているかどうかを確認できます。自動化されたツールとサービスは、データの機密レベルを特定して分類するのに役立ちます。 また、自動化によって環境を継続的に監視して、データが不正な方法で保存または処理されているかどうかを検出して警告できるため、是正措置を迅速に講じることができます。

一般的なアンチパターン:

- データの識別と分類を手動プロセスでしか行っていないため、ミスが起こりやすく、時間もかかる。 特にデータ量が増えてくると、データ分類が非効率になり、一貫性を欠くことにつながりかねません。
- 組織全体のデータ資産を追跡および管理するメカニズムがない。
- 組織内でデータが移動したり進化したりする過程で、データを継続的に監視および分類する必要性を見落としている。

このベストプラクティスを活用するメリット: データ識別と分類を自動化すると、より一貫性のある正確なデータ保護コントロールの適用が可能になり、人為的ミスのリスクが軽減されます。 自動化

により、機密データへのアクセスと移動を可視化できるため、不正処理を検出して是正措置を講じることができます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

ワークロードの初期設計段階では、人間の判断でデータが分類されることが少なくありませんが、予防的統制として、テストデータの識別と分類を自動化するシステムの導入を検討してください。例えば、代表的なデータをスキャンして機密性を判断するためのツールやサービスを開発者に提供できます。 AWS では、データセットを Amazon S3 にアップロードし、Amazon Macie、Amazon Comprehend、または Amazon Comprehend Medical を使用してスキャンできます。 同様に、ユニットテストや統合テストの一環としてデータをスキャンして、予期しない場所に機密データが存在していないか検知することも検討してください。この段階で機密データに関する警告を行うことで、本番環境へのデプロイ前にセキュリティ保護のギャップを浮き彫りにすることができます。AWS Glue、Amazon SNS、Amazon CloudWatch での機密データ検出などの機能を使用して、PII を検出し、修正アクションを実行することもできます。どの自動化ツールやサービスでも、機密データがどのように定義されているかを理解し、他の人間によるソリューションや自動化されたソリューションで適宜補強することで、ギャップを埋めることができます。

発見的統制として、環境を継続的に監視し、機密データがコンプライアンスに違反する方法で保存されていないかを検出してください。 これにより、機密データが適切な匿名化や伏字化を行わないままログファイルに出力されたり、データ分析環境にコピーされたりする事態を検知できます。 Amazon S3 に保存されているデータは、Amazon Macie を使用して、機密データがないか継続的に監視できます。

実装手順

1. SEC07-BP01 で説明されている組織内のデータ分類スキームを確認します。
    a. 組織のデータ分類スキームを理解することにより、会社のポリシーに沿った自動識別と分類のための正確なプロセスを確立できます。
2. 環境の初期スキャンを実行して、自動で識別と分類を行います。
    a. データを最初にフルスキャンすることで、環境内のどこに機密データが存在するかを包括的に把握できます。フルスキャンが初期段階では不要な場合や、コスト面で事前に完了するのが難しい場合は、データサンプリング手法で成果が得られるか評価してください。例えば、S3 バケット全体で広範にわたって機密データを自動検出するように Amazon Macie を設定できます。 この機能では、サンプリング手法を使用して、機密データの所在の事前分析をコスト効率

よく実行します。 その後、機密データの検出ジョブを用いて、S3 バケットの詳細な分析を実行できます。他のデータストアも S3 にエクスポートして Macie でスキャンできます。

b. スキャン内で識別されたデータストレージリソースに対して、SEC07-BP02 で定義されたアクセスコントロールを確立します。

3. 環境の継続的なスキャンを設定します。

a. Macie の自動機密データ検出機能を使用して、環境を継続的にスキャンできます。 機密データの保存が許可されている既知の S3 バケットは、Macie の許可リストを使用して除外できます。

4. 識別と分類をビルドとテストのプロセスに組み込みます。

a. ワークロードの開発中に、開発者がデータの機密性をスキャンするために使用できるツールを特定します。 これらのツールを統合テストの一環として使用して、予期しない場所に機密データが存在する場合に警告し、その後のデプロイを防ぎます。

5. 許可されていない場所で機密データが見つかったときに対処するためのシステムまたはランブックを実装します。

a. 自動修復を使用して、データへのアクセスを制限します。例えば、このデータをアクセスが制限された S3 バケットに移動する場合や、属性ベースのアクセス制御 (ABAC) を使用している場合は、オブジェクトにタグ付けできます。さらに、データが検出された場合は、そのデータをマスキングすることを検討してください。

b. データ保護チームとインシデント対応チームに、インシデントの根本原因を調査するようにアラートを出します。チームで把握した事例は、将来のインシデントを防ぐのに役立ちます。

リソース

関連ドキュメント:

- AWS Glue: Detect and process sensitive data
- Using managed data identifiers in Amazon SNS
- Amazon CloudWatch Logs: 機密性の高いログデータをマスキングで保護する

関連する例:

- Enabling data classification for Amazon RDS database with Macie
- Detecting sensitive data in DynamoDB with Macie

関連ツール:

- Amazon Macie
- Amazon Comprehend
- Amazon Comprehend Medical
- AWS Glue

SEC07-BP04 スケーラブルなデータのライフサイクル管理を定義する

データのライフサイクルの要件を、関連するさまざまなレベルのデータ分類や取り扱いに応じて把握してください。 例えば、データを初めて環境に取り込んだときの取り扱い方法や、データの変換方法、破棄のルールなどが該当します。保存期間、アクセス、監査、出所追跡などの要素を考慮してください。

期待される成果: 取り込みに近いポイントおよび時点でデータを分類します。データの分類にマスキングやトークン化、機密情報を保護するその他の対策が必要な場合は、そうした作業をできるだけ取り込みポイントおよび時点に近いポイントおよび時点で行います。

保管しておくことが適切でなくなったデータは、その分類に基づいて、ポリシーに従って削除します。

一般的なアンチパターン:

- データのライフサイクル管理に画一的なアプローチを実装し、さまざまな機密度やアクセス要件が考慮されていない。
- 利用可能なデータとバックアップされているデータの両方ではなく、いずれか一方の視点でのみライフサイクル管理を検討している。
- データがその価値や出所を確認することなく、ワークロードに入力された時点で有効だと仮定されている。
- データのバックアップや保護を行う代わりに、データの耐久性に頼り切っている。
- データが有用でなくなり、必要な保持期間が過ぎても保持し続けている。

このベストプラクティスを活用するメリット: 明確に定義されスケーラブルなデータライフサイクル管理戦略は、適切なコントロールを維持しながら、規制コンプライアンスの維持、データセキュリティの向上、ストレージコストの最適化、効率的なデータアクセスと共有を可能にします。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

ワークロード内のデータは多くの場合、動的です。 ワークロード環境に入ってくるときの形式は、ビジネスロジック、レポート、分析、機械学習で保存または使用されるときの形式とは異なる場合があります。 さらに、データの価値は時間とともに変化する可能性があります。一部のデータは時間に依存する性質があり、古くなるにつれて価値を失います。 データのこうした変更が、データ分類スキームや関連する統制の下での評価にどのように影響するかを検討してください。 可能な場合は、Amazon S3 ライフサイクルポリシーや Amazon Data Lifecycle Manager などの自動ライフサイクルメカニズムを使用して、データ保持、アーカイブ、有効期限のプロセスを設定します。DynamoDB に保存されたデータの場合、Time to Live (TTL) 機能を使用して、項目ごとの有効期限のタイムスタンプを定義できます。

使用可能なデータと、バックアップとして保存されているデータは区別します。 AWS Backup を使用して、AWS サービス間のデータのバックアップを自動化することを検討してください。 Amazon EBS スナップショットは、EBS ボリュームをコピーし、ライフサイクル、データ保護、保護メカニズムへのアクセスなどの S3 機能を使用して保存する方法を提供します。これらの機能のうち 2 つは S3 Object Lock と AWS Backup ボールトロックです。これにより、バックアップのセキュリティと制御を強化できます。バックアップに関して、職務とアクセス権を明確に分離して管理します。バックアップはアカウントレベルで分離し、イベントの発生時に影響を受ける環境から分離した状態を維持できるようにします。

ライフサイクル管理のもう 1 つの要素は、データ出所追跡と呼ばれるワークロードの進行状況に応じたデータの履歴を記録することです。これにより、データがどこから来たのか、変換されている場合はどのような変換か、どの所有者やプロセスがいつそれらの変更を行ったのかを確実に把握できます。 こうした履歴は、潜在的なセキュリティイベントが発生した際の問題のトラブルシューティングや調査に役立ちます。 例えば、Amazon DynamoDB テーブルの変換に関するメタデータをログに記録できます。 データレイク内では、変換後のデータのコピーをデータパイプラインのステージごとに異なる S3 バケットに保存できます。スキーマとタイムスタンプ情報を AWS Glue Data Catalog に保存します。 どのソリューションを使用する場合でも、エンドユーザーの要件を考慮して、データの出所に関するレポートに必要な適切なツールを判断してください。 そうすることで、出所を最も適切に追跡する方法を決定できます。

実装手順

1. ワークロードのデータタイプ、機密レベル、アクセス要件を分析してデータを分類し、適切なライフサイクル管理戦略を定義します。
2. 法律、規制、組織の要件に沿ったデータ保持ポリシーと自動破棄プロセスを設計し、実装します。

3. ワークロードの要件や規制の変化に応じて、データライフサイクル管理の戦略、統制、ポリシーを継続的に監視、監査、調整するためのプロセスと自動化を確立します。
   a. AWS Config で自動ライフサイクル管理が有効になっていないリソースを検出する

リソース

関連するベストプラクティス:

- COST04-BP05 データ保持ポリシーを適用する
- SUS04-BP03 ポリシーを使用してデータセットのライフサイクルを管理する

関連ドキュメント:

- データ分類に関するホワイトペーパー
- AWS Blueprint for Ransomware Defense
- DevOps Guidance: Improve traceability with data provenance tracking

関連する例:

- How to protect sensitive data for its entire lifecycle in AWS
- Build data lineage for data lakes using AWS Glue, Amazon Neptune, and Spline

関連ツール:

- AWS Backup
- Amazon Data Lifecycle Manager
- AWS Identity and Access Management Access Analyzer

保管中のデータの保護

保管中のデータとは、ワークロードの任意の期間に永続的ストレージに保持されるすべてのデータを指します。例えば、ブロックストレージ、オブジェクトストレージ、データベース、アーカイブ、IoT デバイス、データが保持されているその他のストレージ媒体などがあります。暗号化と適切なアクセスコントロールが実装されている場合は、保管中のデータを保護することで不正アクセスのリスクを軽減できます。

暗号化とトークン化は、共に重要なデータ保護スキームですが、異なる特徴を持ちます。

トークン化とは、機密情報を表すトークン (顧客のクレジットカード番号を表すトークンなど) を定義するためのプロセスです。トークンはそれ自体に意味があってはいけません。また、トークン化中のデータから派生してはいけません。このため、暗号化ダイジェストはトークンとしては使用できません。トークン化方式を慎重に計画することで、コンテンツの保護を強化し、コンプライアンス要件を確実に満たすことができます。例えば、クレジットカード番号の代わりにトークンを利用すると、クレジットカード処理システムに関するコンプライアンスの範囲を狭めることができます。

暗号化とは、プレーンテキストに復号化するために必要な秘密鍵がないとコンテンツを読めないように変換する方法です。必要に応じてトークン化と暗号化の両方を使用して、情報の安全を確保し、保護することができます。この他に、マスキング手段を使用すると、残りのデータが機密とみなされないポイントまでデータの一部を編集できます。例えば、PCI-DSS では、カード番号の最後の 4 桁をコンプライアンススコープの境界外に保持して、インデックスを作成できます。

暗号化キーの使用を監査する: 暗号化キーの使用方法を理解したうえで、監査を実施し、キーのアクセス制御メカニズムが適切に実践されていることを検証します。例えば、AWS KMS キーを使用するすべての AWS サービスでは、毎回の使用を AWS CloudTrail に記録します。その後、Amazon CloudWatch Logs Insights などのツールを使用して AWS CloudTrail にクエリを実行し、キーの使用がすべて有効であることを確認できます。

ベストプラクティス

- SEC08-BP01 安全なキー管理を実装する
- SEC08-BP02 保管中に暗号化を適用する
- SEC08-BP03 保管中のデータの保護を自動化する
- SEC08-BP04 アクセスコントロールを適用する

SEC08-BP01 安全なキー管理を実装する

安全なキー管理には、ワークロード用に保管中のデータを保護するために必要な、キーマテリアルの保管、ローテーション、アクセス制御、監視が含まれます。

期待される成果: スケーラブルで反復可能な、自動化されたキー管理メカニズム。このメカニズムは、キーマテリアルへの最小特権アクセスを適用し、キーの可用性、機密性、整合性の間で適切なバランスを実現します。キーへのアクセスをモニタリングし、キーマテリアルのローテーションが必要な場合は、自動プロセスを使用してキーをローテーションします。人間のオペレーターがキーマテリアルにアクセスすることは許可しません。

一般的なアンチパターン:

- 暗号化されていないキーマテリアルに人間がアクセスする。
- カスタム暗号化アルゴリズムを作成する。
- キーマテリアルへのアクセス許可の範囲が広すぎる。

このベストプラクティスを活用するメリット: ワークロード用の安全なキー管理メカニズムを確立することで、不正アクセスからコンテンツを保護することができます。さらに、データの暗号化を要求する規制要件の対象となる場合があります。効果的なキー管理ソリューションがあれば、それらの規制に合わせた技術的メカニズムを提供して、キーマテリアルを保護することができます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

保管中のデータの暗号化は、基本的なセキュリティコントロールです。この制御を実装するには、ワークロードに、保管中のデータの暗号化に使用されるキーマテリアルを安全に保存および管理するメカニズムが必要です。

AWS は AWS Key Management Service (AWS KMS) を使用して AWS KMS キー用の高い耐久性と安全性を備えた冗長ストレージを提供します。多くの AWS サービスは AWS KMS と統合して、データ暗号化をサポートしています。AWS KMS は、FIPS 140-3 レベル 3 検証済みのハードウェアセキュリティモジュールを使用してキーを保護します。AWS KMS キーをプレーンテキストでエクスポートするメカニズムはありません。

マルチアカウント戦略を使用してワークロードをデプロイする場合、キーを使用するワークロードと同じアカウントに AWS KMS キーを保持することをお勧めします。この分散モデルでは、AWS KMS キーの管理責任はチームにあります。他のユースケースでは、組織は AWS KMS キーを一元管理されたアカウントに保存することもできます。この一元化された構造では、ワークロードアカウントが統合アカウントに保存されているキーにアクセスするために必要なクロスアカウントアクセスを可能にする追加のポリシーが必要ですが、単一のキーを複数の AWS アカウントで共有するユースケースにより適している可能性があります。

キーマテリアルを保管する場所にかかわらず、キーへのアクセスはキーポリシーおよび IAM ポリシーで厳重に管理する必要があります。キーポリシーは、AWS KMS キーへのアクセスを制御する主な方法です。さらに、AWS KMS キーの付与によって、ユーザーに代わってデータを暗号化および復号する AWS のサービスへのアクセスを提供できます。AWS KMS キーへのアクセスを制御するガイダンスを確認してください。

暗号化キーの使用状況を監視して、異常なアクセスパターンを検出する必要があります。AWS マネージドキーと AWS KMS に保存されているカスタマーマネージドキーを使用して実行される操作は AWS CloudTrail でログインできるため、定期的に確認する必要があります。キー破壊イベントのモニタリングには特に注意してください。キーマテリアルの偶発的または悪意のある破壊を防ぐため、キー破壊イベントによってキーマテリアルがすぐに削除されることはありません。AWS KMS の削除には待機時間が設けられており、デフォルトで 30 日間、最短で 7 日間です。管理者は削除アクションを確認し、必要に応じてリクエストをロールバックする時間を確保できます。

AWS の多くのサービスはわかりやすい方法で AWS KMS を使用します。唯一必要なのは、AWS マネージドキーとカスタマーマネージドキーのどちらを使用するかを決定することです。ワークロードでデータを暗号化または復号するために直接 AWS KMS を使用する場合、データを保護するためエンベロープ暗号化を使用する必要があります。AWS Encryption SDK は、アプリケーションにクライアント側の暗号化プリミティブを提供し、エンベロープ暗号化を実装して AWS KMS と統合することができます。

実装手順

1. キーに適したキー管理オプション (AWS 管理またはカスタマー管理) を決定します。
    a. 使いやすさを考慮して、AWS はほとんどのサービスにおいて AWS 所有キーと AWS マネージドキーを提供しています。これにより、キーマテリアルやキーポリシーを管理しなくても保管時の暗号化が可能になります。
    b. カスタマーマネージドキーを使用する場合は、俊敏性、セキュリティ、データ主権、可用性の最適なバランスを実現するデフォルトのキーストアを検討してください。他のユースケースでは、AWS CloudHSM または外部キーストアによりカスタムキーストアの使用が必要な場合もあります。
2. ワークロードに使用しているサービスのリストを確認して、AWS KMS がサービスとどのように統合されているかを理解します。例えば、EC2 インスタンスは暗号化された EBS ボリュームを使用できます。これにより、そのボリュームから作成された Amazon EBS スナップショットもカスタマーマネージドキーを使用して暗号化されていることを確認し、暗号化されていないスナップショットデータが誤って開示されるのを防ぐことができます。
    a. AWS のサービスで AWS KMS を使用する方法
    b. AWS のサービスが提供する暗号化オプションの詳細については、ユーザーガイドの「保管時の暗号化」トピックまたはサービスのデベロッパーガイドを参照してください。
3. AWS KMS の実装: AWS KMS を使用すると、キーの作成と管理が簡単になり、幅広い AWS のサービスやアプリケーションでの暗号化の使用を制御できます。
    a. 開始方法: AWS Key Management Service (AWS KMS)

   b. AWS KMS キーへのアクセスを制御するベストプラクティスを確認してください。
4. AWS Encryption SDK の検討: アプリケーションがクライアント側でデータを暗号化する必要がある場合は、AWS KMS 統合済みの AWS Encryption SDK を使用してください。
   a. AWS Encryption SDK
5. IAM Access Analyzer を有効化して、過度に広範な AWS KMS キーポリシーがないかどうかを自動的に確認し、通知を受け取るようにします。
   a. リソースポリシーの更新によって KMS キーへのパブリックアクセスが許可されないことを確認するには、カスタムポリシーチェックの使用を検討してください。
6. Security Hub CSPM を有効化して、キーポリシーの設定ミス、削除予定のキー、自動ローテーションが有効になっていないキーがある場合に通知を受け取るようにします。
7. AWS KMS キーに適した口グ記録レベルを決定します。AWS KMS への呼び出し (読み取り専用イベントを含む) は口グに記録されるため、CloudTrail に関連する AWS KMS 口グが膨大になる可能性があります。
   a. 組織によっては、AWS KMS の口グ記録アクティビティを別の証跡に分けた方がよい場合があります。詳細については、「AWS KMS デベロッパーガイド」の「CloudTrail で AWS KMS API コールを口グに記録する」セクションを参照してください。

リソース

関連ドキュメント:

- AWS Key Management Service
- AWS cryptographic services and tools
- 暗号化によるデータの保護
- エンベロープ暗号化
- Digital sovereignty pledge
- Demystifying AWS KMS key operations, bring your own key, custom key store, and ciphertext portability
- AWS Key Management Service cryptographic details

関連動画:

- How Encryption Works in AWS
- Securing Your Block Storage on AWS

- AWS data protection: Using locks, keys, signatures, and certificates

関連する例:

- での高度なアクセスコントロールメカニズムの実装AWS KMS

SEC08-BP02 保管中に暗号化を適用する

保管中のプライベートデータを暗号化して機密性を維持し、意図しないデータの開示や流出に対する保護を強化します。暗号化は、最初に復号化されない限り、データを読み取ったりアクセスしたりできないようにデータを保護します。暗号化されていないデータをインベントリして制御し、データ漏洩に関連するリスクを軽減します。

期待される成果: 保管中にデフォルトでプライベートデータを暗号化するメカニズム。これらのメカニズムはデータの機密性を維持し、意図的または不注意によるデータの開示や流出に対する保護層を追加して強化するのに役立ちます。暗号化されていないデータのインベントリを維持し、そのデータを保護するためのコントロールを理解します。

一般的なアンチパターン:

- デフォルトで暗号化する設定を使用しない。
- 複合キーに過度に寛容なアクセスを提供する。
- 暗号化および復号化キーの使用をモニタリングしない。
- データを暗号化せずに保管する。
- データの使用、タイプ、分類に関係なく、すべてのデータに同じ暗号化キーを使う。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

暗号化キーとワークロード内のデータ分類をマッピングします。このアプローチは、1 つまたは非常に少数のデータ暗号化キーを使用する場合、過度に許容されるアクセスから保護するのに役立ちます (「SEC07-BP01 データ分類スキームを理解する」を参照)。

AWS Key Management Service (AWS KMS) は、多くの AWS サービスと統合し、保管中のデータを暗号化しやすくします。例えば、Amazon Elastic Compute Cloud (Amazon EC2) では、新しい EBS ボリュームが自動的に暗号化されるように、アカウントにデフォルトの暗号化を設定できま

す。AWS KMS を使用する際は、どの程度厳格にデータを制限すべきかを検討してください。デフォルトでサービス制御型の AWS KMS キーは、AWS がユーザーに変わって管理および使用します。基盤となる暗号化キーにへのアクセスを細かく管理すべき機密データの場合、カスタマーマネージドキー (CMK) を検討してください。キーポリシーを使用することで、ローテーションやアクセス管理など、CMK を完全に制御できます。

