AWS Well-Architected Framework

セキュリティの柱

セキュリティの柱: AWS Well-Architected Framework

Table of Contents

セキュリティの柱 - AWS Well-Architected Framework

公開日: 2024 年 6 月 27 日 (改訂履歴)

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。 AWS Well-Architected Framework.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推奨事項を適用するうえで役立つガイダンスを提供します。

はじめに

それらの AWS Well-Architected Framework は、AWS でワークロードを構築するための意思決定におけるトレードオフの理解に役立ちます。このフレームワークを使用すれば、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的に測定し、改善点を特定することが可能となります。Well-Architected ワークロードを備えることによって、ビジネス成功の可能性が大幅に高まると私たちは確信しています。

このフレームワークは次の 6 つの柱に基づいています。

- オペレーショナルエクセレンス
- セキュリティ
- 信頼性
- パフォーマンス効率
- コスト最適化
- サステナビリティ

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。これを理解して最新の AWS の推奨事項に従うことで、ビジネス要件および規制要件を満たすことができます。この内容は、最高技術責任者 (CTO)、最高情報セキュリティ責任者 (CSO/CISO)、設計者、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にまとめられています。

このホワイトペーパーを読むことで、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ための AWS の最新の推奨事項と戦略を理解できます。ここでは、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、そのような情報に該当するリソースへの参照が記載されて

います。このホワイトペーパーにある手法を採用すれば、データとシステムを保護し、アクセスをコントロールし、セキュリティイベントに自動的に応答するアーキテクチャを構築できます。

セキュリティの基礎

セキュリティの柱は、クラウドテクノロジーを活用し、セキュリティ体制の向上を可能にするやり方でデータ、システム、資産を保護する方法を表します。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: 最小権限の原則を導入し、各 AWS リソースとの通信における適切な認可のもと、役割分担を徹底させます。ID 管理を一元化し、長期間にわたって一つの認証情報を使用し続けないようにします。
- トレーサビリティの維持: ご使用の環境に対して、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、および変更を行います。ログとメトリクスの収集をシステムに統合して、自動的に調査しアクションを実行します。
- すべてのレイヤーで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深層防御アプローチを適用します。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、すべてのレイヤーに適用します。
- セキュリティのベストプラクティスを自動化する: 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、安全で、より速く、より費用対効果の高いスケーリングが可能になります。バージョン管理されているテンプレートにおいてコードとして定義および管理されるコントロールを実装するなど、セキュアなアーキテクチャを作成します。
- 伝送中および保管中のデータを保護する: データを機密性レベルに分類し、暗号化、トークン分割、アクセスコントロールなどのメカニズムを適宜使用します。
- データに人の手を入れない: データに直接アクセスしたりデータを手動で処理したりする必要を減らしたり、排除したりするメカニズムとツールを使用します。これにより、機密性の高いデータを扱う際の誤処理、改変、ヒューマンエラーのリスクを軽減します。
- セキュリティイベントに備える: 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、ツールとオートメーションにより、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。

- セキュリティの基礎
- ID とアクセス管理
- 検知
- インフラストラクチャ保護
- データ保護
- インシデント対応
- アプリケーションのセキュリティ

責任共有モデル

セキュリティとコンプライアンスは、AWS とお客様との間で共有される責任です。この共有モデルは、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、およびコントロールするため、お客様の運用負担を軽減することができます。お客様は、AWS が提供するセキュリティグループファイアウォールの設定に加え、ゲストオペレーティングシステム (アップデートおよび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の責任と管理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、適用可能な法律および規制に応じて異なります。したがって、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。また、この責任共有モデルの性質によって柔軟性が得られ、お客様がデプロイを統制できます。下の図に示すように、この責任分担は一般に、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 は、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ついて責任を負います。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、および施設で構成されています。

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、お客様が選択した AWS クラウドサービスに応じて異なります。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 (アップデート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した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 責任共有モデル。

このお客様/AWS の責任分担モデルは IT 統制にも拡張されます。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、および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制御を管理することで、お客様の運用管理の負担を軽減することができます。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、お客様は必要に応じて、利用可能な AWS の統制とおよびコンプライアンス文書を使用して、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コントロールの例です。

継承された統制 – お客様が AWS から完全に継承するコントロール。

- 物理統制と環境統制

共有コントロール – インフラストラクチャ層とお客様層の両方に適用されるコントロール。ただしコンテキストまたは観点は異なる。共有コントロールでは、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コントロール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持ちますが、ゲストオペレーティングシステムおよびアプリケーションにパッチを適用する責任はお客様にあります。
- 設定管理 – AWS はインフラストラクチャデバイスの設定の責任を負いますが、ゲストオペレーティングシステム、データベース、およびアプリケーションの設定はお客様ご自身の責任で行います。
- 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。

カスタマー特有 – お客様が AWS のサービス内にデプロイするアプリケーションに基づき、お客様が単独で責任を負うべきコントロール。以下に例を示します。

- サービスおよびコミュニケーション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って、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 (各レイヤーが前のレイヤーの上に構築される) 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、お客様側での責任は何か、AWS から何を継承するかが明確になります。有用なリソースは AWS Artifact でAWS のセキュリティおよびコンプライアンスレポートへのオンデマンドアクセスを付与し、オンライン契約を選択します。

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こになります。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 Identity Center 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こにあります。新しい AWS サービ

スを使って開始する場合、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、検知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ています。

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、製品チームがコントロールするアプリケーションで実装が行われるためです。これは、アプリケーションでの入力確認の実施や、マイクロサービス間で ID が正しく受け渡しされるのを保証することなどが考えられます。設定を担当するのは製品チームですが、中間レイヤーから一部の機能を継承できます。

コントロールを実装する場所がどこであろうと目的は同じで、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」です。つまり 内在するリスクです.そして、可能性、結果、またはその両方を低減させる管理目標を定義することができます。そして、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かを確認することができます。つまり 残存リスクです.管理目標は 1 つまたは複数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、時間的コストにに基づいています。

可能性	リスクレベル				
非常に 可能性が高い	低	中	高	重要	重要
高い	低	中	中	高	重要
可能性がある	低	低	中	中	高
低い	低	低	中	中	高
非常に 可能性が低い	低	低		中	高
結果	最小限	低	中	高	重大

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

AWS では、社内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推奨しています。

アカウントを一元的に管理する: AWS Organizations は、AWS アカウントの作成と管理、およびアカウント作成後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 メールアドレスの検討が重要です。これがパスワードリセットを許可するルートユーザーとなるためです。Organizations は、ワークロードの要件と目的に応じた異なる環境である組織単位 (OU)でアカウントをグループ化できます。

制御を一括設定する: 特定のサービス、リージョン、サービスアクションのみを適切なレベルで許可することによって、AWS アカウントが実行できる操作を制御します。AWS Organizations では、サービスコントロールポリシー (SCP) を使用して、組織レベル、組織単位、アカウントレベルでアクセス許可ガードレールを適用できます。これは、すべての AWS Identity and Access Management (IAM) ユーザーおよびロールに適用されます。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。このサービスを使うと、AWS Organization のアカウント設定の自動化、プロビジョニングの自動化、 ガードレール (予防や検出など) の適用、ダッシュボードによる可視化を実現できます。

サービスとリソースを一括設定する: AWS Organizations では、すべてのアカウントに適用する AWS サービス を設定できます。例えば、組織全体で実行されるすべてのアクションの集中ログ記録を AWS CloudTrailで設定し、メンバーアカウントによるログの無効化を防止できます。また、AWS Config を使用して定義したルールを基にデータを一元的に集約することもできます。これによってワークロードのコンプライアンス監査と、変更への迅速な対応が可能となります。AWS CloudFormation StackSets 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 スタックを集中管理できます。これによって、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、無関係のワークロード、環境を別々のアカウントに分類し、クラウドインフラストラクチャ全体のセキュリティを向上させるアカウント構造。

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が抑えられます。
- 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が確立されてい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コンセプト、パターン、および実装に関するガイダンスについては、「Organizing Your