さらに、Amazon Simple Storage Service (Amazon S3) などのサービスでは、デフォルトですべての新しいオブジェクトが暗号化されるようになりました。この実装により、パフォーマンスに影響を与えずに、セキュリティを強化できます。

Amazon Elastic Compute Cloud (Amazon EC2) や Amazon Elastic File System (Amazon EFS) などのその他のサービスは、デフォルトの暗号化の設定をサポートしています。AWS Config ルール を使用して、Amazon Elastic Block Store (Amazon EBS) ボリューム、Amazon Relational Database Service (Amazon RDS) インスタンス、Amazon S3 バケット、組織内のその他のサービスで暗号化の使用状況を自動的に確認することもできます。

AWS はまた、クライアント側の暗号化も提供するため、クラウドにアップロードする前にデータを暗号化できます。AWS Encryption SDK は、エンベロープ暗号化を使用してデータを保護する方法を提供します。ラッピングキーを提供すると、AWS Encryption SDK が暗号化する各データオブジェクトに対して固有のデータキーを生成します。マネージド単一テナントハードウェアセキュリティモジュール (HSM) が必要な場合は、AWS CloudHSM を検討します。AWS CloudHSM では、FIPS 140-2 レベル 3 検証済み HSM で暗号化キーを生成、インポート、管理できます。AWS CloudHSM のユースケースには、認証局 (CA) 発行用プライベートキーの保護、Oracle データベースに対する Transparent Database Encryption (TDE) の有効化などが挙げられます。AWS CloudHSM Client SDK は、データを AWS にアップロードする前に、AWS CloudHSM 内に保管されたキーを使って、クライアント側でデータを暗号化できるソフトウェアを提供します。Amazon DynamoDB Encryption Client では、DynamoDB テーブルにアップロードする前のアイテムを暗号化および署名することもできます。

実装手順

- 新しい Amazon EBS ボリュームのデフォルトの暗号化を設定する: 新しく作成したすべての Amazon EBS ボリュームを暗号化形式で作成することを指定します。AWS が提供するデフォルトキーを使用するか、作成したキーを使用するかを選択できます。
- 暗号化された Amazon マシンイメージ (AMI) を設定する: 暗号化を有効化して既存の AMI をコピーすると、自動的にルートボリュームとスナップショットが暗号化されます。
- Amazon RDS 暗号化を設定する: 暗号化オプションを有効化して、保管中の Amazon RDS データベースクラスターとスナップショットに対して暗号化を設定します。

- データの分類ごとに適切なプリンシパルへのアクセスを制限するポリシーが適用される AWS KMS キーを作成して設定する: 例えば、本番データを暗号化するための AWS KMS キーを 1 つ作成し、開発データまたはテストデータを暗号化するための別のキーを作成します。他の AWS アカウントに対してキーアクセスを提供することもできます。開発環境と本番環境のアカウントは別にすることを検討してください。本番環境で開発アカウントのアーティファクトを復号化する必要がある場合、開発アーティファクトを暗号化するのに使用する CMK ポリシーを編集し、本番アカウントにアーティファクトを復号化する機能を付与できます。次に、本番環境が本番で使用するために復号化されたデータをインジェストできます。
- 追加の AWS サービスで暗号化を設定する: 使用する他の AWS サービスについては、そのサービスのセキュリティドキュメントを参照して、サービスの暗号化オプションを確認してください。

リソース

関連ドキュメント:

- AWS Crypto Tools ドキュメント
- AWS Encryption SDK
- AWS KMS Cryptographic Details Whitepaper
- AWS Key Management Service
- AWS cryptographic services and tools
- Amazon EBS Encryption
- Default encryption for Amazon EBS volumes
- Amazon RDS リソースの暗号化
- Amazon S3 バケットのデフォルト暗号化を有効にする方法
- 暗号化によるデータの保護

関連動画:

- How Encryption Works in AWS
- Securing Your Block Storage on AWS

SEC08-BP03 保管中のデータの保護を自動化する

自動化を利用して、保管中のデータの統制を検証し、適用します。 自動スキャンを使用してデータストレージソリューションの設定ミスを検出し、可能な場合はプログラムによる自動対応で修復を行います。 CI/CD プロセスに自動化を組み込んで、データストレージの設定ミスを検知し、本番環境に適用されないよう未然に防ぎます。

期待される成果: 自動システムが、コントロールの設定ミス、不正アクセス、予期しない使用方法がないか、データストレージの場所をスキャンして監視します。 データストレージの設定ミスが検出されると、自動修復が開始します。 自動化されたプロセスによってデータのバックアップが作成され、イミュータブル (変更不可能) なコピーがバックアップ元の環境の外部に保管されます。

一般的なアンチパターン:

- デフォルト設定で暗号化を有効にするオプションがサポートされているのに、そうしたオプションを検討しない。
- バックアップと復旧の自動化戦略を策定する際に、運用上のイベントだけでなくセキュリティイベントも考慮していない。
- ストレージサービスに対してパブリックアクセス設定を強制しない。
- 保管中のデータを保護するための統制の監視や監査をしていない。

このベストプラクティスを活用するメリット: 自動化は、データストレージの場所の設定ミスのリスクを防ぐのに役立ちます。設定ミスが本番環境に入り込まないように阻止できます。このベストプラクティスは、設定ミスが起きた場合の検出と修正にも役立ちます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

保管中のデータを保護するためのあらゆる取り組みにおいて、自動化は重要です。「SEC01-BP06 標準的なセキュリティ統制のデプロイを自動化する」では、AWS CloudFormation などの Infrastructure as code (IaC) テンプレートを使用してリソース設定をキャプチャする方法を説明します。 これらのテンプレートはバージョン管理システムにコミットされ、CI/CD パイプラインを通じて AWS でリソースをデプロイするために使用されます。 データストレージソリューションの設定 (Amazon S3 バケットの暗号化設定など) を自動化する場合にも、これらの手法が同様に適用されます。

IaC テンプレートで定義された設定にミスがないか、AWS CloudFormation Guard のルールを CI/CD パイプライン内で使用してチェックできます。 AWS Config を使用して、CloudFormation やそ

の他の IaC ツールでまだ利用できない設定をモニタリングし、設定ミスを確認できます。「SEC04-BP04 非準拠リソースの修復を開始する」で説明されているように、設定ミスに対して Config が生成するアラートは自動的に修正できます。

アクセス許可管理の戦略に自動化を組み込むことも、自動データ保護の要素として不可欠です。「SEC03-BP02 最小特権のアクセスを付与する」および「SEC03-BP04 アクセス許可を継続的に削減する」では、最小特権アクセスポリシーの設定について説明します。AWS Identity and Access Management Access Analyzer によってポリシーは継続的に監視され、権限を削減できる場合はレポートが出力されます。 アクセス許可のモニタリングを自動化するだけでなく、Amazon GuardDuty を設定して、EBS ボリューム (EC2 インスタンスを介して)、S3 バケット、およびサポートされるAmazon Relational Database Service データベースで異常なデータアクセス動作を監視できます。

許可されていない場所に機密データが保存されていることを検知する場合にも、自動化が活躍します。「SEC07-BP03 識別および分類を自動化する」では、Amazon Macie を使用して S3 バケットで予期しない機密データをモニタリングし、自動応答を開始するアラートを生成する方法について説明します。

「REL09 バックアップデータ」のプラクティスに従って、自動データバックアップおよびリカバリ戦略を開発します。データのバックアップと復旧は、運用上のイベントと同様、セキュリティイベントから復旧するために重要です。

実装手順

1. データストレージの設定を IaC テンプレートに取り込みます。 CI/CD パイプラインで自動チェックを行い、設定ミスを検出します。
   a. IaC テンプレートには CloudFormation を、テンプレートの設定ミスをチェックするには CloudFormationGuard を使用できます。
   b. AWS Config を使用して、プロアクティブ評価モードでルールを実行します。この設定を使用して、リソースの作成前に CI/CD パイプラインのステップとしてリソースのコンプライアンスを確認します。
2. データストレージの設定ミスがないか、リソースを監視します。
   a. AWS Config を設定して、データストレージリソースの制御設定の変更をモニタリングし、設定ミスの検出時に修復アクションを呼び出すアラートを生成するようにします。
   b. 自動修復の詳細については、「SEC04-BP04 非準拠リソースの修復を開始する」を参照してください。
3. データアクセス許可を自動化により継続的に監視し、削減します。

   a. IAM Access Analyzer を継続的に実行して、アクセス許可を削減できる場合にアラートを生成できます。
4. 異常なデータアクセス動作を監視し、警告します。
   a. GuardDuty は、EBS ボリューム、S3 バケット、RDS データベースなどのデータストレージリソースについて、既知の脅威シグネチャとベースラインアクセス動作からの逸脱を監視します。
5. 機密データが予期しない場所に保存されていないか監視し、警告します。
   a. Amazon Macie を使用して、S3 バケットの機密データを継続的にスキャンします。
6. 暗号化した安全なデータバックアップの作成を自動化します。
   a. AWS Backup は、AWS の各データソースのバックアップを作成できるマネージドサービスです。 Elastic Disaster Recovery を使用すると、サーバーワークロード全体をコピーし、秒単位の目標復旧時点 (RPO) を提供する継続的なデータ保護を実現できます。 両方のサービスを連携するよう設定し、データバックアップの作成とフェイルオーバー先へのコピーを自動化できます。 そうしておくことで、運用上のイベントやセキュリティイベントの影響を受けた場合でも、データが常時利用可能になります。

リソース

関連するベストプラクティス:

- SEC01-BP06 標準的なセキュリティ統制のデプロイを自動化する
- SEC03-BP02 最小特権のアクセスを付与する
- SEC03-BP04 アクセス許可を継続的に削減する
- SEC04-BP04 非準拠リソースの修復を開始する
- SEC07-BP03 識別および分類を自動化する
- REL09-BP02 バックアップを保護し、暗号化する
- REL09-BP03 データバックアップを自動的に実行する

関連ドキュメント:

- AWS 規範ガイダンス: Automatically encrypt existing and new Amazon EBS volumes
- Ransomware Risk Management on AWS Using the NIST Cyber Security Framework (CSF)

関連する例:

- How to use AWS Config proactive rules and AWS CloudFormation Hooks to prevent creation of noncompliant cloud resources
- Automate and centrally manage data protection for Amazon S3 with AWS Backup
- AWS re:Invent 2023 - Implement proactive data protection using Amazon EBS snapshots
- AWS re:Invent 2022 - Build and automate for resilience with modern data protection

関連ツール:

- AWS CloudFormation Guard
- AWS CloudFormation Guard Rules Registry
- IAM Access Analyzer
- Amazon Macie
- AWS Backup
- Elastic Disaster Recovery

SEC08-BP04 アクセスコントロールを適用する

保管中のデータを保護するには、分離やバージョニングなどのメカニズムを使用してアクセス制御を実施します。最小特権と条件付きアクセスコントロールを適用します。データへパブリックアクセスが付与されるのを防止します。

期待される成果: そのデータについて知る必要がある、許可されたユーザーのみがデータにアクセスできるようにします。定期的なバックアップとバージョニングでデータを保護し、意図しない、または不注意によるデータの改ざんや削除を防止します。重要なデータを他のデータから分離して、機密性とデータ整合性を保護します。

一般的なアンチパターン:

- 機密度要件と分類の異なるデータを一緒に保管する。
- 復号化キーに、過度に寛容なアクセス許可を使用する。
- データを不適切に分類する。
- 重要なデータの詳細なバックアップを保持しない。
- 本番データへの永続的なアクセスを提供する。
- データアクセスを監査することも、定期的にアクセス許可を審査することもしていない。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

保管中のデータの保護は、データの整合性、機密性、規制要件の遵守を維持するうえで重要です。これを実現するために、アクセスコントロール、分離、条件付きアクセス、バージョニングなど、複数のコントロールを実装できます。

アクセスコントロールは最小特権の原則で適用できます。これにより、ユーザーとサービスがタスクの実行に必要とするアクセス許可のみが提供されます。これには、暗号化キーへのアクセスが含まれます。AWS Key Management Service (AWS KMS) ポリシーを確認して、付与するアクセスレベルが適切であり、関連する条件が適用されることを確認します。

レベルごとに異なる AWS アカウント を使用して、異なる分類レベルに基づいてデータを分離し、AWS Organizations を使用してこれらのアカウントを管理できます。この分離は、不正アクセスを防止し、データ漏洩のリスクを最小限に抑えるのに役立ちます。

Amazon S3 バケットのポリシーで付与されるアクセスのレベルを定期的にレビューします。絶対に必要な場合を除き、公開で読み取り可能なバケットや書き込み可能なバケットは使用しないでください。AWS Config を使用して公開されているバケットを検出し、Amazon CloudFront を使用して Amazon S3 からコンテンツを提供することを検討します。パブリックアクセスを許可してはならないバケットが、パブリックアクセスを防ぐように正しく構成されていることを確認します。

Amazon S3 に保存されている重要なデータのバージョニングと Object Lock メカニズムを実装します。Amazon S3 バージョニングは、以前のバージョンのオブジェクトを保存して、誤って削除または上書きされたデータを復元します。Amazon S3 Object Lockは、オブジェクトの必須アクセスコントロールを提供します。これにより、ロックの有効期限が切れるまで、ルートユーザーがオブジェクトを削除または上書きできなくなります。さらに、Amazon Glacier ボールトロックは、Amazon Glacier に保存されているアーカイブにも同様の機能を提供します。

実装手順

1. 最小特権の原則でアクセスコントロールを適用します。
   - ユーザーとサービスに付与されたアクセス許可を確認し、タスクを実行するために必要なアクセス許可のみを持っていることを確認します。
   - AWS Key Management Service (AWS KMS) ポリシーを確認し、暗号化キーへのアクセスを確認します。
2. 異なる分類レベルに基づいてデータを分離します。
   - データ分類レベルごとに異なる AWS アカウント を使用します。

- AWS Organizations を使用してこれらのアカウントを管理します。

3. Amazon S3 バケットとオブジェクトのアクセス許可を確認します。
   - Amazon S3 バケットのポリシーで付与されるアクセスのレベルを定期的にレビューします。
   - 絶対に必要な場合を除き、公開で読み取り可能なバケットや書き込み可能なバケットは使用しないでください。
   - AWS Config を使用して公開されているバケットを検出することを検討してください。
   - Amazon CloudFront を使用して Amazon S3 のコンテンツを提供します。
   - パブリックアクセスを許可してはならないバケットが、パブリックアクセスを防ぐように正しく構成されていることを確認します。
   - データベースや、SQS やサードパーティーのデータストアなど、IAM 認証を使用する他のデータソースにも同じレビュープロセスを適用できます。
4. AWS IAM Access Analyzer を使用する。
   - AWS IAM Access Analyzer を設定して、Amazon S3 バケットを分析し、S3 ポリシーが外部エンティティにアクセスを許可したときに検出結果を生成することができます。
5. バージョニングと Object Lock メカニズムを実装します。
   - Amazon S3 バージョニングを使用して、以前のバージョンのオブジェクトを保持すると、偶発的な削除や上書きから復旧できます。
   - Amazon S3 Object Lock を使用して、オブジェクトの必須アクセスコントロールを提供します。これにより、ロックの有効期限が切れるまで、ルートユーザーがオブジェクトを削除または上書きできなくなります。
   - Amazon Glacier に保存されているアーカイブには、Amazon Glacier ボールトロックを使用します。
6. Amazon S3 インベントリを使用する。
   - S3 オブジェクトのレプリケーションと暗号化ステータスの監査およびレポートには Amazon S3 インベントリを使用します。
7. Amazon EBS と AMI の共有アクセス許可を確認する。
   - Amazon EBS と AMI 共有の共有アクセス許可を確認し、イメージとボリュームがワークロード外の AWS アカウント で共有されていないことを確認します。
8. AWS Resource Access Manager 共有を定期的に確認する。
   - AWS Resource Access Manager を使用して、AWS Network Firewall ポリシー、Amazon Route 53 Resolver ルール、サブネットなど、Amazon VPC 内のリソースを共有できます。
   - 定期的に共有リソースを監査し、共有が不要になったリソースは共有を停止します。

リソース

関連するベストプラクティス:

- SEC03-BP01 アクセス要件を定義する
- SEC03-BP02 最小特権のアクセスを付与します

関連ドキュメント:

- AWS KMS Cryptographic Details Whitepaper
- Amazon S3 リソースへのアクセス許可の管理の導入
- AWS KMS リソースへのアクセス管理の概要
- AWS Config ルール
- Amazon S3 + Amazon CloudFront: A Match Made in the Cloud
- バージョニングの使用
- S3 Object Lock を使ってオブジェクトをロックする
- Sharing an Amazon EBS Snapshot
- 共有 AMI
- Amazon S3 で単一ページのアプリケーションをホストする
- AWS グローバル条件キー
- でのデータ境界の構築AWS

関連動画:

- Securing Your Block Storage on AWS

転送中のデータの保護

転送中のデータとは、システム間で送信されるすべてのデータを指します。これには、ワークロード内のリソース間での通信や他のサービスとエンドユーザーとの通信が含まれます。転送中のデータに適切なレベルの保護を提供することにより、ワークロードのデータの機密性と整合性を守ることができます。

VPC またはオンプレミスの場所間のデータを保護する: AWS PrivateLink を使用して、Amazon Virtual Private Cloud (Amazon VPC) 間、またはオンプレミスから AWS にホストされているサー

ビスへの保護されたプライベートネットワーク接続を構築できます。AWS サービス、サードパーティーサービス、および他の AWS アカウントのサービスを、あたかも自分のプライベートネットワークにあるかのように利用することができます。AWS PrivateLink を使うと、インターネットゲートウェイまたは NAT を使わずに、IP CIDR が重複するアカウント間のサービスにアクセスすることができます。また、ファイアウォールルール、パス定義、またはルートテーブルを設定する必要もありません。トラフィックは Amazon のバックボーンにとどまり、インターネットを横断しないため、お客様のデータは保護されます。HIPAA および EU/US プライバシーシールドなどの業界特有のコンプライアンス規制に対する準拠を維持できます。AWS PrivateLink はサードパーティーソリューションとシームレスに連携し、簡素化されたグローバルネットワークを作成するため、クラウドへの移行を加速させて利用可能な AWS のサービスを活用できます。

ベストプラクティス

- SEC09-BP01 安全な鍵および証明書管理を実装する
- SEC09-BP02 伝送中に暗号化を適用する
- SEC09-BP03 ネットワーク通信を認証する

SEC09-BP01 安全な鍵および証明書管理を実装する

Transport Layer Security (TLS) 証明書は、ネットワーク通信を保護し、インターネットやプライベートネットワーク上のウェブサイト、リソース、ワークロードの ID を確立するために使用されます。

期待される成果: 公開鍵基盤 (PKI) で証明書をプロビジョニング、デプロイ、保存、更新できる、安全な証明書管理システム。安全な鍵と証明書の管理メカニズムは、証明書のプライベートキーの内容が漏洩するのを防ぎ、自動的に証明書の定期更新を行います。また、他のサービスと統合して、ワークロード内のマシンリソースに安全なネットワーク通信と ID を提供します。キーの内容は、決して人的 ID にアクセス可能なものであってはなりません。

一般的なアンチパターン:

- 証明書のデプロイまたは更新プロセス中に手動で手順を実行する。
- プライベート認証機関 (CA) を設計する際、CA 階層に十分な注意を払わない。
- 公共リソースに自己署名証明書を使用する。

このベストプラクティスを活用するメリット:

- 自動デプロイと自動更新により証明書管理を簡素化する
- TLS 証明書を使用して転送中のデータの暗号化を奨励する

- 認証機関による証明書アクションのセキュリティと可監査性を向上させる
- CA 階層のさまざまなレイヤーにおける管理業務を整理する

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

最新のワークロードでは、TLS などの PKI プロトコルを使用して暗号化されたネットワーク通信が広く利用されています。PKI 証明書の管理は複雑になる場合がありますが、証明書のプロビジョニング、デプロイ、更新を自動化することで、証明書管理に伴う手間を軽減できます。

AWS には、汎用 PKI 証明書を管理するための 2 つのサービス、AWS Certificate Manager と AWS Private Certificate Authority (AWS Private CA) があります。ACM は、証明書をプロビジョニング、管理、デプロイして、パブリックとプライベート両方の AWS ワークロードで使用できるようにするための主要なサービスです。ACM は AWS Private CA を使用してプライベート証明書を発行し、他の多くの AWS Managed Services と連携してワークロード用の安全な TLS 証明書を提供します。ACM は、Amazon Trust Services からパブリックに信頼された証明書を発行することもできます。ACM のパブリック証明書は、最新のブラウザとオペレーティングシステムがデフォルトでこれらの証明書を信頼しているため、パブリック側ワークロードで使用できます。

AWS Private CA では、独自のルート認証機関または下位認証機関を確立し、API を通じて TLS 証明書を発行できます。こうした種類の証明書は、TLS 接続のクライアント側で信頼チェーンを制御し管理するシナリオで使用できます。TLS ユースケースに加えて、AWS Private CA は、Kubernetes ポッドへの証明書の発行、Matter デバイス製品認証、コード署名、カスタムテンプレートを使用したその他のユースケースにも使用できます。また、IAM Roles Anywhere を使用することで、プライベート CA によって署名された X.509 証明書が発行されたオンプレミスのワークロードに、一時的な IAM 認証情報を提供することもできます。

ACM と AWS Private CA に加えて、AWS IoT Core は、PKI 証明書のプロビジョニング、管理、IoT デバイスへのデプロイに特化したサポートを提供しています。AWS IoT Core は、公開鍵のインフラストラクチャに IoT デバイスを大規模にオンボーディングするための特殊な仕組みを備えています。

Amazon API Gateway や Elastic Load Balancing などの一部のAWSサービスは、証明書を使用してアプリケーション接続を保護する独自の機能を提供します。例えば、API Gateway と Application Load Balancer (ALB) はどちらも AWS マネジメントコンソール、CLI、または API を使用して作成およびエクスポートするクライアント証明書を使用した相互 TLS (mTLS) をサポートしています。

プライベート CA 階層を確立する際の考慮事項

プライベート CA を確立する必要がある場合、特別な注意を払って事前に CA 階層を適切に設計しておくことが重要です。プライベート CA 階層を作成する場合は、CA 階層の各レベルを個別の AWS アカウントにデプロイすることがベストプラクティスです。この意図的な手順により、CA 階層内の各レベルへの外部からのアクセスが減り、CloudTrail ログデータ内の異常をより簡単に発見できるようになります。また、いずれかのアカウントに不正アクセスがあった場合、アクセス範囲と影響が小さくなります。ルート CA はそれぞれ別のアカウントに保存し、1 件以上の中間 CA 証明書の発行にのみ使用すべきです。

次に、ルート CA のアカウントとは別のアカウントに 1 つ以上の中間 CA を作成し、エンドユーザー、デバイス、または他のワークロードに証明書を発行します。最後に、ルート CA から中間 CA に証明書を発行します。これにより、エンドユーザーまたはデバイスに証明書が発行されます。回復力の計画、クロスリージョンレプリケーション、組織全体での CA の共有など、CA デプロイの計画と CA 階層の設計の詳細については、「Planning your AWS Private CA deployment」を参照してください。

実装手順

1. ユースケースに必要となる適切な AWS サービスを判断します。
    - 多くのユースケースでは、AWS Certificate Manager を使用して既存の AWS パブリックキーインフラストラクチャを活用することができます。ACM は、ウェブサーバー、ロードバランサー、一般的に信頼されている証明書のその他の用途向けに TLS 証明書をデプロイする際に使用できます。
    - 独自のプライベート認証機関の階層を設定する必要がある場合や、エクスポート可能な証明書へのアクセスが必要な場合は、AWS Private CA の使用を検討します。それにより、ACM を使用して、AWS Private CA を使ったさまざまな種類のエンドエンティティ証明書を発行できます。
    - 組み込み型のモノのインターネット (IoT) デバイス向けに証明書を大規模にプロビジョニングする必要があるユースケースについては、AWS IoT Core の使用を検討します。
    - Amazon API Gateway や Application Load Balancer などのサービスでネイティブ mTLS 機能を使用することを検討してください。
2. 可能な限り、証明書の自動更新を実装してください。
    - ACM が発行した証明書に、ACM マネージド型更新を、統合された AWS のマネージドサービスと併せて使用します。
3. 認証機関を保有するアカウントへの

- CloudTrail logs を有効にして、認証機関を持つアカウントへのアクセスを追跡します。CloudTrail でログファイルの整合性検証を設定し、ログデータの信頼性を検証することを検討します。
- プライベート CA が発行または取り消しした証明書を一覧表示する監査レポートを、定期的に生成し、レビューします。これらのレポートは S3 バケットにエクスポートできます。
- プライベート CA をデプロイするときは、証明書失効リスト (CRL) を保存する S3 バケットも確立する必要があります。ワークロードの要件に基づいてこの S3 バケットを設定する方法については、「Planning a certificate revocation list (CRL)」を参照してください。

リソース

関連するベストプラクティス:

- SEC02-BP02 一時的な認証情報を使用する
- SEC08-BP01 安全なキー管理を実装する
- SEC09-BP03 ネットワーク通信を認証する

関連ドキュメント:

- How to host and manage an entire private certificate infrastructure in AWS
- How to secure an enterprise scale ACM Private CA hierarchy for automotive and manufacturing
- Private CA best practices
- How to use AWS RAM to share your ACM Private CA cross-account

関連動画:

- Activating AWS Certificate Manager Private CA (workshop)

関連する例:

- Private CA workshop
- IOT Device Management Workshop (デバイスプロビジョニングを含む)

関連ツール:

- Plugin to Kubernetes cert-manager to use AWS Private CA

SEC09-BP02 伝送中に暗号化を適用する

組織的、法的、コンプライアンス要件を満たすための組織のポリシー、法的義務と標準に基づいて、定義された暗号化要件を適用します。機密データを仮想プライベートクラウド (VPC) の外部に送信する場合は、暗号化されたプロトコルのみを使用します。暗号化を行うと、データが信頼できないネットワークを転送中も、データの機密性を保持できます。

期待される成果: リソースとインターネット間のネットワークトラフィックを暗号化して、データへの不正アクセスを軽減します。セキュリティ要件に従って、内部 AWS 環境内のネットワークトラフィックを暗号化します。転送中のデータはすべて、安全な TLS プロトコルと暗号スイートを使用して暗号化します。

一般的なアンチパターン:

- 廃止されたバージョンの SSL、TLS、および暗号スイートコンポーネント (SSL v3.0、1024-bit RSA キー、および RC4 暗号) を使用する。
- パブリック向けリソースとの間で暗号化されていない (HTTP) トラフィックを許可する。
- X.509 証明書をモニタリングし、期限が切れる前に交換しない。
- TLS に自己署名 X.509 証明書を使用する。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

AWS のサービスには、通信に TLS を使用し、AWS API との通信の際に伝送中データの暗号化を利用できる、HTTPS エンドポイントが用意されています。安全でない HTTP プロトコルは、セキュリティグループを使用して仮想プライベートクラウド (VPC) で監査およびブロックできます。HTTP リクエストは Amazon CloudFront 内または Application Load Balancer の HTTPS に自動的にリダイレクトすることもできます。Amazon Simple Storage Service (Amazon S3) バケットポリシーを使用すると、HTTP 経由でオブジェクトをアップロードする機能を制限し、バケットへのオブジェクトのアップロードに HTTPS の使用を効果的に強制できます。コンピューティングリソースを完全に制御して、サービス全体に伝送中データの暗号化を実装できます。また、外部ネットワークまたは AWS Direct Connect から VPC に VPN で接続することで、トラフィックの暗号化を円滑にすることができます。AWS では 2024 年 2 月で以前のバージョンの TLS の使用が廃止されたため、クライアントが少なくとも TLS 1.2 を使用して AWS API を呼び出すことを確認してください。TLS 1.3 を使用す

ることをお勧めします。転送中の暗号化に特別な要件がある場合は、AWS Marketplace で利用可能なサードパーティーのソリューションを見つけることができます。

実装手順

- 伝送中に暗号化を適用する: 暗号化の要件は、最新の標準とベストプラクティスに基づき、安全なプロトコルのみを許可する必要があります。例えば、Application Load Balancer または Amazon EC2 インスタンスに対してのみ HTTPS プロトコルを許可するよう、セキュリティグループを設定します。
- エッジサービスで安全なプロトコルを設定する: Amazon CloudFront を使用して HTTPS を設定し、自社のセキュリティ体制やユースケースに適したセキュリティプロファイルを使用します。
- 外部接続に VPN を使用する: ポイントツーポイント接続やネットワーク間接続を IPsec VPN で保護し、データのプライバシーと整合性の両方を提供することを検討します。
- ロードバランサーで安全なプロトコルを設定する: リスナーに接続するクライアントがサポートしている暗号スイートのなかで、もっとも堅牢な暗号スイートを提供しているセキュリティポリシーを選びます。Application Load Balancer 用の HTTPS リスナーを作成する
- Amazon Redshift で安全なプロトコルを設定する: Secure Socket Layer (SSL) または Transport Layer Security (TLS) 接続を必須とするように、クラスターを設定します。
- 安全なプロトコルを設定する: AWS サービスのドキュメントをレビューして、転送時の暗号化機能を決定します。
- Amazon S3 バケットへのアップロード時の安全なアクセスを設定する: Amazon S3 バケットポリシーコントロールを使用して、データへの安全なアクセスを適用します。
- AWS Certificate Manager の使用を検討する: ACM を使用すると、AWS サービスで使用するパブリック TLS 証明書のプロビジョニング、管理、デプロイが行えます。
- プライベート PKI のニーズには AWS Private Certificate Authority の使用を検討する: AWS Private CA を使用すると、プライベート認証局 (CA) 階層を作成してエンドエンティティ X.509 証明書を発行することができます。こちらは暗号化された TLS チャネルの作成に使用できます。

リソース

関連ドキュメント:

- CloudFront で HTTPS を使用する
- AWS Virtual Private Network を使用して VPC をリモートネットワークに接続する
- Create an HTTPS listener for your Application Load Balancer

- Tutorial: Configure SSL/TLS on Amazon Linux 2
- SSL/TLS を使用した DB インスタンスへの接続の暗号化
- 接続のセキュリティオプションを設定する

SEC09-BP03 ネットワーク通信を認証する

Transport Layer Security (TLS) や IPsec など、認証をサポートするプロトコルを使用して、通信の ID を検証します。

サービス間、アプリケーション間、またはユーザーへの通信には常に、安全で認証済みのネットワークプロトコルを使用するようにワークロードを設計してください。認証と認可をサポートするネットワークプロトコルを使用すると、ネットワークフローをより強力に制御し、不正アクセスの影響を軽減できます。

期待される成果: ワークロードで、サービス間のデータプレーンとコントロールプレーンのトラフィックフローが明確に定義されます。技術上可能な場合は必ず、認証および暗号化されたネットワークプロトコルをトラフィックフローが使用する。

一般的なアンチパターン:

- ワークロード内のトラフィックフローが暗号化されていない、または認証されていない。
- 複数のユーザーやエンティティで認証情報を再利用している。
- アクセス制御のメカニズムとしてネットワーク統制にばかり依存している。
- 業界標準の認証メカニズムに頼る代わりに、カスタムの認証メカニズムを作成する。
- VPC 内のサービスコンポーネントや他のリソース間のトラフィックフローが必要以上に許可されている。

このベストプラクティスを活用するメリット:

- 不正アクセスによる影響が及ぶ範囲をワークロードの一部に制限します。
- アシュアランスのレベルを上げ、認証済みのエンティティだけがアクションを実行するように徹底します。
- 導入予定のデータ転送インターフェイスを明確に定義し、実際に導入して、サービスの分離を強化します。
- リクエストのアトリビューションと、明確に定義された通信インターフェイスにより、モニタリング、ログ記録、インシデント対応を強化します。

- ネットワーク統制に認証と認可の制御を組み合わせることで、ワークロードに多層防御を提供します。

このベストプラクティスを活用しない場合のリスクレベル: 低

実装のガイダンス

ワークロードのネットワークトラフィックのパターンは、次の 2 つのカテゴリに分類できます。

- East-West トラフィックは、特定のワークロードを構成しているサービス間のトラフィックフローを表します。
- North-South トラフィックはワークロードとコンシューマー間のトラフィックフローを表します。

一般的には North-South トラフィックを暗号化し、認証済みプロトコルを用いて East-West トラフィックを保護する例はあまり見られません。最近のセキュリティ対策では、ネットワークの設計だけで、2 つのエンティティ間に信頼関係があるとは想定しないというのが通例となっています。2 つのサービスが共通のネットワーク境界内に存在する場合でも、それらのサービス間の通信を暗号化し、認証と認可を行うことがベストプラクティスです。

例えば、AWS サービス API は、リクエストの発信側のネットワークに関係なく、AWS Signature Version 4 (SigV4) 署名プロトコルを使用して発信者を認証します。この認証により、AWS API はアクションをリクエストした ID を検証し、その ID をポリシーと組み合わせて、アクションを許可するかどうかを判断する認可決定を行うことができます。

Amazon VPC Lattice や Amazon API Gateway などのサービスでは、同じ SigV4 署名プロトコルを使用して、独自のワークロードの East-West トラフィックに認証と認可を追加できます。SigV4 ベースの認証と認可が要求されるサービスに AWS 環境外のリソースから通信する必要がある場合は、非 AWS リソースに AWS Identity and Access Management (IAM) Roles Anywhere を使用することで、一時的な AWS 認証情報を取得できます。これらの認証情報は、SigV4 を使用してサービスへのリクエストに署名して、アクセスの認可を受けるために使用できます。

East-West トラフィックを認証するメカニズムとしては、TLS 相互認証 (mTLS) も一般的です。モノのインターネット (IoT)、ビジネス間アプリケーション、マイクロサービスの多くは、mTLS を採用しています。TLS 通信のクライアント側とサーバー側の両方が X.509 証明書を使用して、双方のアイデンティティを認証し合います。これらの証明書は AWS Private Certificate Authority (AWS Private CA) で発行できます。Amazon API Gateway などのサービスを使用して、ワークロード間またはワークロード内の通信で mTLS 認証を行うことができます。Application Load Balancer は、内

部または外部向けワークロードの mTLS もサポート」しています。mTLS は TLS 通信の両側に認証情報を提供しますが、認証のメカニズムは提供しません。

最後に、OAuth 2.0 と OpenID Connect (OIDC) の 2 つのプロトコルは、ユーザーがサービスへのアクセスを制御する際によく利用されていますが、最近はサービス間のトラフィックにもよく利用されています。API Gateway の JSON ウェブトークン (JWT) オーソライザーを使用すると、OIDC または OAuth 2.0 の ID プロバイダーが発行した JWT を使用して、API ルートへのアクセスをワークロードで制限することができます。OAuth2 のスコープを基本的な承認決定のソースとして使用できますが、依然として承認チェックをアプリケーション層に実装する必要があります。OAuth2 スコープ単体で複雑な承認ニーズに対応することはできません。

実装手順

- ワークロードのネットワークフローを定義し文書化する: 多層防御戦略を実装するには、まずワークロードのトラフィックフローを定義します。
  - ワークロードを構成するさまざまなサービス間でデータがどのように転送されるかを明確に定義したデータフロー図を作成します。これらのフローを認証済みのネットワークチャネルに実際に流していく前に、まずこの図を用意します。
  - 開発段階とテスト段階でワークロードを計測して、ランタイム時のワークロードの動作がデータフロー図に正確に反映されていることを確認してください。
  - データフロー図は、脅威モデリングを行う (「SEC01-BP07 脅威モデルを使用して脅威を特定し、緩和策の優先順位を付ける」を参照) ときにも役立ちます。
- ネットワーク統制を確立する: データフローに応じたネットワーク統制を確立するときは AWS の機能を使用することを検討します。ネットワーク境界は、それだけでは十分なセキュリティ統制にはなりませんが、ワークロードを保護する多層防御戦略の 1 層にはなります。
  - リソース間のデータフローの確立、定義、制限には、セキュリティグループを使用します。
  - AWS サービスと、AWS PrivateLink をサポートしているサードパーティーのサービスの両方と通信するときは、AWS PrivateLink の使用を検討します。AWS PrivateLink インターフェイスエンドポイントを介して送信されるデータは、AWS ネットワークバックボーン内にとどまり、公開インターネットを経由しません。
- ワークロード内のサービス間に認証と認可を実装する: ワークロード内のトラフィックフローの認証と暗号化を実現するために最も適した AWS サービスのセットを選択します。
  - サービス間通信のセキュリティを保護するときは、Amazon VPC Lattice の使用を検討します。VPC Lattice では、SigV4 認証を認証ポリシーと組み合わせて使用することで、サービス間のアクセスを制御できます。

  - mTLS を使用するサービス間通信では、API Gateway または Application Load Balancer の使用を検討します。AWS Private CA を使用すると、mTLS で使用する証明書を発行できる、プライベート CA 階層を作成できます。
  - OAuth 2.0 または OIDC を使用したサービスを統合するときは、API Gateway で JWT オーソライザーを使用することを検討します。
  - ワークロードと IoT デバイスとの通信には、AWS IoT Core の使用を検討します。これには、ネットワークトラフィックの暗号化と認証の方法が複数用意されています。
- 不正アクセスを監視する: 意図しない通信チャネル、不正なプリンシパルによる保護済みリソースへのアクセス、その他不適切なアクセスパターンを継続的にモニタリングします。
  - VPC Lattice を使用してサービスへのアクセスの管理するときは、VPC Lattice アクセスログを有効にしてモニタリングすることを検討します。これらのアクセスログには、リクエスト元のエンティティに関する情報、ソースとターゲットの VPC などのネットワーク情報、リクエストのメタデータが記録されています。
  - ネットワークフローのメタデータをキャプチャし、異常がないか定期的に点検するときは、VPC フローログを有効にすることを検討します。
  - セキュリティインシデントのプランニング、シミュレーション、対応に関する解説は、「AWS セキュリティインシデント対応ガイド」および「セキュリティの柱 - AWS Well-Architected Framework」内の「インシデント対応」のセクションを参照してください。

リソース

関連するベストプラクティス:

- SEC03-BP07 パブリックおよびクロスアカウントアクセスの分析
- SEC02-BP02 一時的な認証情報を使用する
- SEC01-BP07 脅威モデルを使用して脅威を特定し、緩和策の優先順位を付ける

関連ドキュメント:

- Evaluating access control methods to secure Amazon API Gateway APIs
- REST API の相互 TLS 認証の設定
- How to secure API Gateway HTTP endpoints with JWT authorizer
- Authorizing direct calls to AWS services using AWS IoT Core credential provider
- AWS セキュリティインシデント対応ガイド

関連動画:

- AWS re:invent 2022: Introducing VPC Lattice
- AWS re:invent 2020: Serverless API authentication for HTTP APIs on AWS

関連する例:

- Amazon VPC Lattice Workshop
- Zero-Trust Episode 1 – The Phantom Service Perimeter workshop

インシデントへの対応

成熟した予防的、発見的統制が実装されていても、組織はセキュリティインシデントの潜在的な影響に対応し、影響を緩和するメカニズムを実装する必要があります。準備することで、インシデントの際にチームが効果的に動作し、問題を切り分け、封じ込め、フォレンジックを実行し、運用を既知の正常な状態に復元する能力に強く影響します。セキュリティインシデントが起こる前にツールとアクセス権を整備し、ゲームデー (実践訓練) を通じてインシデント対応を定期的に実施しておけば、ビジネスの中断を最小限に抑えながら復旧することができます。

トピック

AWS におけるインシデント対応の諸側面

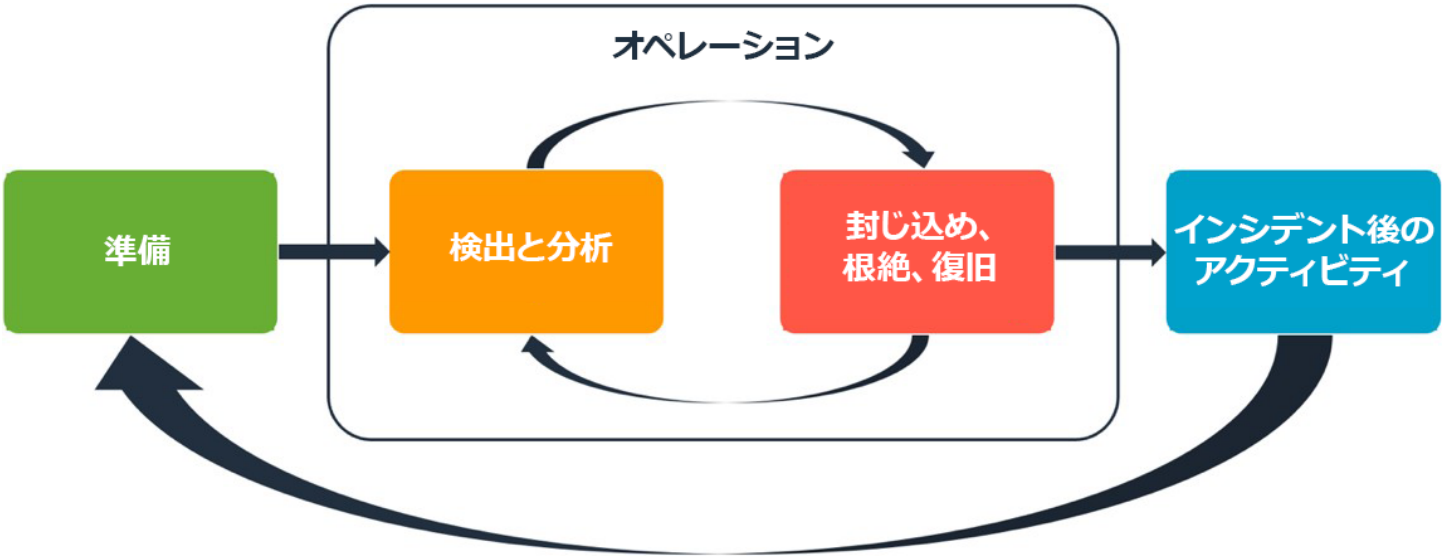
組織内のすべての AWS ユーザーは、セキュリティインシデント対応プロセスの基本を理解している必要があり、セキュリティ担当者はセキュリティ問題への対応方法を理解している必要があります。教育、トレーニング、経験は、クラウドインシデント対応プログラムを成功させるために不可欠であり、起こり得るセキュリティインシデントに対処する前に十分な余裕を持って実施するのが理想的です。クラウドでのインシデント対応プログラムの成功基盤は、準備、オペレーション、インシデント後アクティビティです。