AWS Environment Using Multiple Accounts」(複数のアカウントを使用した AWS 環境の組織化) 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。優れた設計によりこの継承を利用して、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 および AWS Control Tower は、AWS 環境でこのマルチアカウント構造を実装および管理するのに使用できる 2 つのサービスです。AWS Organizations では、1 つまたは複数の OU 層 (それぞれに複数のメンバーアカウントを含む) で定義された階層にアカウントを組織化できます。サービス管理ポリシー (SCP) により、組織管理者がメンバーアカウントできめ細やかな予防的コントロールを確立し、AWS Config はメンバーアカウントに関して積極的かつ検出的コントロールを確立するのに使用できます。多くの AWS サービス が AWS Organizations と統合し、組織内のすべてのメンバーアカウントで、委任管理制御とサービス固有のタスク実行を提供します。

AWS Organizations 上の層にある AWS Control Tower は、ランディングゾーンにマルチアカウント AWS 環境向けの ワンクリックベストプラクティスセットアップを提供します。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 と比較して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ガードレール。
- 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ガードレール。
- AWS Control Tower Account Factory では、事前承認されたベースラインと組織内の構成オプションを含むアカウントを自動的にデプロイできます。

実装手順

1. 組織単位構造を設計する: 組織単位を適切に設計することにより、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保守に必要な管理負担を軽減できます。組織単位構造は、貴社のビジネスニーズ、データ重要度、およびワークロード構造に合致したものである必要があります。

2. マルチアカウント環境向け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一貫したセキュリティとインフラストラクチャ基盤を提供します。そこから組織はワークロードを迅速に開発、立ち上げ、デプロイできます。カスタムビルドのランディングゾーンまたは AWS Control Tower を使用して、環境のオーケストレーションを実行できます。
3. ガードレールを確立する: ランディングゾーンを通して環境に一貫性のあるセキュリティガードレールを実装します。AWS Control Tower は、必須とオプションのコントロールのリストを提供します。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、指定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 使用時はコンソール経由で、または AWS Organizations で IAM アクセス許可を調整することにより実行できます。
5. AWSCloudFormation 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しやすくなります。

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

関連するドキュメント:

- AWS Control Tower
- AWS セキュリティ監査のガイドライン
- IAM ベストプラクティス
- CloudFormation StackSets を使用して、複数の AWS アカウント とリージョン全体にリソースをプロビジョニングする
- 組織関連の FAQ
- AWS Organizations 用語およびコンセプト
- AWS Organizations マルチアカウント環境のサービスコントロールポリシーのため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
- 複数のアカウントを使用した AWS 環境の組織化

関連動画:

- 自動化とガバナンスにより AWS の大規模な採用を可能にする
- Well-Architected の手法によるセキュリティのベストプラクティス
- AWS Control Tower を使って複数のアカウントをビルドおよび統制する
- 既存の組織に対して Control Tower を有効化する

関連ワークショップ:

- Control Tower Immersion Day

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: ルートユーザーをセキュリティ保護することにより、ルートユーザー認証情報を不正使用した場合の偶発的または意図的な損害が生じる可能性が低減されます。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により、アカウントでアクションをコントロールおよび監査できるという安心感が向上する。

このベストプラクティスが確立されてい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。アカウント作成に使用した E メールアドレスとパスワードを使用すれば、ルートユーザーとしてログインできます。AWS ルートユーザーに付与されるアクセス許可が昇格したため、特にそれを必要とす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注意して保護し、AWS アカウント ルートユーザーに対しては多要素認証 (MFA) を必ず有効にしておく必要があります。

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 ベンチマークバージョン 1.4.0 に相互参照されます。AWS アカウント およびリソースのセキュリティ保護については、これらのステップに加え、AWS ベストプラクティス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. アカウントに対して、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すでにある場合、それらのキーを使うプロセスを、AWS Identity and Access Management (IAM) ロールからの臨時アクセスキーを使い、次に ルートユーザーのアクセスキーを削除することにより、移行させる必要があります。
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、AWS Organization 管理アカウントからのパスワードリセットフローを使って、メンバーアカウントへのアクセスを獲得することを検討してください。
    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。ルートユーザーの MFA を有効にする
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対しては、予防的コントロールを有効にします。

a. メンバーアカウントに対して、ルートユーザー向けのルートアクセスキーの作成を許可しない予防的ガードレールの有効化を検討してください。

b. メンバーアカウントに対して、ルートユーザーとしてのアクションを許可しない予防的ガードレールの有効化を検討してください。

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 を設定するには、仮想 MFA または ハードウェア MFA デバイスのいずれかを有効化する手順に従ってください。