これらの各側面を理解するには、以下の説明を参考にしてください。

- 準備: 検出制御を有効にし、必要なツールやクラウドサービスへの適切なアクセスを検証することで、インシデント対応チームが AWS 内のインシデントを検出して対応できるように準備します。さらに、信頼性の高い一貫した応答を検証するために、手動と自動の両方で必要なプレイブックを準備します。
- オペレーション: NIST のインシデント対応フェーズ (検出、分析、封じ込め、根絶、復旧) に従って、セキュリティイベントと潜在的なインシデントに対処します。

- インシデント後アクティビティ: セキュリティイベントとシミュレーションの結果を反復することで、対応の有効性を改善し、対応と調査から得られる価値を高め、リスクをさらに軽減します。インシデントから学び、改善活動に対する強いオーナーシップを持つ必要があります。

下図は、前述の NIST のインシデント対応ライフサイクルに沿った、これらの側面のフローを示しています。ここでの業務には、検出と分析に加えて、封じ込め、根絶、復旧が含まれています。



AWS におけるインシデント対応の諸側面

クラウドレスポンスの設計目標

「NIST SP 800-61 コンピュータセキュリティインシデント処理ガイド」などで定義されてインシデント対応の一般的なプロセスとメカニズムは変わりませんが、クラウド環境でのセキュリティインシデントへの対応に関連する以下の特定の設計目標を評価することをお勧めします。

- 対応目標の確立: ステークホルダー、法律顧問、組織のリーダーと協力してインシデント対応の目標を決定します。共通の目標には、問題の封じ込めと緩和、影響を受けたリソースの復旧、フォレンジック用のデータの保全、既知の安全な運用への復帰、そして最終的にはインシデントからの学習などがあります。
- クラウドを使用して応答する: イベントとデータが発生するクラウド内に応答パターンを実装します。
- 持っているものと必要なものを知る: ログ、リソース、スナップショット、その他の証拠は、対応専用の一元化されたクラウドアカウントにコピーして保存します。管理ポリシーを適用するタグ、メタデータ、メカニズムを使用します。使用しているサービスを把握し、それらのサービスを調査

するための要件を特定する必要があります。環境を把握しやすくするために、タグ付けを使用することもできます。

- 再デプロイメカニズムを使用する: セキュリティの異常が設定ミスに起因する場合は、適切な設定でリソースを再デプロイして差異を取り除くだけで解決できる場合があります。セキュリティ侵害の可能性が見つかった場合は、根本原因に対する適切で検証済みの緩和策が再デプロイに含まれていることを確認します。
- 可能な場合は自動化する: 問題が発生したり、インシデントが繰り返されたりした場合は、一般的なイベントをプログラムで優先順位付けして対応するメカニズムを構築します。自動化が不十分で、特殊かつ複雑、または機密性の高いインシデントには、人手で対応します。
- スケーラブルなソリューションを選択する: 組織のアプローチのスケーラビリティがクラウドコンピューティングと適合しているように努めます。環境全体にスケールできる検出および対応のメカニズムを実装して、検出から対応までの時間を効果的に短縮します。
- プロセスを学び、改善する: プロセス、ツール、人員におけるギャップを積極的に特定し、それらを修正する計画を実施します。シミュレーションは、ギャップを見つけてプロセスを改善する安全な方法です。

これらの設計目標は、インシデント対応と脅威検知の両方を実施する能力について、アーキテクチャの実装を確認することを促すものです。クラウドの実装を計画するときは、インシデントへの対応を検討します。フォレンジックに基づいた対応方法論を使用するのが理想的です。これは、場合によっては、このような対応タスク用に複数の組織、アカウント、ツールを特別に設定することを意味します。これらのツールと機能は、デプロイパイプラインによってインシデント対応担当者が利用できるようにする必要があります。リスクを大きくする可能性があるため、静的な状態のままにしないでください。

準備

インシデントへの準備は、タイムリーかつ効果的なインシデント対応にとって重要です。準備は次の 3 つの分野にわたって行われます。

- 人員: セキュリティインシデントに備えて人員を準備するには、インシデント対応に関連するステークホルダーを特定し、インシデント対応とクラウド技術に関するトレーニングを行う必要があります。
- プロセス: セキュリティインシデントに備えてプロセスを準備するには、アーキテクチャの文書化、徹底的なインシデント対応計画の策定、セキュリティイベントへの一貫した対応のためのプレイブックの作成が必要です。

- テクノロジー: セキュリティインシデントに備えてテクノロジーを準備するには、アクセスの設定、必要なログの集約と監視、効果的なアラートメカニズムの実装、対応と調査機能の開発が必要です。

これらの各分野は、効果的なインシデント対応にとって等しく重要です。3 つすべてが揃わなければ、インシデント対応プログラムは完全でも効果的でもありません。インシデントに備えるには、人員、プロセス、テクノロジーを緊密に連携して準備する必要があります。

ベストプラクティス

- SEC10-BP01 重要な人員と外部リソースを特定する:
- SEC10-BP02 インシデント管理計画を作成する
- SEC10-BP03 フォレンジック機能を備える:
- SEC10-BP04 セキュリティインシデント対応プレイブックを作成し、テストする
- SEC10-BP05 アクセスを事前プロビジョニングする
- SEC10-BP06 ツールを事前デプロイする
- SEC10-BP07 シミュレーション行う

SEC10-BP01 重要な人員と外部リソースを特定する:

組織のインシデント対応体制を整えるため、組織内外の担当者、リソース、法的義務を特定します。

期待される成果: 主要な担当者、その連絡先情報、セキュリティイベントに対応する際のその役割から成る一覧を作成します。この情報を定期的に見直し、組織内外のツールの観点から人員配置の変更を反映させます。この情報の文書化にあたっては、セキュリティパートナー、クラウドプロバイダー、SaaS (Software-as-a-Service) アプリケーションなど、サードパーティーのサービスプロバイダーやベンダーをすべて考慮します。セキュリティイベントの発生時は、適切な責任とアクセス権を持つ担当者が状況を適切に理解して、対応と復旧にあたることができます。

一般的なアンチパターン:

- 主要担当者の連絡先と、セキュリティイベントへの対応時の役割や責任について最新情報をまとめたリストが用意されていない。
- イベントへの対応や復旧の際に、担当者、依存関係、インフラストラクチャ、ソリューションについて全員がわかっているものだと想定している。
- 主要なインフラストラクチャやアプリケーションの設計を記載したドキュメントまたはナレッジリポジトリがない。

- セキュリティイベント発生時の効果的な対応方法を新しい人員に指導する適切なオンボーディングプロセス (イベントシミュレーションの実施など) が用意されていない。
- 主要担当者が一時的に不在の場合や、セキュリティイベントの発生時に対応できない場合に備えたエスカレーションパスが用意されていない。

このベストプラクティスを活用するメリット: このベストプラクティスを活用することで、イベント発生時に適切な担当者とその役割とを特定するのにかかる、トリアージと対応の時間を短縮することができます。主要担当者とその役割の最新リストが用意してあれば、適切な担当者をイベントのトリアージと復旧に投入でき、イベント発生時の時間の無駄使いを極力抑えることができます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

組織内の主要な担当者を特定する: インシデント対応に必要な、組織内の担当者の連絡先一覧を保存します。この情報を定期的に見直し、組織編成の変更、昇進、チームの変更など、人員配置に変更があった場合は適宜更新してください。インシデント管理者、インシデント対応者、コミュニケーションリーダーなどの主要な役割については特に重要です。

- インシデント管理者: インシデント管理者は、イベント対応時のすべての権限を有します。
- インシデント対応者: インシデント対応者はインシデントの調査および修正を担当する人です。これらの人員はイベントの種類によって異なりますが、通常は、影響を受けたアプリケーションを担当する開発者や運用チームです。
- コミュニケーションリーダー: コミュニケーションリーダーは、組織内外とのコミュニケーション、特に公的機関、規制当局、顧客とのコミュニケーションを担当します。
- オンボーディングプロセス: インシデント対応活動に効果的に貢献するために必要なスキルと知識を新入社員に身につけさせるために、定期的にトレーニングとオンボーディングを実施します。オンボーディングプロセスの一環としてシミュレーションと実践演習を取り入れて準備を整える
- 対象分野のエキスパート (SME): 分散型の自律的なチームの場合は、ミッションクリティカルなワークロードに SME を指名しておくことが推奨されます。SME は、イベントに関与する重要なワークロードの運用とデータ分類に関する深い知識を共有してくれます。

テーブル形式の例:

```
  | Role | Name | Contact Information | Responsibilities |
1 | —- | —- | —- | —- |
```

```
2 | Incident Manager | Jane Doe| jane.doe@example.com | Overall authority during
 response |
3 | Incident Responder | John Smith | john.smith@example.com | Investigation and
 remediation |
4 | Communications Lead | Emily Johnson | emily.johnson@example.com | Internal and
 external communications |
5 | Communications Lead | Michael Brown | michael.brown@example.com | Insights on
 critical workloads |
```

主要連絡先の入手、対応プランの策定、オンコールスケジュールの自動化、エスカレーションプランの作成のため、AWS Systems Manager Incident Manager 機能を使用することを検討します。オンコールスケジュールでスタッフ全員を自動でローテーションさせ、ワークロードの責任を所有者間で分担できます。これにより、関連するメトリクスやログの生成、ワークロードにとって重要なアラームしきい値の定義など、優れた取り組みが促されます。

外部パートナーを特定する: 顧客向けに差別化されたソリューションを構築するため、多くの企業が、独立系ソフトウェアベンダー (ISV)、パートナー、下請け業者が構築したツールを使用しています。これらの関係先の担当者に、インシデントへの対応と復旧を支援してもらいます。サポートケースを通じて AWS の対象分野のエキスパートに迅速に連絡できるように、適切なレベルの サポートにサインアップすることをお勧めします。ワークロード用のすべての重要なソリューションプロバイダーに対して、同様の取り決めを検討してください。一部のセキュリティイベントについては、上場企業は該当イベントとその影響を関連する公的機関や規制当局に通知する義務があります。関連部門や担当者の連絡先情報を管理し、更新します。

実装手順

1. インシデント管理ソリューションを設定します。
    a. セキュリティツール用アカウントに Incident Manager をデプロイすることを検討してください。
2. インシデント管理ソリューションで連絡先を定義します。
    a. インシデントの発生時に連絡が取れるように、連絡先ごとに少なくとも 2 種類の連絡チャネル (SMS、電話、E メールなど) を定義します。
3. 対応計画を定義します。
    a. インシデント発生時の対応要員として最適な連絡先を特定します。個々の連絡先ではなく、対応担当者の役割に合わせたエスカレーション計画を定義します。インシデントの解決に直接関与していない場合でも、外部機関への情報提供を担当する可能性のある連絡先を含めておくことを検討してください。

リソース

関連するベストプラクティス:

- OPS02-BP03 パフォーマンスに責任を持つ所有者が運用アクティビティに存在する

関連ドキュメント:

- AWS セキュリティインシデント対応ガイド

関連する例:

- AWS customer playbook framework
- Prepare for and respond to security incidents in your AWS environment

関連ツール:

- AWS Systems Manager Incident Manager

関連動画:

- Amazon's approach to security during development

SEC10-BP02 インシデント管理計画を作成する

インシデント対応のために最初に作成する文書は、インシデント対応計画です。インシデント対応計画は、インシデント対応プログラムと戦略の基礎となるように設計されています。

このベストプラクティスを活用するメリット: インシデント対応のプロセスを熟考し、明確に定義することは、インシデント対応プログラムを成功させ、拡張性を持たせるための鍵となります。セキュリティイベントが発生した場合、明確な手順とワークフローがあれば、タイムリーに対応できます。既にインシデント対応プロセスがある場合もあります。現在の状態にかかわらず、インシデント対応プロセスを定期的に更新、反復、テストすることが重要です。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

インシデント管理計画は、セキュリティインシデントの潜在的な影響への対応、復旧、軽減に不可欠です。インシデント管理計画は、セキュリティインシデントをタイムリーに特定し、修復、対応するための体系的なプロセスです。

クラウドには、オンプレミス環境と同じオペレーション上のロールと要件があります。インシデント管理計画を作成する際は、ビジネス成果とコンプライアンス要件と最も合致する対応および復旧戦略を組み込むことが重要です。例えば、米国で FedRAMP に準拠している AWS のワークロードを運用する場合は、NIST SP 800-61 Computer Security Handling Guide の推奨事項に従ってください。同様に、個人を特定できる情報 (PII) を保存するワークロードを操作する場合は、データレジデンシーと使用に関連する問題を保護し、対応する方法を検討してください。

AWS のワークロードに関するインシデント管理計画を策定するときは、AWS 責任共有モデルの検討から開始して、インシデント対応に向けた多層防御の戦略を構築していきます。このモデルでは、AWS はクラウドのセキュリティを管理します。クラウド内のセキュリティについてはお客様の責任です。つまり、お客様はコントロールを保持するとともに、実装しようとするセキュリティコントロールに責任を持つということです。「AWS セキュリティインシデント対応ガイド」には、クラウド中心のインシデント管理計画を策定するための主要なコンセプトと基本のガイダンスが記載されています。

効果的なインシデント管理計画は、クラウド運用の目標に沿って継続的に繰り返し、最新の状態に保つ必要があります。インシデント管理計画を作成して進化させるにあたり、以下に記載の実装計画を使用することを検討してください。

実装手順

1. セキュリティイベントを処理するための組織内の役割と責任を定義します。これには、以下のようなさまざまな部門の担当者が関与する必要があります。
   - 人事 (HR)
   - 経営陣
   - 法務部
   - アプリケーションの所有者とデベロッパー (対象分野のエキスパート、または SME)
2. インシデント発生時の責任者、説明責任、相談者、情報伝達者 (RACI) を明確に示します。RACI チャートを作成して、迅速かつ直接的なコミュニケーションを容易にし、イベントのさまざまなステージにわたるリーダーシップを明確に概説します。
3. インシデント中にアプリケーションの所有者とデベロッパー (SME) を関与させます。彼らから影響の測定に役立つ貴重な情報とコンテキストを得られるためです。これらの SME との関係を構築

し、実際のインシデントが発生する前に、SME とインシデント対応シナリオの演習を行ってください。

4. 信頼できるパートナーや外部の専門家を調査や対応プロセスに参加させることにより、さらなる専門知識や視点を得ることができます。
5. インシデント管理の計画とロールを、組織を管理する現地の規制やコンプライアンス要件に合わせます。
6. インシデント対応計画を定期的に練習してテストし、定義されたすべての役割と責任を含めます。これにより、プロセスを合理化し、セキュリティインシデントへの対応が調整され、効率的な対応が行われていることを確認できます。
7. ロール、責任、RACI チャートを定期的に、または組織構造や要件の変化に応じて見直して更新します。

AWS 対応チームとサポートを理解する

- AWS サポート
  - サポート は、AWS ソリューションの成功とオペレーションの正常性をサポートするツールと専門知識にアクセスできる一連のプランを用意しています。AWS 環境の計画、導入、最適化に役立つテクニカルサポートや、より多くのリソースが必要な場合は、AWS ユースケースに最適なサポートプランを選択できます。
  - AWS リソースに影響する問題に関してサポートを得るための連絡窓口として、AWS マネジメントコンソール (サインインが必要) のサポートセンター を検討します。サポートへのアクセスは AWS Identity and Access Management によって制御されます。サポートの機能を利用する方法については、「Getting started with サポート」を参照してください。
- AWS カスタマーインシデント対応チーム (CIRT)
  - AWS カスタマーインシデント対応チーム (CIRT) は、AWS 責任共有モデルの顧客側のセキュリティイベントが発生したときに顧客にサポートを提供する、24 時間対応の専門のグローバル AWS チームです。
  - AWS CIRT がお客様をサポートすると、AWS で発生しているセキュリティイベントの優先順位付けと復旧を支援します。AWS サービスログを使用して根本原因の分析を支援し、復旧のための推奨事項を提示します。また、将来のセキュリティイベントを回避するのに役立つセキュリティに関する推奨事項やベストプラクティスを提供することもできます。
  - AWS のお客様は、サポート case から AWS CIRT と連携することができます。
- AWS Security Incident Response

  - re:Invent 2024 で発表された AWS Security Incident Response は、最新のトリアージテクノロジーとヒューマンインザループの両方を使用するマネージドセキュリティインシデント対応サービスです。このサービスは、すべての GuardDuty の検出結果と、トリアージのために AWS Security Hub CSPM に送信されたサードパーティーの検出結果を取り込み、調査が必要な検出結果についてのみ顧客に警告します。また、このサービスは、顧客がセキュリティイベントの発生時に事後対応ケースを送信し、AWS の高度なインシデント対応チームからサポートを受けるためのポータルも提供します。
- DDoS 対応のサポート
  - AWS が提供する AWS Shield は、AWS で実行中のウェブアプリケーションを保護する、分散型サービス拒否攻撃 (DDoS) のマネージド型防御サービスです。Shield を使用すれば、常時検出と自動インライン緩和の機能により、アプリケーションのダウンタイムとレイテンシーを最小限に抑えることができ、DDoS 防御に サポートを使う必要がなくなります。Shield は、AWS Shield Standard と AWS Shield Advanced の 2 種類に分かれています。両者の違いに関する詳細は、「AWS Shield の特徴」を参照してください。
- AWS Managed Services (AMS)
  - AWS Managed Services (AMS) は AWS インフラストラクチャ管理を継続的に提供するため、お客様はアプリケーションに集中できます。AMS は、ベストプラクティスを実行してインフラストラクチャを管理することで、運用のオーバーヘッドとリスクを減らします。AMS は、変更リクエスト、モニタリング、パッチ管理、セキュリティ、バックアップサービスなどの一般的なアクティビティを自動化し、インフラストラクチャをプロビジョニング、実行、サポートする、ライフサイクル全般にわたるサービスを提供します。
  - AMS は、一連のセキュリティ検出コントロールの展開に責任を持ち、24 時間 365 日、第一線でアラートに対応します。アラートが発生すると、AMS は標準的な自動プレイブックと手動プレイブックに従って、一貫した対応が行われていることを確認します。これらのプレイブックはオンボーディング中に AMS の顧客に共有されるため、顧客は AMS と対応策を練り、調整することができます。

インシデント対応計画の策定

インシデント対応計画は、インシデント対応プログラムと戦略の基礎となるように設計されています。インシデント対応計画は正式な文書にする必要があります。インシデント対応計画には通常、次のセクションが含まれます。

- インシデント対応チームの概要: インシデント対応チームの目標と機能の概要が記されています。
- 役割と責任: インシデントに対応する利害関係者が一覧表示され、インシデント発生時のそれぞれの役割が詳しく記されています。

- コミュニケーションプラン: 連絡先とインシデント発生時の連絡方法が記されています。
- 通信手段のバックアップ: インシデント関連の通信のバックアップ方法としては、帯域外通信を確保することがベストプラクティスです。安全な帯域外通信チャネルを提供するアプリケーションの例は AWS Wickr です。
- インシデント対応の各段階と取るべき措置: インシデント対応の各段階 (検出、分析、根絶、封じ込め、復旧など) を一覧にし、各段階で取るべき措置を大まかに記しています。
- インシデントの深刻度と優先順位の決定: インシデントの深刻度の分類方法、インシデントの優先付け方法、深刻度の定義がエスカレーション手順にどう影響するか、を詳しく説明しています。

これらのセクションは、さまざまな規模や業界の企業で共通していますが、各組織のインシデント対応計画は異なります。組織に最適なインシデント対応計画を立てる必要があります。

リソース

関連するベストプラクティス:

- SEC04 検知

関連ドキュメント:

- AWS セキュリティインシデント対応ガイド
- NIST: Computer Security Incident Handling Guide

SEC10-BP03 フォレンジック機能を備える:

セキュリティインシデントが発生する前に、セキュリティイベントの調査を支援するフォレンジック機能の整備を検討します。

このベストプラクティスを活用しない場合のリスクレベル: 中

AWS には、従来のオンプレミスフォレンジックの概念が適用されます。AWS クラウドでフォレンジック機能の構築を開始するための重要な情報については、「Forensic investigation environment strategies in the AWS クラウド」を参照してください。

フォレンジックのための環境と AWS アカウント構造が整ったら、次の 4 つのフェーズにわたってフォレンジックに適した方法論を効果的に実行するために必要なテクノロジーを定義します。

- 収集: AWS CloudTrail、AWS Config、VPC フローログ、ホストレベルのログなどの関連 AWS ログを収集します。可能であれば、影響を受けた AWS リソースのスナップショット、バックアップ、メモリダンプを収集します。
- 調査: 関連する情報を抽出して評価することにより、収集されたデータを検証します。
- 分析: 収集したデータを分析してインシデントを解明し、そこから結論を導き出します。
- レポート: 分析フェーズから得られた情報を報告します。

実装手順

フォレンジック環境を準備する

AWS Organizations では、AWS リソースの拡大とスケールに合わせて、AWS 環境を一元的に管理および運用できます。AWS 組織を利用することで、AWS アカウントを統合して 1 つのユニットとして管理できるようになります。組織単位 (OU) を使用すると、アカウントをまとめてグループ化し、単一の単位として管理できます。

インシデント対応には、セキュリティ OU および フォレンジック OU を含むインシデント対応の機能をサポートする AWS アカウント構造があると便利です。セキュリティ OU 内には、次のアカウントが必要です。