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てください。 アカウントごとに最大 8 台の MFA デバイスを登録できます。

- 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無効になることに注意してください。

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。
- 定期的にルートユーザーパスワードを変更することにより、誤って露出したパスワードが不正使用されるリスクを低減します。

検出コントロール

- 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 を有効にすることにより、RootCredentialUsage 所見を使ってルートユーザー API 認証情報の使用をモニタリングおよびアラートを発行します。

- 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コントロール、またはAWS Control Tower を使用している場合は、Control Tower 内にある強く推奨されるコントロールを評価および実装します。

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 タスクを実行するには、ルートユーザーとしてのみログインしてください。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た E メールアドレスと、その他の連絡先に記載された 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、<abuse@amazon.com>から受信したセキュリティ通知が届いていないかどうかモニタリングしてください。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の策定に関する詳細については、AWS セキュリティインシデント対応ガイドと、セキュリティの柱のホワイトペーパーの「インシデント対応」セクションに記載された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

関連するドキュメント:

- AWS Control Tower
- AWS セキュリティ監査のガイドライン
- IAM ベストプラクティス
- Amazon GuardDuty – ルート認証情報使用アラート
- CloudTrail によるルート認証情報使用モニタリングに関するステップバイステップガイダンス
- AWS での使用が認可された MFA トークン
- AWS に break glass アクセスを実装する
- AWS アカウント を改善するためのトップ 10 セキュリティアイテム
- AWS アカウント の不正なアクティビティに気付いた場合はどうすればよいですか?

関連動画:

- 自動化とガバナンスにより AWS の大規模な採用を可能にする
- Security Best Practices the Well-Architected Way (Well-Architected の手法によるセキュリティのベストプラクティス)
- AWS re:inforce 2022 – Security best practices with AWS IAM からの「Limiting use of AWS root credentials (AWS ルート認証情報の使用を制限する)」

関連する例とラボ:

- ラボ: AWS アカウント and root user (AWS アカウントのセットアップとルートユーザー)

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、ビルド、実行から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です。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採用することです。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やり方です。ガバナンスモデルとプロセスは、「あるワークロー

ドの管理目標が達成され、そのワークロードに適切であるということはどのように確認すればよいですか?」という質問への回答に当たります。 意思決定へのアプローチが一定していると、ワークロードのデプロイが加速され、組織内のセキュリティ機能の水準を向上させるのに役立ちます。

ワークロードを安全に運用するには、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。組織レベルおよびワークロードレベルにおいて、「運用上の優秀性」で定義した要件とプロセスを抽出し、それらをすべての領域に適用します。AWS や業界のレコメンデーションおよび脅威インテリジェンスを最新に保つことで、脅威モデルと管理目標を進化させることができます。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。

オートメーションは、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、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、繰り返し行われる作業を一貫して行うことができないというリスクはあります。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、オートメーション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、オートメーションです。デプロイを実行するコードをテストして、それを使ってデプロイを実施することができます。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、まず非運用環境でオートメーションとデプロイされたアプリケーションをテストします。こうすることで、オートメーションをテストして、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い段階でフィードバックが得られるため、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、設定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、オートメーションを使用すると、これが非常に簡単になります。追加の管理目標を定義すると、すべてのワークロードのオートメーション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することができます。複数のチームが利用できるサービスの例としては、AWS アカウントの作成プロセス、人の ID の一元化、ログの共通設定、AMI やコンテナのベースイメージ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サイクル時間を改善して、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。チームの一貫性が高まれば、管理目標を検証し、管理態勢とリスクポジションを利害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統制目標が、コンプライアンス要件に即して明確に定義されています。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が確立されてい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。AWS Artifact を使用して、目標とするフレームワークに沿った文書やレポートを