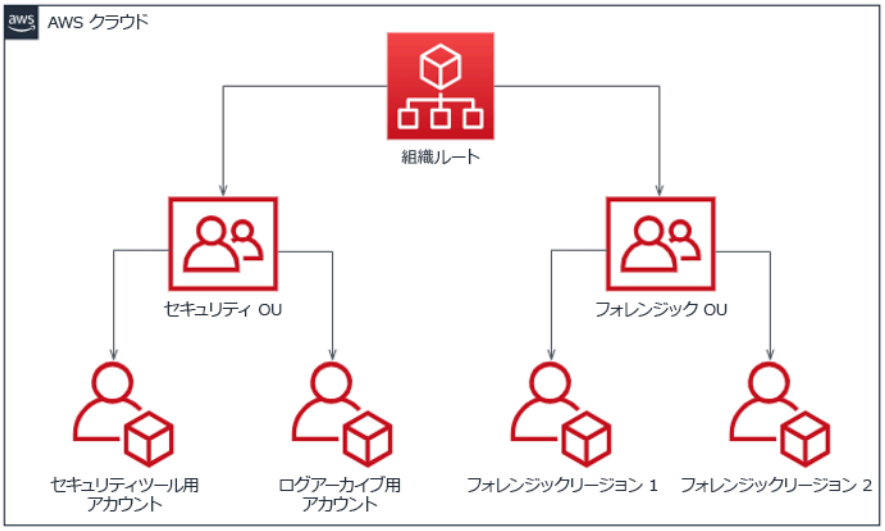
- ログアーカイブ: 限られたアクセス許可を持つログアーカイブ用の AWS アカウントにログを集約します。
- セキュリティツール: セキュリティサービスをセキュリティツール用の AWS アカウントに一元化します。このアカウントは、セキュリティサービスの委任管理者として機能します。

フォレンジック OU 内では、お客様のビジネスモデルと運用モデルに最適なフォレンジックアカウントに応じて、フォレンジック用に 1 つのアカウントを実装するか、事業を展開するリージョンごとにアカウントを実装できます。リージョンごとにフォレンジックアカウントを作成すると、そのリージョン外での AWS リソースの作成をブロックし、リソースが意図しないリージョンにコピーされるリスクを低減できます。例えば、米国東部 (バージニア北部) リージョン (us-east-1) および米国西部 (オレゴン) (us-west-2) のみで運用する場合、フォレンジック OU には 2 つのアカウントがあります。1 つは us-east-1 用で、もう 1 つは us-west-2 用です。

複数のリージョンのフォレンジック AWS アカウントを作成できます。そのアカウントに AWS リソースをコピーする場合は、データ主権に関する要件に準拠しているか注意する必要があります。新しいアカウントのプロビジョニングには時間がかかるため、インシデントのかなり前にフォレンジッ

クアカウントを作成して実装し、対応担当者が効果的に対応できるように準備しておくことが重要です。

次の図は、リージョンごとのフォレンジックアカウントを持つフォレンジック OU を含むアカウント構造の例を示しています。



インシデント対応のためのリージョンごとのアカウント構造

バックアップとスナップショットをキャプチャする

主要なシステムとデータベースのバックアップをセットアップすることは、セキュリティインシデントからの回復とフォレンジックのために重要です。バックアップを作成しておけば、システムを以前の安全な状態に復元できます。AWS では、さまざまなリソースのスナップショットを作成できます。スナップショットでは、こうしたリソースのポイントインタイムバックアップを作成できます。バックアップや復旧をサポートできる AWS のサービスは数多くあります。これらのサービス、バックアップと復旧のアプローチの詳細については、バックアップと復旧についての規範ガイダンスおよび「Use backups to recover from security incidents」を参照してください。

特にランサムウェアのような状況では、バックアップをしっかりと保護することが重要です。バックアップの保護に関するガイダンスについては、「Top 10 security best practices for securing backups in AWS」を参照してください。バックアップの保護に加えて、バックアップと復元のプロセスを定期的にテストして、導入しているテクノロジーとプロセスが想定どおりに機能することを確認する必要があります。

フォレンジックを自動化する

セキュリティイベント中、インシデント対応チームは、イベント前後の期間の証拠を、正確性を維持しながら迅速に収集して分析できなければなりません (特定のイベントやリソースに関連するログ

のキャプチャ、Amazon EC2 インスタンスのメモリダンプの収集など)。インシデント対応チームにとって、関連する証拠を手作業で収集することは困難であり、時間もかかります。多数のインスタンスやアカウントが対象となる場合は特にそうです。さらに、手作業による収集では人為的ミスが起こりやすくなります。このような理由から、フォレンジックの自動化を可能な限り開発し、実装する必要があります。

AWS には、フォレンジック用の自動化リソースが多数用意されており、これらのリソースは以下のリソースセクションに一覧表示されています。これらのリソースは、当社が開発し、お客様が実装したフォレンジックパターンの例です。手始めに参考にするリファレンスアーキテクチャとしては有効かもしれませんが、環境、要件、ツール、フォレンジックプロセスに基に変更するか、新しいフォレンジック自動化パターンを作成することを検討してください。

リソース

関連ドキュメント:

- AWS Security Incident Response Guide - Develop Forensics Capabilities
- AWS Security Incident Response Guide - Forensics Resources
- Forensic investigation environment strategies in the AWS クラウド
- How to automate forensic disk collection in AWS
- AWS 規範ガイダンス - Automate incident response and forensics

関連動画:

- Automating Incident Response and Forensics

関連する例:

- Automated Incident Response and Forensics Framework
- Automated Forensics Orchestrator for Amazon EC2

SEC10-BP04 セキュリティインシデント対応プレイブックを作成し、テストする

インシデント対応プロセスを準備する上で重要なのは、プレイブックを作成することです。インシデント対応プレイブックには、セキュリティイベントが発生したときに従うべき規範的なガイダンスと

手順が記載されています。明確な体制と手順があると、対応が簡単になり、人為的ミスの可能性が低くなります。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

プレイブックは、次のようなインシデントシナリオ向けに作成する必要があります。

- 予想されるインシデント: プレイブックは、予測されるインシデントに合わせて作成する必要があります。これには、サービス拒否 (DoS)、ランサムウェア、認証情報の漏えいなどの脅威が含まれます。
- 既知のセキュリティ上の検出結果またはアラート: Amazon GuardDuty などの既知のセキュリティ検出結果やアラートに対処するには、プレイブックを作成する必要があります。GuardDuty の検出結果を受け取った場合、プレイブックには、アラートの誤った処理や無視を防ぐための明確な手順が記載されている必要があります。修復の詳細とガイダンスについては、「GuardDuty によって検出されたセキュリティ問題の修復」を参照してください。

プレイブックには、起こりうるセキュリティインシデントを適切に調査して対応するために、セキュリティアナリストが実行すべき技術的な手順を記載する必要があります。

AWS Customer Incident Response Team (CIRT) は、脅威シナリオ、タイプ、リソース別に整理されたインシデント対応プレイブックを含む GitHub リポジトリを公開しました。これらのプレイブックは、既存のインシデント対応手順に合わせて適用することも、新しいインシデント対応手順を開発するための基盤として使用することもできます。

実装手順

プレイブックに記載すべき項目には次のようなものがあります。

- プレイブックの概要: このプレイブックがどのようなリスクやインシデントシナリオに対応しているか。このプレイブックの目的は何か。
- 前提条件: このインシデントシナリオには、どのようなログ、検出メカニズム、自動ツールが必要か。どのような通知が想定されるか。
- コミュニケーションとエスカレーションに関する情報: 関与している人員およびその連絡先情報。各利害関係者の責任は何か。
- 対応ステップ: インシデント対応の各フェーズで、どのような戦術的措置を講じるべきか。アナリストはどのようなクエリを実行すべきか。望ましい結果を得るためにどのようなコードを実行すべきか。

  - 検知: インシデントはどのように検出されるか。
  - 分析: 影響範囲はどのように特定されるか。
  - 封じ込め: 影響範囲を限定するために、インシデントをどのように隔離するか。
  - 根絶: どのようにして脅威を環境から取り除くか。
  - 復旧: 影響を受けたシステムやリソースをどのようにして本番環境に戻すか。
- 期待される結果: クエリとコードが実行された後、プレイブックで想定される結果はどのようなものか。

リソース

関連する Well-Architected のベストプラクティス:

- SEC10-BP02 - インシデント管理計画を作成する

関連ドキュメント:

- Framework for Incident Response Playbooks
- Develop your own Incident Response Playbooks
- Incident Response Playbook Samples
- Building an AWS incident response runbook using Jupyter playbooks and CloudTrail Lake

SEC10-BP05 アクセスを事前プロビジョニングする

インシデント対応者が AWS に事前プロビジョニングされた正しいアクセス権を持っていることを検証しておき、調査から復旧までに必要な時間を短縮します。

一般的なアンチパターン:

- ルートアカウントをインシデント対応に使用する
- 既存のアカウントに変更を加える
- ジャストインタイムの権限昇格を提供する際に IAM アクセス許可を直接操作する

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

AWS は、可能であれば長期的な認証情報への依存を削減または排除し、一時的な認証情報とジャストインタイムの権限昇格メカニズムを優先することを推奨します。長期的な認証情報は、セキュリティリスクにさらされやすく、オペレーションのオーバーヘッドを増大させます。ほとんどの管理タスクと、インシデント対応タスクについては、管理アクセスの一時的な昇格と併せて ID フェデレーションを実装することをお勧めします。このモデルでは、ユーザーはより高いレベルの権限 (インシデント対応ロールなど) への昇格をリクエストします。ユーザーに昇格の資格がある場合、リクエストは承認者に送信されます。リクエストが承認された場合、ユーザーは、一時的な AWS 認証情報のセットを受け取り、これを使用してタスクを完了できます。これらの認証情報の期限が切れたら、ユーザーは新たな昇格リクエストを送信する必要があります。

インシデント対応の大半のケースでは、一時的な権限昇格を使用することをお勧めします。そのための適切な方法は、AWS Security Token Service および セッションポリシーを使用してアクセスのスコープを定義することです。

ID フェデレーションを使用できないケースがあります。例えば次のケースです。

- 侵害を受けた ID プロバイダー (IdP) に関連する停止状態
- 設定ミスや人的エラーに起因する、フェデレーションアクセス管理システムの障害
- 分散型サービス拒否 (DDoS) イベントやシステムがレンダリング不可となるなどの悪意あるアクティビティ

上記のケースでは、緊急 break glass アクセス設定により、インシデントの調査とタイムリーな修復を許可する必要があります。適切な許可を持つユーザー、グループ、ロールを使用してタスクを実行し、AWS リソースにアクセスすることをお勧めします。ルートユーザーは、ルートユーザー認証情報が必要なタスクのみに使用します。インシデント対応者が AWS と他の関連システムへの適切なレベルのアクセス権を持っていることを検証するには、専用のアカウントへの事前プロビジョニングをお勧めします。このアカウントには特権アクセスが必要で、アカウントは厳格に制御、監視されなければなりません。このアカウントは、必要なタスクの実行で要求される最小特権で構成しなければなりません。アクセス権のレベルは、インシデント管理計画の一環として作成されたプレイブックに基づいている必要があります。

ベストプラクティスとして、特定の目的のための専用のユーザーとロールを使用します。IAM ポリシーの追加によりユーザーまたはロールアクセスを一時的に昇格させると、インシデント対応中にユーザーがどのアクセス権を持っていたかが明確でなくなり、昇格された権限が取り消されないリスクが生じます。

できるだけ多くの依存関係を削除し、できるだけ多くの障害シナリオでアクセスが可能になることを検証することが重要です。そのためには、インシデント対応ユーザーが、専用のセキュリティアカウントでユーザーとして作成されており、既存のフェデレーションまたはシングルサインオン (SSO) ソリューションにより管理されていないことを検証するためのプレイブックを作成します。個々のインシデント対応者は、自分の名前が付いたアカウントを持つ必要があります。アカウント設定では、強力なパスワードポリシーと多要素認証 (MFA) を適用する必要があります。インシデント対応プレイブックで AWS マネジメントコンソールへのアクセスのみが要求されている場合、そのユーザーのアクセスキーが設定されてはならず、アクセスキー作成を明示的に禁止する必要があります。これは IAM ポリシーまたはサービスコントロールポリシー (SCP) で設定できます。詳細は、「AWS Security Best Practices for AWS Organizations SCPs」に記載されています。ユーザーは、他のアカウントのインシデント対応ロールを引き受ける以外の権限を持つべきではありません。

インシデント対応中、調査、修復、または復旧アクティビティをサポートするためのアクセス権を社内または社外の他の個人に付与する必要が生じる可能性があります。この場合、前述のプレイブックメカニズムを使用します。また、インシデント完結後直ちに追加のアクセス権を取り消すためのプロセスが必要です。

インシデント対応ロールの使用が適切に監視および監査されていることを検証するには、この目的のために作成された IAM ユーザーアカウントが個人間で共有されないようにすること、および特定のタスクで必要な場合を除き、AWS アカウントのルートユーザー が使用されないようにすることが不可欠です。ルートユーザーが必要な場合 (例えば、特定のアカウントへの IAM アクセスが利用できない場合) は、用意されたプレイブックに従って別個のプロセスを使用し、ルートユーザーのサインイン認証情報と MFA トークンの使用の可否を検証します。

インシデント対応ロールの IAM ポリシーを設定するには、IAM Access Analyzer を使用して AWS CloudTrail ログに基づいてポリシーを生成することを検討してください。そのためには、非本番アカウントのインシデント対応ロールに管理者アクセス権を付与し、プレイブックを一とおり実行します。完了したら、実行されたアクションのみを許可するポリシーを作成できます。このポリシーは、すべてのアカウントのすべてのインシデント対応ロールに適用できます。各プレイブックについて個別の IAM ポリシーを作成すると、管理と監査が容易になるでしょう。プレイブックの例には、ランサムウェア、データ侵害、本番環境へのアクセス不可、その他のシナリオについての対応計画が含まれています。

インシデント対応アカウントを使用して、別の AWS アカウントのインシデント対応専用の IAM ロールを引き受けます。これらのロールは、セキュリティアカウントのユーザーのみが引き受け可能なように設定する必要があります。信頼関係では、呼び出しプリンシパルが MFA を使用して認証されたことを要求する必要があります。ロールは、スコープが厳密に定義された IAM ポリシーを使用してアクセスを制御する必要があります。すべての `AssumeRole` リクエストが CloudTrail に記録さ

れ、アラートが送信されるようにします。また、これらのロールを使用して実行されたアクションがログに記録されるようにします。

IAM アカウントと IAM ロールの両方を CloudTrail ログで見つけやすくするために、これらに明快な名前を付けることを強くお勧めします。例えば、IAM アカウントに *<USER_ID>*-BREAK-GLASS、IAM ロールに BREAK-GLASS-ROLE という名前を付けます。

CloudTrail を使用して、AWS アカウントの API アクティビティをログに記録します。また、インシデント対応ロールの使用状況に関するアラートを設定するために使用する必要があります。ルートキーを使用する際のアラートの設定に関するブログ記事を参照してください。インストラクションに変更を加えて、Amazon CloudWatch メトリクスフィルターをインシデント対応 IAM ロールに関連する AssumeRole イベントに対して設定できます。

```
{ $.eventName = "AssumeRole" && $.requestParameters.roleArn =
 "<INCIDENT_RESPONSE_ROLE_ARN>" && $.userIdentity.invokedBy NOT EXISTS && $.eventType !
= "AwsServiceEvent" }
```

インシデント対応ロールは高いレベルのアクセス権を持っている可能性があるため、これらのアラートは幅広いグループに送信され、すみやかに対応が取られることが重要です。

インシデント対応中、対応者は、IAM によって直接保護されていないシステムへのアクセスが必要となる可能性があります。これには Amazon Elastic Compute Cloud インスタンス、Amazon Relational Database Service データベース、Software-as-a-Service (SaaS) プラットフォームが含まれます。SSH や RDP などのネイティブプロトコルではなく、AWS Systems Manager Session Manager を使用して Amazon EC2 インスタンスへの管理アクセスを行うことを強くお勧めします。このアクセスは、IAM を使用して制御できます。それにより安全が確保され、監査が行われます。また、AWS Systems Manager Systems Manager Run Command ドキュメントを使用してプレイブックの一部を自動化することも可能です。それにより、ユーザーのエラーを減らし、復旧にかかる時間を短縮できます。データベースとサードパーティーツールへのアクセスでは、アクセス認証情報を AWS Secrets Manager に保管し、インシデント対応者ロールにアクセス権を付与することをお勧めします。

最後に、インシデント対応 IAM ユーザーアカウントの管理は、Joiners、Movers、および Leavers プロセスに追加し、定期的にテストして、意図されたアクセスのみが許可されていることを検証する必要があります。

リソース

関連ドキュメント:

- Managing temporary elevated access to your AWS environment
- AWS セキュリティインシデント対応ガイド
- AWS Elastic Disaster Recovery
- AWS Systems Manager Incident Manager
- IAM ユーザー用のアカウントパスワードポリシーの設定
- AWS での多要素認証 (MFA) の使用
- クロスアカウントアクセス用に MFA を設定する
- IAM Access Analyzer を使用した IAM ポリシーの設定
- Best Practices for AWS Organizations Service Control Policies in a Multi-Account Environment
- How to Receive Notifications When Your AWS Account's Root Access Keys Are Used
- Create fine-grained session permissions using IAM managed policies
- Break glass access

関連動画:

- Automating Incident Response and Forensics in AWS
- DIY guide to runbooks, incident reports, and incident response
- Prepare for and respond to security incidents in your AWS environment

SEC10-BP06 ツールを事前デプロイする

復旧までの調査時間を短縮できるように、セキュリティ担当者は適切なツールを事前にデプロイしておきます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

セキュリティ対応と運用機能を自動化するために、AWS の包括的な API とツールセットを使用できます。ID 管理、ネットワークセキュリティ、データ保護、モニタリング機能を完全に自動化し、既に導入されている一般的なソフトウェア開発方法を使用して提供できます。セキュリティオートメーションを構築すれば、担当者がセキュリティ上の位置づけを監視し、イベントに手動で応答する代わりに、システムが監視、レビューを行い応答を開始できます。

インシデント対応チームが同じ方法でアラートに対応し続けると、アラート疲れになるリスクがあります。時間の経過とともに、チームはアラートに対する感度が鈍くなり、通常の状況の処理で間違いを犯したり、異常なアラートを見逃したりする可能性があります。自動化を利用すれば、繰り返し発生する通常のアラートを処理する機能を使用してアラート疲れを回避し、機密性の高いインシデントや独自のインシデントの処理を人間に任せることができます。Amazon GuardDuty、AWS CloudTrail Insights、および Amazon CloudWatch Anomaly Detection などの異常の検出システムを統合することで、よくあるしきい値ベースのアラートの負担を減らすことができます。

プロセス内のステップをプログラムで自動化すれば、手動プロセスを改善できます。イベントに対する修復パターンを定義したら、そのパターンを実行可能なロジックに分解して、そのロジックを実行するコードを記述できます。その後、対応者は、そのコードを実行して問題を修正します。時間の経過とともに、より多くのステップを自動化し、最終的には一般的なインシデントのクラス全体を自動的に処理できるようになります。

セキュリティ調査中、インシデントの全容とタイムラインを記録して理解するために、関連ログを確認できる必要があります。ログはまた、関心のある特定のアクションが発生したことを示すアラート生成にも必須です。クエリと取得のメカニズムとアラートを選択、有効化、保存、セットアップし、アラート発行を設定することが非常に重要となります。さらに、ログデータを検索するツールとして、Amazon Detective が有効です。

AWS では、200 を超えるクラウドサービスと数千の機能を提供しています。インシデント対応戦略をサポートし、簡素化できるサービスを確認することをお勧めします。

ログ記録に加えて、タグ付け戦略を策定して実装する必要があります。タグ付けを行うことで、AWS リソースの目的についての背景情報を付け加えることができます。タグ付けは自動化にも使用できます。

実装手順

分析とアラート発行のためのログを選択して設定する

インシデント対応のログ記録の設定については、次のドキュメントを参照してください。

- Logging strategies for security incident response
- SEC04-BP01 サービスとアプリケーションのログ記録を設定する

検出と対応をサポートするセキュリティサービスを有効にする

AWS は検出、予防、対応の機能を備えているほか、カスタムセキュリティソリューションの構築に使用できるサービスも提供しています。セキュリティインシデント対応に最も関連性の高いサービ

スのリストについては、「クラウド機能の定義」および「Security Incident Response のホームページ」を参照してください。

タグ付け戦略を策定し、実装する

AWS リソースを取り巻くビジネスユースケースやかかわりのある内部関係者についての背景情報の入手は難しい場合があります。これを達成する方法の 1 つとして、タグを使用して、ユーザー定義のキーと値で構成されるメタデータを AWS リソースに割り当てる方法があります。タグを作成して、目的、所有者、環境、処理されるデータの種類など、任意の基準でリソースを分類できます。

一貫したタグ付け戦略があると、AWS リソースに関する背景情報をすばやく特定、識別できるため、応答時間を短縮し、組織の背景情報の把握に費やす時間を最小限に抑えることができます。タグは、対応の自動化を開始するためのメカニズムとしても機能します。タグ付けする対象の詳細については、「Tagging your AWS resources」を参照してください。まず、組織全体に導入するタグを定義する必要があります。その後、このタグ付け戦略を導入し、適用します。導入と適用の詳細については、「Implement AWS resource tagging strategy using AWS Tag Policies and Service Control Policies (SCPs)」を参照してください。

リソース

関連する Well-Architected のベストプラクティス:

- SEC04-BP01 サービスとアプリケーションのログ記録を設定する
- SEC04-BP02 標準化した場所にログ、検出結果、メトリクスを取り込む

関連ドキュメント:

- Logging strategies for security incident response
- Incident response cloud capability definitions

関連する例:

- Amazon GuardDuty と Amazon Detective を使用した脅威の検出と対応
- Security Hub Workshop
- Vulnerability Management with Amazon Inspector

SEC10-BP07 シミュレーション行う

組織が成長し進化するにつれて、脅威の状況も変化するため、インシデント対応能力を継続的に見直すことが重要になります。この評価を行う方法の 1 つとして、シミュレーション (ゲームデーとも呼ばれる) の実施があります。シミュレーションでは、脅威アクターの戦術、手法、手順 (TTP) を模倣するように設計された現実のセキュリティイベントシナリオを使用します。これにより、組織は実際に発生する可能性のある模擬サイバーイベントに対応することで、インシデント対応能力を訓練し、評価できます。

このベストプラクティスを活用するメリット: シミュレーションにはさまざまな利点があります。

- サイバー脅威への準備状況を検証し、インシデント対応者の信頼度を高めます。
- ツールとワークフローの精度と効率性をテストします。
- インシデント対応計画に沿うように、コミュニケーションとエスカレーションの方法を改良します。
- あまり一般的でないベクトルに対応する機会を提供します。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

シミュレーションには主に 3 つのタイプがあります。

- 机上演習: 机上でのシミュレーションは、インシデントに対応するさまざまな利害関係者が参加して役割や責任を実践し、確立されたコミュニケーションツールやプレイブックを活用するディスカッションベースのセッションです。演習は、通常はバーチャル会場、実際の施設、またはそれらの組み合わせが可能で、丸 1 日かけて進行します。ディスカッションベースのため、机上演習ではプロセス、人材、コラボレーションに焦点を当てます。テクノロジーは議論に不可欠ですが、インシデント対応ツールやスクリプトを実際に使用することは、一般的に机上演習には含まれません。
- パープルチーム演習: パープルチーム演習は、インシデント対応者 (ブルーチーム) と模擬の脅威アクター (レッドチーム) のコラボレーションレベルを高めるものです。ブルーチームはセキュリティオペレーションセンター (SOC) のメンバーで構成されますが、実際のサイバーイベントに関与する他の利害関係者が参加することもあります。レッドチームは、攻撃的なセキュリティのトレーニングを受けたペンテストチームまたは主な利害関係者で構成されています。レッドチームは、シナリオが正確で実現可能なものになるように、演習のファシリテーターと協力して作業しま

す。パープルチーム演習では、インシデント対応の取り組みを支援する検出メカニズム、ツール、標準運用手順 (SOP) に重点が置かれます。

- レッドチーム演習: レッドチーム演習では、攻撃側 (レッドチーム) は、あらかじめ決められた範囲から、ある目的または一連の目的を達成するためのシミュレーションを行います。防御側 (ブルーチーム) は、演習の範囲と期間について必ずしも知識を持っているとは限らないため、実際のインシデントにどのように対応するか、より現実的に評価できます。レッドチーム演習では侵入テストになる可能性があります。そのため慎重に行い、コントロールを実施して、演習によって環境に実害を与えないことを確認してください。

定期的にサイバーシミュレーションを実施することを検討してください。演習は、タイプごとにそれぞれのメリットを参加者と組織全体にもたらします。それほど複雑ではないタイプのシミュレーション (机上演習など) から始めて、より複雑なシミュレーションタイプ (レッドチーム演習) に進むこともできます。セキュリティの成熟度、リソース、目標とする成果に基づいてシミュレーションタイプを選択する必要があります。お客様によっては、複雑さやコスト面から、レッドチーム演習を選択しない場合があります。

実装手順

選択したシミュレーションタイプにかかわらず、シミュレーションは通常、以下のような実施手順に従います。

1. 演習の中核要素の定義: シミュレーションのシナリオと目的を定義します。いずれも、リーダーの承認が必要です。
2. 主な利害関係者の特定: 少なくとも、演習には演習のファシリテーターと参加者が必要です。シナリオによっては、追加で法務、コミュニケーション、経営幹部などの利害関係者が関与する場合があります。
3. シナリオの構築とテスト: 特定の要素が実現不可能な場合は、シナリオの構築中に再定義が必要なこともあります。このステージのアウトプットとして、シナリオの最終版が完成することが期待されます。
4. シミュレーションの進行: シミュレーションのタイプによって、使用する進行内容 (紙ベースのシナリオと技術的に高度なシミュレーションシナリオの比較) が決まります。ファシリテーターは、演習進行の戦略を目的に合わせて調整し、最大の効果が得られるように、できるだけすべての参加者に演習に参加してもらう必要があります。
5. アフターアクションレビュー (AAR) の作成: うまくいった部分、改善の余地がある部分、潜在的なギャップを特定します。AAR では、シミュレーションの有効性だけでなく、シミュレートされ

たイベントに対するチームの反応も測定して、今後のシミュレーションの進捗を経時的に追跡できるようにする必要があります。

リソース

関連ドキュメント:

- AWS Incident Response Guide

関連動画:

- AWS GameDay - Security Edition
- 効果的なセキュリティインシデント対応シミュレーションの実行

オペレーション

インシデント対応の実施では、オペレーションが中核となります。ここで、セキュリティインシデントへの対応と修復が行われます。オペレーションには、検出、分析、封じ込み、根絶、復旧 の 5 つのフェーズが含まれます。これらのフェーズと目標の説明は、次の表にあります。

[Phase] (フェーズ)	目標
検出	潜在的なセキュリティイベントを特定します。
分析	セキュリティイベントがインシデントかどうかを判断し、インシデントの範囲を評価します。
封じ込み	セキュリティイベントの範囲を最小限に抑え、制限します。
根絶	セキュリティイベントに関連する不正なリソースやアーティファクトを削除します。セキュリティインシデントの原因となった緩和策を実装します。

[Phase] (フェーズ)	目標
復旧	システムを既知の安全な状態に復元し、これらのシステムを監視して脅威が再発しないことを確認します。

これらのフェーズは、効果的かつ堅牢な方法で対応するために、セキュリティインシデントに対応して運用する際の指針となるはずです。実際に実行するアクションは、インシデントによって異なります。例えば、ランサムウェアが関係するインシデントは、パブリック Amazon S3 バケットに関連するインシデントとは異なる対応手順を踏む必要があります。さらに、これらのフェーズは必ずしも連続して発生するわけではありません。封じ込みおよび根絶後は、分析に戻って対策が効果的だったかどうかを把握する必要があるかもしれません。

人員、プロセス、テクノロジーを綿密に準備することが、業務を効果的に行う鍵となります。したがって、準備セクションのベストプラクティスに従って、アクティブなセキュリティイベントに効果的に対応できるようにします。

詳細については、「AWS セキュリティインシデント対応ガイド」の「オペレーション」セクションを参照してください。

インシデント後のアクティビティ

脅威の状況は絶えず変化しているため、環境を効果的に保護するためには、組織の能力も同様に動的なものにすることが重要です。継続的な改善の鍵は、インシデントとシミュレーションの結果を反復することで、想定されるセキュリティインシデントを効果的に検出、対応、調査する能力を向上させることです。これにより、想定される脆弱性や、対応までの時間を短縮し、安全なオペレーションに復帰することができます。以下のメカニズムは、組織がどのような状況でも効果的に対応するための最新の能力と知識を十分に備えていることを確認するのに役立ちます。

ベストプラクティス

- SEC10-BP08 インシデントから学ぶためのフレームワークを確立する

SEC10-BP08 インシデントから学ぶためのフレームワークを確立する

教訓フレームワークと根本原因分析プロセスを導入することは、インシデント対応能力の向上だけでなく、インシデントの再発防止にも役立ちます。各インシデントから学ぶことで、同じ失敗、露出、

設定ミスの繰り返しを防ぐことができ、セキュリティ体制が強化されるだけでなく、予防できたはずの状況に無駄にする時間を最小限に抑えることができます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

以下の点を大まかに確立して達成する教訓フレームワークを実装することが重要です。

- 事後検証会を実施するタイミング
- 事後検証会を通して行うこと
- 事後検証会の実施方法
- そのプロセスに関わる人物、またかかわり方
- 改善の余地がある領域の特定方法
- 改善事項を効果的に追跡、実装する方法

フレームワークは、個人に焦点を当てたり非難したりするのではなく、ツールやプロセスの改善に焦点を当てるべきです。

実装手順

前述の大局的な成果とは別に、プロセスから最大の価値 (実行可能な改善につながる情報) を引き出すためには、適切な質問を行うことが重要です。教訓についての議論を進めるうえで役立つ質問には次のようなものがあります。

- どのようなインシデントでしたか。
- インシデントが最初に特定されたのはいつでしたか。
- どのようにして特定されましたか。
- どのシステムからアクティビティについてのアラートが発行されましたか。
- どのようなシステム、サービス、データが関与しましたか。
- 具体的に何が起きましたか。
- 何がうまくいきましたか。
- 何がうまくいきませんでしたか。
- インシデントに対応できなかった、またはスケールに失敗したのはどのプロセスまたは手順ですか。
- 次の領域で改善できることは何でしょうか。

- People
  - 連絡する必要があった担当者に実際に連絡がつきましたか。また、連絡先リストの情報は最新のものでしたか。
  - インシデントに効果的に対応して調査するために必要なトレーニングや能力を欠いていましたか。
  - 適切なリソースは用意されていましたか。
- プロセス
  - 対応はプロセスと手順に従って進められましたか。
  - この (タイプの) インシデントについて、プロセスと手順が文書化され、利用可能になっていましたか。
  - 必要なプロセスや手順が欠けていましたか。
  - 対応担当者は、問題に対応するために必要な情報にタイムリーにアクセスできましたか。
- テクノロジー
  - 既存のアラートシステムは、アクティビティを効果的に特定してアラートを出しましたか。
  - どうすれば検出までの時間を 50% 短縮できたでしょうか。
  - この (タイプの) インシデントに備えて、既存のアラートを改善する、または新しいアラートを作成する必要がありますか。
  - 既存のツールでインシデントを効果的に調査 (検索/分析) できましたか。
  - この (タイプの) インシデントをより早く特定するにはどうすればよいでしょうか。
  - この (タイプの) インシデントの再発を防ぐにはどうすればよいでしょうか。
  - 改善計画の所有者は誰ですか。また、その実施状況をどのように検証しますか。
  - 追加のモニタリング、予防的統制やプロセスを導入し、テストするまでのスケジュールはどのようになっていますか。

このリストはすべてを網羅しているわけではありませんが、インシデントから最も効果的に学び、セキュリティ体制の継続的な改善に向けて、組織とビジネスのニーズを見極め、その分析方法を特定するための出発点として活用いただくことを目的としています。最も重要なのは、事後検証会を標準的なインシデント対応プロセスと文書化の一部として取り入れ、想定されるものとして関係者全員にも定着させることです。

リソース

関連ドキュメント:

- AWS Security Incident Response Guide - Establish a framework for learning from incidents
- NCSC CAF guidance - Lessons learned

アプリケーションセキュリティ

アプリケーションのセキュリティ (AppSec) は、開発するワークロードのセキュリティ特性の設計、構築、テストの各方法の全体的なプロセスを説明するものです。適切なトレーニングを受けた人材を組織に配置した上で、ビルドのセキュリティ特性を理解してインフラストラクチャをリリースし、自動化を使用してセキュリティ問題を識別する必要があります。

ソフトウェア開発ライフサイクル (SDLC) とリリース後プロセスの一部としてアプリケーションセキュリティテストを採用することで、本番環境でのアプリケーションのセキュリティ問題の特定、修正、防止のための構造的なメカニズムを確立することができます。

ワークロードを設計、構築、デプロイ、運用する際、アプリケーション開発手法にはセキュリティコントロールを含めるようにします。そのうえで、継続的な欠陥削減のプロセスと技術的負債の低減を調整します。例えば、脅威モデリングを設計フェーズで使用すると、設計上の欠陥を早期に発見することができ、後になって問題を軽減するのと比較して、欠陥の修正をより簡単に、より安価に行うことができます。

一般的に、SDLC の早期に欠陥を解決することで、コストと複雑性を抑えられます。最も簡単な問題解決の方法は、そもそも問題を抱えないことです。したがって、最初に脅威モデルを作成することで、設計段階から正しい結果に焦点を合わせることができます。AppSec プログラムが成熟するにつれて、自動化を使用してテストの数を増やしたり、ビルダーへのフィードバックの忠実性を高めたり、セキュリティレビューに必要な時間を短縮したりすることができます。これらすべてのアクションは、構築するソフトウェアの品質を改善し、機能を本稼働環境に実装するまでの時間を短縮します。

これらの実装ガイドラインは、組織と文化、パイプラインのセキュリティ、パイプライン内のセキュリティ、依存関係の管理の 4 つの領域に焦点を当てています。各領域は、実装可能な一連の原則と、ワークロードの設計、開発、ビルド、デプロイ、運用のエンドツーエンドビューを提供します。

AWS には、アプリケーションのセキュリティプログラムに対処する際に使用できる多くのアプローチがあります。これらのアプローチの一部はテクノロジーに依存し、他のアプローチはアプリケーションのセキュリティプログラムにおける人や組織にフォーカスします。

ベストプラクティス

- SEC11-BP01 アプリケーションのセキュリティに関するトレーニングを実施する
- SEC11-BP02 開発およびリリースライフサイクル全体を通じてテストを自動化する
- SEC11-BP03 定期的にペンテストを実施する
- SEC11-BP04 コードレビューを実施する

- SEC11-BP05 パッケージと依存関係のサービスを一元化する
- SEC11-BP06 ソフトウェアをプログラムでデプロイする
- SEC11-BP07 パイプラインのセキュリティ特性を定期的に評価する
- SEC11-BP08 ワークロードチームにセキュリティのオーナーシップを根付かせるプログラムを構築する

SEC11-BP01 アプリケーションのセキュリティに関するトレーニングを実施する

安全な開発と運用に関するトレーニングをチームに提供し、安全で高品質なソフトウェアの構築を支援します。この手法は、開発ライフサイクルの早期の段階でセキュリティの問題を防止、検出、修正するのに役立ちます。脅威モデリング、安全なコーディング方法、安全な設定と運用のためのサービスの使用に関するトレーニングを検討してください。セルフサービスリソースを通じてチームにトレーニングへのアクセスを提供し、継続的な改善のために定期的にフィードバックを収集します。

期待される成果: 最初からセキュリティを念頭に置いてソフトウェアを設計および構築するために必要な知識とスキルをチームに提供します。脅威モデリングと安全な開発手法に関するトレーニングを通じて、チームは潜在的なセキュリティリスクと、ソフトウェア開発ライフサイクル (SDLC) 中にそれらを軽減する方法について理解を深めることができます。セキュリティに対するこのような積極的なアプローチはチームの文化の一部となり、潜在的なセキュリティ上の問題を早期に特定し、解決できるようになります。その結果、チームは高品質で安全なソフトウェアと機能をより効率的に提供できるようになり、配信タイムラインが全体的に短縮されます。組織内には、セキュリティの所有権がすべてのビルダー間で共有されており、協力的で包括的なセキュリティ文化があります。

一般的なアンチパターン:

- セキュリティレビューまで待ち、その後、システムのセキュリティ特性を検討する。
- セキュリティに関するすべての意思決定を中央のセキュリティチームに委ねる。
- SDLC での意思決定が、組織のセキュリティに対する全体的な期待や方針との関連性が説明されていない。
- セキュリティレビュープロセスの実行が遅すぎる。

このベストプラクティスを活用するメリット:

- 開発サイクルの早い段階で、組織のセキュリティ要件に関するより良い理解を得る。

- 潜在的なセキュリティの問題をすばやく識別および修正し、機能リリースまでの時間を短縮する。
- ソフトウェアとシステムの品質の向上。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

安全で高品質なソフトウェアを構築するため、アプリケーションの安全な開発と運用に関する一般的な方法について、チームにトレーニングを提供します。この手法は、開発ライフサイクルの早期の段階でセキュリティの問題を防止、検出、修正するのに役立ちます。これにより、配信タイムラインが短縮されます。

この手法を実践するには、脅威モデリングワークショップ などの AWS リソースを使用して、脅威モデリングに関するチームのトレーニングを検討してください。脅威モデリングは、チームが潜在的なセキュリティリスクを理解し、最初からセキュリティを念頭に置いてシステムを設計するのに役立ちます。さらに、安全な開発プラクティスに関する AWS トレーニング と認定、業界、または AWS パートナートレーニングへのアクセスを提供できます。大規模な設計、開発、保護、効率的な運用に対する包括的なアプローチの詳細については、「AWS DevOps ガイダンス」を参照してください。

組織のセキュリティレビュープロセスを明確に定義して伝え、チーム、セキュリティチーム、その他の関係者の責任についての概要を説明します。セキュリティ要件を満たすためのセルフサービス型のガイダンス、コード例、テンプレートを紹介します。AWS CloudFormation、AWS Cloud Development Kit (AWS CDK) (AWS CDK) コンストラクト、およびサービスカタログなどの AWS サービスを使用して、事前に承認された安全な設定を提供し、カスタムセットアップの必要性を削減できます。

セキュリティレビュープロセスやトレーニングに関するチームからのフィードバックを定期的に収集し、このフィードバックを継続的な改善に役立てます。ゲームデーやバグバッシュキャンペーンを開催して、セキュリティ問題を特定して対処しながら、チームのスキルを向上させます。

実装手順

1. トレーニングニーズの把握: アンケート、コードレビュー、チームメンバーとのミーティングを通じて、安全な開発手法に関するチーム内の現在のスキルレベルと知識のギャップを評価します。
2. トレーニングの計画: 特定されたニーズに基づいて、脅威モデリング、安全なコーディングの実践、セキュリティテスト、安全なデプロイの実践などの関連トピックをカバーするトレーニング計画を作成します。脅威モデリングワークショップ、AWS トレーニング と認定、業界や AWS パートナートレーニングプログラムなどのリソースを活用します。

3. トレーニングの計画および提供: チームの定期的なトレーニングセッションやワークショップを計画します。これらはチームの希望や都合に合わせ、インストラクターによる指導またはセルフペースで進めることができます。学習効果を高めるために、実践的な演習や実例を推奨します。
4. セキュリティレビュープロセスの定義する: セキュリティチームやその他の関係者と協力して、アプリケーションのセキュリティレビュープロセスを明確に定義します。開発チーム、セキュリティチーム、その他の関連する関係者など、プロセスに関与する各チームまたは個人の責任を文書化します。
5. セルフサービスリソースの作成: 組織のセキュリティ要件を満たす方法を示すセルフサービスのガイダンス、コード例、テンプレートを作成します。CloudFormation、AWS CDK コンストラクト、およびサービスカタログなどの AWS サービスを使用して、事前に承認された安全な設定を提供し、カスタムセットアップの必要性を削減できます。
6. コミュニケーションとソーシャル化: セキュリティレビュープロセスと利用可能なセルフサービスリソースをチームに効果的に伝達します。これらのリソースに精通してもらうためにトレーニングセッションやワークショップを実施して、それらの使用方法を理解していることを確認します。
7. フィードバックの収集と改善: セキュリティレビュープロセスやトレーニングに関するチームからのフィードバックを定期的に収集します。このフィードバックを使用して、改善する必要がある分野を特定し、トレーニング教材、セルフサービスリソース、セキュリティレビュープロセスを継続的に改善します。
8. セキュリティ演習の実施: ゲームデーまたはバグバッシュキャンペーンを企画して、アプリケーション内のセキュリティ問題を特定して対処します。これらの演習は、潜在的な脆弱性を発見するだけでなく、チームで安全な開発と運用に関するスキルを向上させる実践的な学習機会となります。
9. 継続的な学習と改善: チームメンバーに対し、最新の安全な開発手法、ツール、テクニックを習得し、学習を継続するように促します。進化を続けるセキュリティの状況とベストプラクティスを反映させるため、トレーニング教材やリソースを定期的にレビューし、更新してください。

リソース

関連するベストプラクティス:

- SEC11-BP08 ワークロードチームにセキュリティのオーナーシップを根付かせるプログラムを構築する

関連ドキュメント:

- AWS トレーニング と認定
- How to think about cloud security governance
- 脅威モデリングへのアプローチ方法
- Accelerating training – The AWS Skills Guild
- AWS DevOps Sagas

関連動画:

- Proactive security: Considerations and approaches

関連する例:

- 脅威モデリングについてのワークショップ
- Industry awareness for developers

関連サービス:

- AWS CloudFormation
- AWS Cloud Development Kit (AWS CDK) (AWS CDK) Constructs
- 「 Service Catalog」

SEC11-BP02 開発およびリリースライフサイクル全体を通じてテストを自動化する

開発およびリリースライフサイクル全体を通じて、セキュリティ特性のテストを自動化します。自動化により、リリース前にソフトウェアの潜在的な問題を一貫して繰り返し特定することが容易になります。これにより、提供されるソフトウェアのセキュリティ問題のリスクが低減されます。