探してください。これらの文書では、AWS が引き受ける責任の範囲を説明し、残りの貴社の責任となる範囲について助言しています。各種フレームワークの統制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) を使用して、非準拠のリソース構成やアクションをすべての AWS Organizations にわたって予防できます。AWS Config でルールを実装して非準拠のリソースを監視および報告し、動作に確信が持てたらルールを強制モデルに切り替えます。事前定義のマネージドルール一式をサイバーセキュリティフレームワークに合わせて導入するには、最初の選択肢として、AWS Security Hub の標準の利用を検討してください。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。望ましい統制の検知機能が Security Hub に組み込まれていない場合は、AWS Config 適合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して、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、監査担当者、その他の補完サービスによる支援を適宜受けることができます。

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. 統制目標に沿った Security Hub の標準と AWS Config 適合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

関連するドキュメント:

- 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: 業界の公開情報に最新の脅威や推奨事項が反映された時点で把握できます。 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情報を入手することで、脅威の状況における重要な変化を見逃し、ビジネスに影響が及ぶリスクを減らすことができます。 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が確立されてい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。 既知の敵対的戦術、技術、手順 (TTP) に関する文書については、MITRE ATT&CK ナレッジベースを参照してください。MITRE の共通脆弱性識別子 (CVE) リストを確認して、利用している製品の既知の脆弱性に関する情報を入手してください。Open Worldwide Application Security Project (OWASP) の認知度の高い OWASP Top 10 プロジェクトで、ウェブアプリケーションの重大なリスクを把握してください。

AWS セキュリティイベントと推奨される修復手順については、CVE に関する AWS セキュリティ速報で最新情報を入手してください。

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 には、アカウント内の異常な行動や脅威の兆候の検出に関し、業界の脅威インテリジェンスの最新情報が常に反映されます。Amazon Inspector は、CVE のデータベースを自動的に最新の状態に保ち、継続的なスキャン機能に活用しています。 AWS WAF と AWS Shield Advanced の両方で、新しい脅威が出現した時点で自動的に更新されるマネージドルールグループが用意されています。

自動フリート管理やパッチ適用については、「Well-Architected 運用上の優秀性の柱」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、セキュリティスコープを縮小できるかどうかを判断してください。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 サービスを選択する際に、セキュリティ管理のスコープが考慮されています。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負担を全体的に軽減でき、セキュリティリスクと総保有コストを削減できます。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が確立されてい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、Amazon Relational Database Service (Amazon RDS) を使用してデータベースをデプロイする、Amazon Elastic Kubernetes Service (Amazon EKS) や Amazon Elastic Container Service (Amazon ECS) を使用してコンテナをオーケストレーションする、サーバーレスオプションを使用するなどが考えられます。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 セキュリティ統制の証拠として、AWS Artifact で検出した監査アーティファクトを監査人または規制当局に提出できます。また、AWS の一部の監査アーティファクトで提供される責任ガイダンスと「AWS Customer Compliance Guides」を参考にして、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して、環境全体でこれらの更新を追跡および管理できます。

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

関連するドキュメント:

- 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: 標準化されたセキュリティ統制が IaC テンプレートで定義され、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と、時間経過に伴う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が確立されてい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」で説明しているプラクティスを実践すると、AWS Organizations を使用して管理する環境ごとに複数の AWS 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) を使用してアプリケーションコード開発と同じ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のと同じように、IaC を使用して標準のセキュリティ統制を定義し、デプロイすることができます。

可能な限り、セキュリティ統制は宣言的な方法で定義し (AWS CloudFormation で定義する場合と同様)、ソース管理システムに保存してください。 DevOps のプラクティスを実践して、統制のデプロイを自動化してリリースの予測可能性を向上し、AWS CloudFormation Guard などのツールを使ってテストを自動化し、デプロイした統制が目的の構成から逸脱している場合に検出します。 AWS CodePipeline、AWS CodeBuild、AWS CodeDeploy などのサービスを使って、CI/CD パイプライン