期待される成果: 自動テストの目的は、開発ライフサイクルを通じて潜在的な問題を早期に、かつ頻繁にプログラムで検出する方法を提供することです。リグレッションテストを自動化すると、テスト済みのソフトウェアに変更を加えた後も、そのソフトウェアが期待どおりに動作することを確認するための機能テストおよび非機能テストを実行できます。機能していない認証、または不足している認証など、一般的な設定ミスをチェックするためのセキュリティユニットテストを定義すると、開発プロセスの初期にこれらの問題を特定し、修正できます。

テストの自動化では、アプリケーションの要件と目的とする機能性に基づいた、アプリケーション検証の目的別テストケースを使用します。自動化テストの結果は、生成されたテスト結果とそれぞれの目的とする成果の比較に基づいており、テストのライフサイクル全体を迅速化します。リグレッションテストやユニットテストスイートなどのテスト手法は、自動化に最適です。セキュリティ特性のテストを自動化することで、ビルダーはセキュリティレビューを待つことなく、自動化されたフィードバックを取得することができます。静的または動的なコード分析形式の自動化テストは、コード品質を改善し、開発ライフサイクルの初期における潜在的なソフトウェアの問題の検出に役立ちます。

一般的なアンチパターン:

- 自動化テストのテストケースおよび結果のコミュニケーションが欠如している。
- リリース直前にのみ自動化テストを実施する。
- 自動化テストケースの要件を頻繁に変更する。
- セキュリティテストの結果の対処方法に関するガイダンスが欠如している。

このベストプラクティスを活用するメリット:

- システムのセキュリティ特性を評価するチームへの依存が低減します。
- 複数のワークストリームにわたる一貫した検出結果により、一貫性が向上します。
- 本稼働ソフトウェアでセキュリティ問題が発生する可能性が低減されます。
- ソフトウェアの問題を早期に検出することで、検出から修正までの時間が短縮されます。
- システム的な動作、または複数のワークストリームにわたって繰り返される動作の可視性が向上し、組織全体で改善が促進されます。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

ソフトウェアを構築する際は、アプリケーションのビジネスロジックに基づいた機能性要件と、アプリケーションの信頼性、パフォーマンス、セキュリティに重点を置いた非機能性要件の両方をテストする、ソフトウェアテストのさまざまなメカニズムを採用します。

静的アプリケーションセキュリティテスト (SAST) は、ソースコードの異常なセキュリティパターンを分析し、欠陥が発生しやすいコードを検出します。SAST はさまざまな既知のセキュリティ問題のテストにおいて、ドキュメント (要件仕様、設計文書、設計仕様) やアプリケーションのソースコードなどの静的なインプットに依存します。静的コードアナライザーは、大規模なコードの迅速な分析

に役立ちます。NIST Quality Group は、Source Code Security Analyzer の比較を提供します。これには、Byte Code Scanner と Binary Code Scanner 用のオープンソースツールが含まれています。

潜在的な想定外の動作を識別するために、実行中のアプリケーションでテストを実施する、動的分析セキュリティテスト (DAST) 手法によって静的テストを補完します。動的テストは、静的分析では検出できない潜在的な問題の検出に使用できます。コードリポジトリ、ビルド、パイプラインステージでテストを実施することで、コード内にあるさまざまな種類の潜在的な問題をチェックできます。Amazon Q Developers は、ビルダーの IDE で、セキュリティスキャンなどのコードレコメンデーションを提供します。Amazon CodeGuru Security は、アプリケーション開発中の重大な問題、セキュリティ問題、見つけにくいバグを特定し、コード品質を向上させるための推奨事項を提示します。ソフトウェア部品表 (SBOM) の抽出では、ソフトウェアの構築に使用されるさまざまなコンポーネントの詳細と関係を含む正式なレコードを抽出することもできます。これにより、脆弱性管理を通知し、ソフトウェアまたはコンポーネントの依存関係とサプライチェーンのリスクを迅速に把握できます。

Security for Developers ワークショップでは、AWS CodeBuild、AWS CodeCommit、AWS CodePipeline などの AWS 開発者ツールを使用して、SAST および DAST テスト手法を含むリリースパイプラインの自動化を行います。

SDLC を進める中で、セキュリティチームと共に定期的なアプリケーションレビューを含む反復プロセスを確立します。リリース準備レビューの一環として、これらのセキュリティレビューで得たフィードバックに対処し、検証します。これらのレビューにより、アプリケーションの堅固なセキュリティを確立し、潜在的な問題に対処するための実行可能なフィードバックをビルダーに提供できます。

実装手順

- セキュリティテストを含む、一貫した IDE、コードレビュー、CI/CD ツールを実装します。
- 修正が必要な問題を単にビルダーに伝えるのではなく、SDLC のどこでパイプラインをブロックするのが適切かを考慮します。
- Automated Security Helper (ASH) は、オープンソースのコードセキュリティスキャンツールの例です。
- デベロッパー IDE と統合された Amazon Q Developer や、コミット時にコードをスキャンするための Amazon CodeGuru Security などの自動ツールを使用してテストまたはコード分析を実行すると、ビルダーは適切なタイミングでフィードバックを得ることができます。
- AWS Lambda を使用して構築する場合は、Amazon Inspector を使用して関数内のアプリケーションコードをスキャンできます。

- 自動化テストを CI/CD パイプラインに含める際は、ソフトウェアの問題の通知と修正を追跡するチケットシステムを使用します。
- 検出結果を生成するセキュリティテストでは、修正のガイダンスをリンクすることで、ビルダーのコード品質改善を支援します。
- 自動化ツールからの検出結果を定期的に分析し、次の自動化、ビルダートレーニング、啓発活動の優先順位付けに役立てます。
- CI/CD パイプラインの一部として SBOM を抽出するには、Amazon Inspector SBOM Generator を使用して、アーカイブ、コンテナイメージ、ディレクトリ、ローカルシステム、コンパイルされた Go と Rust バイナリの SBOM を CycloneDX SBOM 形式で作成します。

リソース

関連するベストプラクティス:

- DevOps Guidance: DL.CR.3 Establish clear completion criteria for code tasks

関連ドキュメント:

- Continuous Delivery and Continuous Deployment

- AWS DevOps コンピテンシーパートナー
- AWS Security Competency Partners for Application Security
- Choosing a Well-Architected CI/CD approach
- Secrets detection in Amazon CodeGuru Security
- Amazon CodeGuru Security Detection Library
- Accelerate deployments on AWS with effective governance
- How AWS approaches automating safe, hands-off deployments
- How Amazon CodeGuru Security helps you effectively balance security and velocity

関連動画:

- Hands-off: Automating continuous delivery pipelines at Amazon
- Automating cross-account CI/CD pipelines

- The Software Development Prcess at Amazon
- Testing software and systems at Amazon

関連する例:

- Industry awareness for developers
- Automated Security Helper (ASH)
- AWS CodePipeline Governance - Github

SEC11-BP03 定期的にペンテストを実施する

定期的にソフトウェアのペンテストを実施します。このメカニズムは、自動化されたテストや手動のコードレビューでは検出できない、ソフトウェアの潜在的な問題の特定に役立ちます。また、検出統制の効率を理解する上でも役立ちます ペンテストでは、保護する必要があるデータを公開する、想定よりも広範なアクセス許可を付与するなど、ソフトウェアを予期しない方法で実行できるかどうかの確認を試行します。

期待される成果: ペンテストは、アプリケーションのセキュリティ特性を検出、修正、検証するために使用します。ソフトウェア開発ライフサイクル (SDLC) の一環として、定期的かつ計画的にペンテストを実施します。ペンテストでの検出結果は、ソフトウェアのリリース前に対処する必要があります。ペンテストでの検出結果を分析し、自動化によって検出できる問題があるかどうかを確認します。アクティブなフィードバックメカニズムを含む、定期的で反復可能なペンテストを実施することで、ビルダーへのガイダンスの提供やソフトウェア品質の向上に役立ちます。

一般的なアンチパターン:

- 既知のセキュリティの問題、または広く発生しているセキュリティの問題に対してのみペンテストを実施する。
- 依存するサードパーティーツールやライブラリを除いてアプリケーションのペンテストを実施する。
- 実装されたビジネスロジックを評価せずに、パッケージセキュリティの問題のみについてペンテストを実施する。

このベストプラクティスを活用するメリット:

- リリース前のソフトウェアのセキュリティ特性に関する信頼性を向上させます。

- 望ましいアプリケーションパターンを特定する機会を創出して、ソフトウェアの品質をさらに向上させます。
- 開発サイクルの早期に、ソフトウェアのセキュリティ特性を改善するための自動化や、追加のトレーニングを特定するフィードバックループを確立します。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

ペンテストは、計画されたセキュリティ侵入シナリオを実行して、セキュリティコントロールを検出、修正、検証する、構造化されたセキュリティテストです。ペンテストは、アプリケーションの現在の設計とその依存関係に基づいてデータを収集する調査から始まります。セキュリティに特化した、厳選されたテストシナリオの一覧を作成し、実施します。これらのテストの主な目的は、環境への意図しないアクセスやデータへの不正アクセスに悪用される可能性のある、アプリケーションのセキュリティ問題を発見することです。新しい機能をリリースしたり、機能や技術的な実装の大きな変更をアプリケーションに加えたりした場合は、必ずペンテストを実施する必要があります。

ペンテストを実施するのに最適な開発ライフサイクルのステージを特定します。このテストは、システムの機能がリリース間近の状態で、なおかつ問題の修正に十分な時間を確保できる時期に行う必要があります。

実装手順

- ペンテストの範囲を設定するための構造化されたプロセスを用意し、このプロセスを脅威モデルに基づいて行うことは、コンテキストを維持するための良い方法です。
- ペンテストの実施に最適な開発ライフサイクルの時期を特定します。これは、アプリケーションで大きな変更が予定されておらず、問題の修正に十分な時間を確保できる時期である必要があります。
- ペンテストから期待される検出結果、および問題の修正に関する情報の取得について、ビルダーへのトレーニングを実施します。
- 一般的、または反復的なテストを自動化するためのツールを使用して、ペンテストプロセスの時間を短縮します。
- ペンテストでの検出結果を分析してシステム的なセキュリティの問題を特定し、このデータを使用して、追加の自動化テストと継続的なビルダー教育に役立てます。

リソース

関連するベストプラクティス:

- SEC11-BP01 アプリケーションのセキュリティに関するトレーニングを実施する
- SEC11-BP02 開発およびリリースライフサイクル全体を通じてテストを自動化する

関連ドキュメント:

- AWS Penetration Testing provides detailed guidance for penetration testing on AWS
- Accelerate deployments on AWS with effective governance
- AWS セキュリティコンピテンシーパートナー
- Modernize your penetration testing architecture on AWS Fargate
- AWS Fault Injection Simulator

関連する例:

- Automate API testing with AWS CodePipeline (GitHub)
- Automated security helper (GitHub)

SEC11-BP04 コードレビューを実施する

コードレビューを実装して、開発中のソフトウェアの品質とセキュリティを検証します。コードレビューでは、元のコード作成者以外のチームメンバーがコードをレビューし、潜在的な問題、脆弱性、コーディング標準とベストプラクティスへの準拠を確認します。このプロセスは、元のデベロッパーが見落とした可能性のあるエラー、不整合、セキュリティ上の欠陥を検出するのに役立ちます。自動ツールを使用してコードレビューを支援します。

期待される成果: 開発中にコードレビューを含め、記述されるソフトウェアの品質を向上させます。コードレビュー中に特定された学習を通じて、経験の浅いチームメンバーをスキルアップさせます。自動化の機会を特定し、自動化ツールとテストを使用してコードレビュープロセスをサポートします。

一般的なアンチパターン:

- デプロイ前にコードを確認しない。

- コードの作成とレビューを同じ担当者が行っている。
- コードレビューの支援と調整に自動化を使用していない。
- コードレビューの前に、アプリケーションセキュリティについてビルダーをトレーニングしていない。

このベストプラクティスを活用するメリット:

- コード品質の向上。
- 共通のアプローチの再利用によるコード開発の一貫性の向上。
- ペンテストや後工程において検出される問題が低減する。
- チーム内での知識の移転が改善される。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

コードレビューは、開発中のソフトウェアの品質とセキュリティを検証するのに役立ちます。手動レビューでは、元のコード作成者以外のチームメンバーがコードをレビューし、潜在的な問題、脆弱性、コーディング標準およびベストプラクティスへの準拠を確認します。このプロセスは、元のデベロッパーが見落とした可能性のあるエラー、不整合、セキュリティ上の欠陥を検出するのに役立ちます。

自動コードレビューの実施には、Amazon CodeGuru Security を検討してください。CodeGuru Security は、機械学習と自動推論を使用してコードを分析し、潜在的なセキュリティの脆弱性とコーディングの問題を特定します。自動コードレビューを既存のコードリポジトリや継続的インテグレーション/継続的デプロイ (CI/CD) パイプラインと統合します。

実装手順

1. コードレビュープロセスを確立します。
   - コードをメインブランチにマージする前や本番環境にデプロイする前など、コードレビューを実行するタイミングを定義します。
   - チームメンバー、シニアデベロッパー、セキュリティエキスパートなど、コードレビュープロセスに関与するユーザーを決定します。
   - 使用するプロセスやツールなど、コードレビューの方法を決定します。
2. コードレビューツールをセットアップします。

- GitHub Pull Requests や CodeGuru Security など、チームのニーズに合ったコードレビューツールを評価して選択します。
- 選択したツールを既存のコードリポジトリと CI/CD パイプラインと統合します。
- レビュー担当者の最小数や承認ルールなどのコードレビュー要件を適用するようにツールを構成します。

3. コードレビューチェックリストとガイドラインを定義します。
   - レビュー対象の概要を示すコードレビューチェックリストまたはガイドラインを作成します。コードの品質、セキュリティの脆弱性、コーディング標準の遵守、パフォーマンスなどの要因を考慮してください。
   - チェックリストまたはガイドラインを開発チームと共有し、期待されている内容を全員が理解していることを確認します。
4. コードレビューのベストプラクティスについて、デベロッパーにトレーニングを実施します。
   - 効果的なコードレビューを行う方法について、チームにトレーニングを提供します。
   - アプリケーションセキュリティの原則と、レビュー中に確認する一般的な脆弱性についてチームに教育します。
   - 知識の共有とペアプログラミングセッションを奨励して、経験の浅いチームメンバーのスキルアップを図ります。
5. コードレビュープロセスを実装します。
   - プルリクエストの作成やレビュー担当者の割り当てなど、コードレビュー手順を開発ワークフローに統合します。
   - マージまたはデプロイの前に、コードの変更がコードレビューを受けることを必須にします。
   - レビュープロセス中のオープンなコミュニケーションと建設的なフィードバックを奨励します。
6. モニタリングと改善:
   - コードレビュープロセスの有効性を定期的に確認し、チームからフィードバックを収集します。
   - 自動化またはツールの改善の機会を見極め、コードレビュープロセスを合理化します。
   - 学習内容と業界のベストプラクティスに基づいて、コードレビューチェックリストまたはガイドラインを継続的に更新および改良します。
7. コードレビューの文化を育む:
   - コード品質とセキュリティを維持するために、コードレビューの重要性を強調します。
   - コードレビュープロセスからの学習の成果と、プロセスの成功を称えます。

- デベロッパーが安心してフィードバックを提供し、受け取ることができるような協力的で協力的な環境を奨励します。

リソース

関連するベストプラクティス:

- SEC11-BP02 開発およびリリースライフサイクル全体を通じてテストを自動化する

関連ドキュメント:

- DevOps Guidance: DL.CR.2 Perform peer review for code changes
- About pull requests in GitHub

関連する例:

- Automate code reviews with Amazon CodeGuru Security
- Automating detection of security vulnerabilities and bugs in CI/CD pipelines using Amazon CodeGuru Security CLI

関連動画:

- Continuous improvement of code quality with Amazon CodeGuru Security

SEC11-BP05 パッケージと依存関係のサービスを一元化する

ビルダーチームに対して、ソフトウェアパッケージとその他の依存関係を取得するための一元化されたサービスを提供します。これにより、記述するソフトウェアに含まれる前にパッケージの検証が可能になります。また、これは組織で使用中のソフトウェアを分析するためのデータソースとなります。

期待される成果: 作成するコードに加えて、外部ソフトウェアパッケージからワークロードを構築します。これにより、JSON パーサーや暗号化ライブラリなど、繰り返し使用される機能の実装が簡単になります。これらのパッケージと依存関係のソースを一元管理することによって、セキュリティチームが使用前に検証できるようになります。このアプローチを手動および自動化されたテストフ

ローと組み合わせて使用することによって、開発中のソフトウェアの品質についての信頼性を高めることができます。

一般的なアンチパターン:

- インターネット上の任意のリポジトリからパッケージを取得する。
- ビルダーに提供する前に、新しいパッケージをテストしていない。

このベストプラクティスを活用するメリット:

- 構築中のソフトウェアで使用されるパッケージをより良く理解できる。
- 誰が、どのようなパッケージを使用しているかを把握することで、パッケージの更新が必要な場合に、ワークロードチームに通知することができる。
- 問題のあるパッケージがソフトウェアで使用されるリスクを低減する。

このベストプラクティスを活用しない場合のリスクレベル: 中

実装のガイダンス

ビルダーが簡単に使用できるように、パッケージと依存関係の一元化されたサービスを提供します。一元化されたサービスは、実装されるモノリシックシステムとしてではなく、論理的に一元化します。このアプローチを使用することで、ビルダーのニーズに合ったサービスを提供できます。更新が発生したときや新しい要件が発生したときに、リポジトリにパッケージを追加する効率的な方法を実装する必要があります。AWS CodeArtifact や同様の AWS パートナーソリューションなどの AWS のサービスでは、この機能を提供する方法を提供しています。

実装手順

- ソフトウェアを開発しているすべての環境で利用可能な、論理的に一元化されたリポジトリサービスを実装します。
- AWS アカウントのベンディングプロセスの一部として、リポジトリへのアクセスを含めます。
- パッケージをリポジトリに公開する前に、パッケージの自動化テストを構築します。
- 最も頻繁に使用されるパッケージ、言語、および変更量が最も多いチームのメトリクスを維持します。
- 新しいパッケージのリクエストとフィードバックの提供を行うための、自動化されたメカニズムをビルダーチームに提供します。

- リポジトリ内のパッケージを定期的にスキャンして、新しく発見された問題の潜在的な影響を特定します。

リソース

関連するベストプラクティス:

- SEC11-BP02 開発およびリリースライフサイクル全体を通じてテストを自動化する

関連ドキュメント:

- DevOps ガイダンス: DL.CS.2 各ビルド後にコードアーティファクトに署名する
- Supply chain Levels for Software Artifacts (SLSA)

関連する例:

- Accelerate deployments on AWS with effective governance
- Tighten your package security with CodeArtifact Package Origin Control toolkit
- Multi Region Package Publishing Pipeline (GitHub)
- Publishing Node.js Modules on AWS CodeArtifact using AWS CodePipeline (GitHub)
- AWS CDK Java CodeArtifact Pipeline Sample (GitHub)
- Distribute private .NET NuGet packages with AWS CodeArtifact (GitHub)

関連動画:

- Proactive security: Considerations and approaches
- The AWS Philosophy of Security (re:Invent 2017)
- When security, safety, and urgency all matter: Handling Log4Shell

SEC11-BP06 ソフトウェアをプログラムでデプロイする

可能な場合は、ソフトウェアのデプロイをプログラムで行います。この手法により、デプロイに失敗したり、人為的エラーにより予期しない問題が発生したりする可能性を低減できます。

期待される成果: テストするワークロードのバージョンはデプロイするバージョンであり、デプロイは毎回一貫して実行されます。ワークロードの設定を外部化して、変更なしでさまざまな環境にデプロイできるようになります。環境間に何も変更がないことを確認するために、ソフトウェアパッケージの暗号化署名を使用します。

一般的なアンチパターン:

- 本番環境にソフトウェアを手動でデプロイする。
- 環境に合わせて手動でソフトウェアに変更を加える。

このベストプラクティスを活用するメリット:

- ソフトウェアリリースプロセスの信頼性が向上します。
- 変更に失敗してビジネスの機能性に影響を与えるリスクが低減されます。
- 変更リスクの低減により、リリース頻度が向上します。
- デプロイ中に予期しないイベントが発生した場合に自動ロールバックが機能します。
- テスト済みのソフトウェアとデプロイされたソフトウェアが同じであることを、暗号化によって証明できます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

堅牢で信頼性の高いアプリケーションインフラストラクチャを維持するには、安全で自動化されたデプロイプラクティスを実装します。このプラクティスでは、本番環境から永続的な人間のアクセスを削除し、デプロイに CI/CD ツールを使用し、環境固有の設定データを外部化します。このアプローチに従うことによって、セキュリティを強化し、ヒューマンエラーのリスクを減らし、デプロイプロセスを合理化できます。

AWS アカウント 構造を構築し、本番環境から永続的な人的アクセスを削除できます。このプラクティスは、不正な変更や偶発的な変更のリスクを最小限に抑え、本番稼働システムの整合性を向上させます。直接ヒューマンアクセスの代わりに、AWS CodeBuild や AWS CodePipeline などの CI/CD ツールを使用してデプロイを実行できます。これらのサービスを使用して、構築、テスト、デプロイのプロセスを自動化できるため、手動の介入が減り、一貫性が向上します。

セキュリティとトレーサビリティをさらに強化するために、アプリケーションパッケージがテストされ、デプロイ中にこれらの署名を検証した後に、アプリケーションパッケージに署名できます。そ

れには、AWS Signer や AWS Key Management Service (AWS KMS) などの暗号化ツールを使用します。パッケージに署名して検証することによって、認可および検証されたコードのみを環境にデプロイできます。

さらに、チームはワークロードを設計して、AWSSystems Manager Parameter Store などの外部ソースから環境固有の設定データを取得できます。この方法により、アプリケーションコードが構成データから分離されるため、アプリケーションコード自体を変更することなく、構成を個別に管理および更新できるようになります。

インフラストラクチャのプロビジョニングと管理を合理化するには、AWS CloudFormation や AWS CDK などの Infrastructure as Code (IaC) ツールを使用することを検討してください。これらのツールを使用して Infrastructure as Code (IaC) を定義できるため、さまざまな環境にわたるデプロイの一貫性と再現性が向上します。

ソフトウェアのデプロイが成功したことを検証するには、カナリアデプロイを検討します。カナリアデプロイでは、本番環境全体にデプロイする前に、インスタンスまたはユーザーのサブセットに変更をロールアウトします。その後、変更の影響をモニタリングし、必要に応じてロールバックできるため、広範な問題のリスクを最小限に抑えることができます。

「Organizing Your AWS Environment Using Multiple Accounts」ホワイトペーパーで概説されている推奨事項に従ってください。このホワイトペーパーでは、セキュリティと分離をさらに強化するため、環境 (開発、ステージング、本番など) を個別の AWS アカウント に分離する方法について説明します。

実装手順

1. AWS アカウント 構造のセットアップ:
    - 「Organizing Your AWS Environment Using Multiple Accounts」ホワイトペーパーのガイダンスに従って、異なる環境 (試験、開発、ステージング、本番稼働) 用に個別の AWS アカウント を作成します。
    - 各アカウントの適切なアクセスコントロールとアクセス許可を設定して、本番環境への直接アクセスを制限します。
2. CI/CD パイプラインの実装:
    - AWS CodeBuild や AWS CodePipeline などのサービスを使用して CI/CD パイプラインを設定します。
    - アプリケーションコードを自動的に構築、テスト、各環境にデプロイするようにパイプラインを設定します。

- コードリポジトリを CI/CD パイプラインと統合して、バージョン管理とコード管理を行います。

3. アプリケーションパッケージに署名して検証します。
   - AWS Signer または AWS Key Management Service (AWS KMS) を使用して、テストと検証後にアプリケーションパッケージに署名します。
   - ターゲット環境にデプロイする前に、アプリケーションパッケージの署名を検証するようにデプロイプロセスを設定します。
4. 設定データの外部化:
   - AWS Systems Manager パラメータストアに環境固有の設定データを保存します。
   - アプリケーションコードを変更して、デプロイ時またはランタイム中にパラメータストアから設定データを取得します。
5. Infrastructure as Code (IaC) の実装:
   - AWS CloudFormation または AWS CDK などの IaC ツールを使用して、Infrastructure as Code (IaC) を定義および管理します。
   - CloudFormation テンプレートまたは CDK スクリプトを作成して、アプリケーションに必要な AWS リソースをプロビジョニングおよび設定します。
   - IaC を CI/CD パイプラインと統合して、アプリケーションコードの変更と共にインフラストラクチャの変更を自動的にデプロイします。
6. カナリアデプロイの実装:
   - カナリアデプロイをサポートするようにデプロイメントプロセスを構成します。カナリアデプロイでは、変更がインスタンスまたはユーザーのサブセットにロールアウトされてから、運用環境全体にデプロイされます。
   - AWS CodeDeploy や AWS ECS などのサービスを使用してカナリアデプロイを管理し、変更の影響を監視します。
   - カナリアデプロイ中に問題が検出された場合は、ロールバックメカニズムを実装して以前の安定バージョンに戻します。
7. モニタリングと監査:
   - デプロイ、アプリケーションのパフォーマンス、インフラストラクチャの変更を追跡するための監視およびログ記録メカニズムを設定します。
   - Amazon CloudWatch や AWS CloudTrail などのサービスを使用して、ログやメトリクスを収集および分析します。
   - 監査とコンプライアンスチェックを実装して、セキュリティのベストプラクティスと規制要件の遵守を検証します。

8. 継続的な改善:

- 定期的にデプロイのプラクティスを確認して更新し、以前のデプロイから得たフィードバックと教訓を取り入れます。
- デプロイプロセスのできるだけ多くを自動化して、手動介入や潜在的なヒューマンエラーを減らします。
- クロスファンクショナルチーム (オペレーションやセキュリティなど) と協力して、デプロイプラクティスを調整し、継続的に改善します。

これらのステップに従うことによって、安全で自動化されたデプロイプラクティスを AWS 環境に実装できます。これにより、セキュリティが向上し、ヒューマンエラーのリスクが軽減され、デプロイプロセスが合理化されます。

リソース

関連するベストプラクティス:

- SEC11-BP02 開発およびリリースライフサイクル全体を通じてテストを自動化する
- DL.CI.2 ソースコードの変更時に自動的に構築をトリガー

関連ドキュメント:

- Accelerate deployments on AWS with effective governance
- 安全なハンズオフデプロイメントの自動化
- Code signing using AWS Certificate Manager Private CA and AWS Key Management Service asymmetric keys
- コード署名、AWS Lambda の信頼性および整合性のコントロール

関連動画:

- Hands-off: Automating continuous delivery pipelines at Amazon

関連する例:

- を使用したブルー/グリーンデプロイAWS Fargate

SEC11-BP07 パイプラインのセキュリティ特性を定期的に評価する

アクセス許可の分離に特に注意しながら、Well-Architected セキュリティの柱の原則をパイプラインに適用します。パイプラインインフラストラクチャのセキュリティ特性を定期的に評価します。パイプラインのセキュリティを効率的に管理することで、パイプラインを通過するソフトウェアのセキュリティを確保することができます。

期待される成果: ソフトウェアの構築とデプロイに使用されるパイプラインは、環境内の他のワークロードと同じ推奨プラクティスに従います。パイプラインに実装するテストは、それを使用するチームでは編集できません。パイプラインには、一時的な認証情報を使用して実行しているデプロイに必要なアクセス許可のみを付与します。パイプラインが間違った環境にデプロイされないように保護を実装します。構築環境の整合性を検証できるように、パイプラインを出力状態に設定します。

一般的なアンチパターン:

- ビルダーが回避可能なセキュリティテスト。
- デプロイパイプラインへの広範すぎるアクセス許可。
- 入力を検証するように設定されていないパイプライン。
- CI/CD インフラストラクチャに関連付けられているアクセス許可を定期的にレビューしない。
- 長期的認証情報、または強化された認証情報の使用。

このベストプラクティスを活用するメリット:

- パイプラインによって構築およびデプロイされるソフトウェアの整合性についての信頼性が向上します。
- 不審なアクティビティが存在するデプロイを停止する能力を備えることができます。

このベストプラクティスを活用しない場合のリスクレベル: 高

実装のガイダンス

デプロイパイプラインはソフトウェア開発ライフサイクルの重要なコンポーネントであり、環境内の他のワークロードと同じセキュリティ原則とプラクティスに従う必要があります。これには、適切なアクセスコントロールの実装、入力の検証、CI/CD インフラストラクチャに関連付けられたアクセス許可の定期的なレビューと監査が含まれます。

アプリケーションの構築とデプロイを担当するチームが、パイプラインに実装されているセキュリティテストとチェックを編集またはバイパスできないことを確認します。この懸念の分離は、ビルドとデプロイプロセスの完全性を維持するのに役立ちます。

まずは、AWS デプロイパイプラインリファレンスアーキテクチャ の使用を検討してください。このリファレンスアーキテクチャは、AWS で CI/CD パイプラインを構築するための安全でスケーラブルな基盤を提供します。

さらに、AWS Identity and Access Management Access Analyzer などのサービスを使用して、パイプラインアクセス許可の最小特権 IAM ポリシーを生成し、パイプラインのステップとしてワークロードアクセス許可を検証できます。これにより、パイプラインとワークロードには特定の機能に必要なアクセス許可のみがあることが確認され、不正アクセスやアクションのリスクが軽減されます。

実装手順

- AWS デプロイパイプラインリファレンスアーキテクチャから始めます。
- AWS IAM Access Analyzer を使用して、パイプラインの最小特権 IAM ポリシーをプログラムで生成することを検討してください。
- パイプラインをモニタリングやアラートと統合することで、予期しないアクティビティや異常なアクティビティの通知を受け取ることができます。AWS マネージドサービスでは、Amazon EventBridge を使用することで、AWS Lambda や Amazon Simple Notification Service (Amazon SNS) などのターゲットにデータをルーティングできます。

リソース

関連ドキュメント:

- AWS デプロイパイプラインリファレンスアーキテクチャ
- のモニタリングAWS CodePipeline
- のセキュリティに関するベストプラクティスAWS CodePipeline

関連する例:

- DevOps monitoring dashboard (GitHub)

SEC11-BP08 ワークロードチームにセキュリティのオーナーシップを根付かせるプログラムを構築する

ビルダーチームが、作成するソフトウェアに関するセキュリティの決定を行えるようにするプログラムやメカニズムを構築します。セキュリティチームは、引き続きレビュー中にこれらの決定を検証する必要がありますが、ビルダーチームにセキュリティのオーナーシップを根付かせることで、より迅速でセキュアなワークロードの構築が可能になります。このメカニズムは、構築するシステムの運用に良い影響を与える、オーナーシップのカルチャーを育みます。

期待される成果: チームにセキュリティの所有権と意思決定が組み込まれます。セキュリティについての考え方についてチームにトレーニングを実施した、あるいは埋め込みまたは関連するセキュリティ担当者でチームを補強しました。その結果、チームは開発サイクルの早い段階で、より質の高いセキュリティ上の意思決定を行えます。

一般的なアンチパターン:

- セキュリティ設計に関するすべての意思決定をセキュリティチームに委ねる。
- 開発プロセスの十分に早い段階でセキュリティ要件を策定していない。
- プログラムの運用に関して、ビルダーとセキュリティスタッフからのフィードバックを入手していない。

このベストプラクティスを活用するメリット:

- セキュリティレビューを完了するまでの時間が短縮されます。
- セキュリティのレビュー段階でようやく検出されるセキュリティ問題が低減されます。
- 作成されるソフトウェアの全体的な品質が向上します。
- システム的な問題や価値の高い改善領域を特定し、理解する機会が創出されます。
- セキュリティレビューでの検出結果に基づくやり直しが低減されます。
- セキュリティ機能に関する認識が改善されます。

このベストプラクティスを活用しない場合のリスクレベル: 低

実装のガイダンス

SEC11-BP01 アプリケーションのセキュリティに関するトレーニングを実施する のガイダンスから始めます。その後、組織に最適と思われるプログラムの運用モデルを特定します。主な 2 つのパ

ターンは、ビルダーへのトレーニングの実施、またはビルダーチームへのセキュリティスタッフの配置です。初期アプローチの決定後、組織に適したモデルであるかどうかを検証するために、単一または小規模のワークロードチームに対してテストを実施します。プログラムの実行と成功には、組織のビルダーおよびセキュリティ部門のリーダーシップによるサポートが役立ちます。このプログラムを構築する際は、プログラムの価値を測定できるメトリクスを選択することが重要です。AWS がこの課題にどのようにアプローチしたかを学ぶことは、良い学習経験になります。このベストプラクティスは、主に組織の変化と文化に重点を置いています。ビルダーとセキュリティのコミュニティ間のコラボレーションをサポートするツールを使用します。

実装手順

- アプリケーションのセキュリティに関するビルダーのトレーニングから始めます。
- ビルダー教育のためのコミュニティとオンボーディングプログラムを作成します。
- プログラムの名称を決定します。よく使用される名称は、ガーディアン、チャンピオン、アドボケイトなどです。
- ビルダートレーニング、セキュリティエンジニアの配置、セキュリティスタッフとの連携、という三択から、使用するモデルを選択します。
- セキュリティ部門、ビルダー部門、および他の潜在的な関連部門から、プロジェクトスポンサーを特定します。
- プログラムに参加する人数、レビューに要した時間、ビルダーやセキュリティスタッフからのフィードバックを測定するメトリクスを追跡します。これらのメトリクスを使用して、改善を行います。

リソース

関連するベストプラクティス:

- SEC11-BP01 アプリケーションのセキュリティに関するトレーニングを実施する
- SEC11-BP02 開発およびリリースライフサイクル全体を通じてテストを自動化する

関連ドキュメント:

- 脅威モデリングへのアプローチ方法
- How to think about cloud security governance
- セキュリティ所有権を分散するメカニズムである Security Guardians プログラムの構築方法AWS

- Security Guardians プログラムを構築してセキュリティ所有権を分散する方法

関連動画:

- Proactive security: Considerations and approaches
- AWS および Toyota Motor North America の AppSec ツールと文化のヒント

結論

セキュリティは、継続的な取り組みです。発生したインシデントは、アーキテクチャのセキュリティを向上させるための機会として扱う必要があります。強力な ID コントロール、セキュリティイベントへの自動対応化、複数レベルでのインフラストラクチャの保護、暗号化による適切に分類されたデータの管理により、あらゆる組織に実装する必要がある多層防御が可能になります。このホワイトペーパーで説明したプログラム関数と AWS の機能やサービスがあれば、このような取り組みもより簡単に実現できます。

AWS は、ビジネス価値を実現しながら、情報、システム、アセットを保護するアーキテクチャの構築と運用を支援します。

寄稿者

このドキュメントには、次の個人および組織が貢献しました。

- Amazon Web Services、Principal Solutions Architect、Jay Michael
- Amazon Web Services、Principal Security Consultant、Kiaan Sumeet
- Amazon Web Services、Principal Solutions Architect、Michael Fischer
- Amazon Web Services、Principal Solutions Architect、Conor Colgan
- Amazon Web Services、Principal Solutions Architect、Security & Compliance、Dave Walker
- Amazon Web Services、Principal Solutions Architect、Security & Compliance、Patrick Palmer
- Amazon Web Services、Security Consultant、Monka Vu Minh
- Amazon Web Services、Security Consultant、Kurt Kumar
- Amazon Web Services、Security Solutions Architect、Fahima Khan
- Amazon Web Services、Senior Security Solutions Architect、Mutaz Hajeer
- Amazon Web Services、Senior Security Solutions Architect、Luis Pastor
- Amazon Web Services、Senior Security Solutions Architect、Colin Igbokwe
- Amazon Web Services、Senior Security Solutions Architect、Geoff Sweet
- Amazon Web Services、Senior Security Solutions Architect、Anthony Harvey
- Amazon Web Services、Senior Security Solutions Architect、Sowjanya Rajavaram
- Amazon Web Services、Senior Solutions Architect、Krishna Prasad
- Amazon Web Services、Senior Solutions Architect、Faisal Farooq
- Amazon Web Services、Senior Solutions Architect、Arun Krishnaswamy
- Amazon Web Services、Senior Solutions Architect、Dan Girard
- Amazon Web Services、Senior Solutions Architect、Marc Luescher
- Amazon Web Services、Senior Technical Account Manager、Kyle Nicodemus
- Amazon Web Services、Senior Technical Account Manager、Irina Szabo
- Amazon Web Services、Solutions Architect、Arun Sivaraman
- Amazon Web Services、Technical Account Manager、Stephen Novak
- Amazon Web Services、Technical Account Manager、Jonathan Risbrook
- Amazon Web Services、Practice Manager - Global Financial Services、Freddy Kasprzykowski
- Amazon Web Services、Principal Security Consultant、Pat Gaw

- Amazon Web Services、Principal Solutions Architect、Jason Garman
- Amazon Web Services、Principal Solutions Architect、Mark Keating
- Amazon Web Services、Principal Solutions Architect、Zach Miller
- Amazon Web Services、Principal Solutions Architect、Maitreya Ranganath
- Amazon Web Services、Principal Solutions Architect、Reef Dsouza
- Amazon Web Services、Security Solutions Architect、Brad Burnett
- Amazon Web Services、Security Solutions Architecture、Senior Manager、Matt Saner
- Amazon Web Services、Senior Security Solutions Architect、Priyank Ghedia
- Amazon Web Services、Senior Security Solutions Architect、Arthur Mnev
- Amazon Web Services、Senior Security Solutions Architect、Kyle Dickinson
- Amazon Web Services、Senior Security Solutions Architect、Kevin Boland
- Amazon Web Services、Senior Security Solutions Architect、Anna McAbee
- Amazon Web Services、Senior Security Solutions Architect、Recep Meric Degirmenci
- Amazon Web Services、Senior Security Technical Product Manager、Daniel Salzedo
- Amazon Web Services、Senior Solutions Architect、Jake Izumi
- Amazon Web Services、Senior Solutions Architect、Bert Bullough
- Amazon Web Services、Solutions Architect、Robert McCall
- Amazon Web Services、AWS Enterprise Support、ESL TAM、Angela Chao
- Amazon Web Services、Senior ANZ Security Spec. Solutions Architect、Pratima Singh
- Amazon Web Services、AWS Security、Office of the CISO、Principal、Darran Boyd
- Amazon Web Services、Senior Security Solutions Architect、Byron Pogson

詳細情報

追加情報については、次の資料を参照してください。

- AWS Well-Architected フレームワークホワイトペーパー
- AWS アーキテクチャセンター

ドキュメントの改訂

このホワイトペーパーの更新に関する通知を受け取るには、RSS フィードにサブスクライブしてください。

変更	説明	日付
ベストプラクティスガイダンスの更新	ベストプラクティスは、SEC 2、SEC 3、SEC 4、SEC 6、SEC 7、SEC 8、SEC 9、SEC 10、SEC 11 の各分野における新しいガイダンスで更新されました。ガイダンスは、柱全体で更新および改良されました。	2024 年 11 月 6 日
ベストプラクティスガイダンスの更新	柱全体で大規模なベストプラクティスの更新を実施。複数のベストプラクティスを順序変更、統合。SEC 1、4、5、6、7、8、9 を大幅に変更。	2024 年 6 月 27 日
ベストプラクティスガイダンスの更新	「ワークロードを安全に運用する」および「転送中のデータの保護」のベストプラクティスガイダンスを更新。	2023 年 12 月 6 日
ベストプラクティスガイダンスの更新	「インシデント対応」のガイダンスとベストプラクティスを大幅に更新。 「準備」のベストプラクティスを更新。インシデント対応に「オペレーション」および「インシデント後の活動」を追加。「SEC10-BP08 インシ	2023 年 10 月 3 日

	デントから学ぶためのフレームワークを確立する」ベストプラクティスを追加。	
ベストプラクティスガイダンスの更新	「準備」および「シミュレート」のベストプラクティスを更新、新しいガイダンスを追加。	2023 年 7 月 13 日
新しいフレームワーク用に更新。	規範ガイダンスを使用してベストプラクティスを更新、および新しいベストプラクティスを追加。アプリケーションのセキュリティ (AppSec) の新しいベストプラクティス領域を追加。	2023 年 4 月 10 日
ホワイトペーパーの更新	新しい実装ガイダンスを使用してベストプラクティスを更新。	2022 年 12 月 15 日
ホワイトペーパーの更新	ベストプラクティスに加筆し、改善計画を追加。	2022 年 10 月 20 日
マイナーな更新	最新のベストプラクティスを反映して IAM 情報を更新。	2022 年 6 月 28 日
マイナーな更新	AWS PrivateLink 情報を追加し、壊れたリンクを修正。	2022 年 5 月 19 日
マイナーな更新	AWS PrivateLink が追加されました。	2022 年 5 月 6 日
マイナーな更新	非包括的言語を削除。	2022 年 4 月 22 日
マイナーな更新	VPC Network Access Analyzer に関する情報を追加。	2022 年 2 月 2 日
マイナーな更新	壊れたリンクを修正。	2021 年 5 月 27 日

マイナーな更新	全体を通した編集。	2021 年 5 月 17 日
メジャーな更新	ガバナンスに関するセクションを追加、さまざまなセクションに詳細を追加、全体を通して新機能やサービスを追加。	2021 年 5 月 7 日
マイナーな更新	リンクを更新しました。	2021 年 3 月 10 日
マイナーな更新	壊れたリンクを修正。	2020 年 7 月 15 日
新しいフレームワークの更新	アカウント、ID、アクセス権限の管理に関するガイダンスを更新。	2020 年 7 月 8 日
新しいフレームワークの更新	すべての領域、新しいベストプラクティス、サービス、機能のアドバイスを拡張する更新。	2020 年 4 月 30 日
ホワイトペーパーの更新	新しい AWS のサービスと機能を反映する更新、および参照の更新。	2018 年 7 月 1 日
ホワイトペーパーの更新	システムセキュリティの設定とメンテナンスのセクションを更新し、新しい AWS のサービスと機能を反映。	2017 年 5 月 1 日
初版発行	セキュリティの柱 - AWS Well-Architected フレームワークを公開。	2016 年 11 月 1 日

注意

お客様は、本書に記載されている情報を独自に評価する責任を負うものとします。本書は、(a) 情報提供のみを目的とし、(b) AWS の現行製品と慣行について説明しており、これらは予告なしに変更されることがあり、(c) AWS およびその関連会社、サプライヤー、またはライセンサーからの契約上の義務や保証をもたらすものではありません。AWS の製品やサービスは、明示または黙示を問わず、一切の保証、表明、条件なしに「現状のまま」提供されます。お客様に対する AWS の責任は AWS 契約によって規定されます。本書は、AWS とお客様との間で締結されるいかなる契約の一部でもなく、その内容を修正するものでもありません。

AWS 用語集

AWS の最新の用語については、「AWS の用語集リファレンス」の「AWS 用語集」を参照してください。