を構築できます。「Organizing Your AWS Environment Using Multiple Accounts」のガイダンスを参考にして、これらのサービスを専用のアカウントに設定し、他のデプロイパイプラインと分けておくことを検討してください。

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。 具体的な方法の一例としては、Service Catalog を使用してテンプレートをプロダクトとして公開します。ワークロードチームがこのプロダクトを独自のパイプラインデプロイに組み込むことができます。 AWS Control Tower をご利用の場合は、手始めに使え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 Account Factory 機能もあります。この機能を使えば、定義された標準を使用して、ワークロードチームが新しい AWS アカウント を作成できます。 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

関連するドキュメント:

- Organizing Your AWS Environment Using Multiple Accounts

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

脅威のモデル化を実行し、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が確立されていない場合のリスクレベル: 高

実装のガイダンス

脅威のモデル化とは何ですか?

「脅威のモデル化は、価値のある対象を保護する文脈で、脅威と緩和策を特定、伝達、理解するためのもの」 – The Open Web Application Security Project (OWASP) Application Threat Modeling

脅威をモデル化すべきなのはなぜですか?

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。

脅威のモデル化は、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、「シフトレフト (前倒し)」セキュリティの業界原則と一致しています。最終的に脅威のモデル化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。

脅威のモデル化は、いつ実行すべきですか?

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のモデル化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

脅威のモデル化の実行方法を教えてください

脅威のモデル化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。1 つのアプローチは、Shostack's 4 Question Frame for Threat Modeling (脅威のモデル化のための Shostack の 4 つの質問フレーム) から始めるやり方です。これは、脅威のモデル化の演習に構造を与える自由形式の質問です。

1. うまくいっているものは何か?

   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。構築している対象を視覚化できるため、モデルや図を作成するのが、この質問に対する回答として最も良くある方法です。たとえば、データフロー図などです。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。たとえば、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. どんな問題が起きる可能性があるでしょうか?

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、チーム内の個人全員と脅威のモデル化に関与する関係担当者間の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 などの脅威を特定するモデルを使用すると実施しやすくなります。これは、評価するためのさまざまなカテゴリ (スプーフィング、改ざん、否認、情報漏洩、サービス拒否、権限昇格) を提案するものです。さらに、既存のリストを見直し、OWASP トップ 10、HiTrust 脅威カタログ、そして組織独自の脅威カタログなどのインスピレーションを調査することもブレインストーミングに役立ちます。

3. それをどうするのですか?

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 とお客様との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?」ということも尋ねていると理解することが重要です。お客様と AWS 間の責任のバランスを理解することにより、お客様のコントロール下にある脅威のモデル化演習の範囲を理解するのに役立ちます。これは通常、AWS サービス設定オプションとお客様独自のシステムごとの緩和策を組み合わせたものです。

共有責任の AWS 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範囲内であることに気づくと思います。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からの監査レポートは、AWS 顧客向けに 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、たとえば、AWS 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、専用のセキュリティに関する章が入っており、緩和策とみなされるセキュリティコントロールに関するガイダンスを提供します。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。これらのコントロールは、入力の検証、セッションの取扱い、および範囲の取り扱いなどが考えられます。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. うまくいきましたか?

   狙いは、チームと組織が脅威モデルの質と、脅威のモデル化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。深く掘り下げて実践的な学習を行うため、お客様とチームが「Threat modeling the right way for builders training course (ビルダー向けの正しい脅威モデル化トレーニングコース)」またはワークショップを終了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ライフサイクルに脅威モデル化を統合する方法についてガイダンスを求めている場合、AWS セキュリティブログの「How to approach threat modeling (脅威のモデル化にアプローチする方法)」を参照してください。

Threat Composer

脅威のモデル化の実行に役立てるため、Threat Composer ツールの使用を検討してください。脅威のモデル化における価値実現までの時間の短縮を目的としたツールです。このツールは、以下の用途で役立ちます。

- 脅威の文法に沿って、自然な非線形のワークフローに当てはまる有益な脅威の文章を記述する
- 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 にアクセスして、システム定義の Example Workspace に切り替えてください。

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

関連するドキュメント:

- 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 modelling
- AWS Summit ANZ 2022 - Scaling security – Optimise for fast and secure delivery

関連トレーニング:

- Threat modeling the right way for builders – AWS Skill Builder virtual self-paced training
- Threat modeling the right way for builders - AWS ワークショップ

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が確立されてい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 トピックを購読してください。日々の最新情報をかいつまんで幅広くご案内しています。Amazon GuardDuty や AWS Security Hub 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

ID とアクセス管理

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
- Permissions management

ID 管理

AWS ワークロードを安全に運用するには、2 種類の ID を管理する必要があります。

- ユーザー ID: 管理者、開発者、オペレーター、アプリケーションのエンドユーザーは、AWS 環境とアプリケーションにアクセスできる ID が必要です。これらのユーザーは、あなたの組織のメンバー、または共同作業を行う外部ユーザーで、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、インタラクティブなコマンドラインツールを介して AWS リソースを操作します。
- 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。このような ID には、Amazon EC2 インスタンスや AWS Lambda 関数など、AWS 環境で実行されているマシンが含まれます。また、アクセスを必要とする外部関係者のマシン ID を管理することもできます。さらに、AWS 環境にアクセスする必要があるマシンが AWS 外にある可能性もあります。

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 Identity Center (AWS シングルサインオンの後継サービス)、およびサードパーティー ID プロバイダー向けに強力なサインインメカニズムを使用することにより、AWS の認証情報に対する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が確立されてい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 ベストプラクティスは、AWS に認証する際にフェデレーション (直接フェデレーションまたは AWS IAM Identity Center を使用) を使って、一元化された ID プロバイダーに依存する方法です。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する必要があります。

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。ユーザー (およびルートユーザーを必要とする タスク) へのアクセスを設定するには、アカウントのルートユーザーのみを使用する必要があります。.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 MFA を有効化し、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セキュリティ保護することが重要です。

AWS IAM Identity Center でユーザーを作成する場合、そのサービスでサインインプロセスをセキュリティ保護します。消費者アイデンティティについては、Amazon Cognito user pools を使用して、

そのサービスで、またはAmazon Cognito user pools がサポートする ID プロバイダーの 1 つを使ってサインインプロセスをセキュリティ保護します。

AWS Identity and Access Management (IAM) ユーザーを使用している場合、IAM を使っ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。

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通りです。実際に行う設定は、貴社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 が必要です。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。
- ユーザー自身によるパスワードの変更を許可します。
- 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で、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 Identity Center レコメンデーション

- IAM Identity Center 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 user pools 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。
- ユーザーに対して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 user pools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 Identity Center 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

関連するドキュメント:

- AWS IAM Identity Center (AWS シングルサインオンの後継サービス) 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 の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 (AWS SSO を使用した大規模なユーザー権限の管理)
- Mastering identity at every layer of the cake (すべての層での ID の把握)

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。長期的認証情報を使用すると、多くのリスクが生じます。たとえば、パブリックな GitHub リポジトリにコードでアップロードすることができます。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 users 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が確立されてい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 Management Console 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

従業員、管理者、開発者、オペレーター、および顧客などの人的 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 の直接フェデレーションで、または AWS IAM Identity Center (AWS IAM Identity Center の後継サービス) および好みの ID プロバイダーを使っ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。
- 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。
- 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 user pools を使用して、一時的な認証情報を提供できます。認証情報のアクセス許可は、IAM ロールによって設定されます。 認証されていないゲストユーザーには、制限付きのアクセス権限を持つ IAM ロールを個別に定義できます。

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選択肢として使えず、長期的認証情報を使う必要があるシナリオがあります。これらの状況では、が定期的に認証情報を監査してローテーションし、さらに 長期的認証情報が必要なユースケースに対して定期的にアクセスキーをローテーションします。長期的認証情報が必要となるかもしれない例には、WordPress プラグインやサードパーティーの AWS クライアントなどが考えられます。長期的認証情報を使用すべき状況、またはデータベースログインなどの AWS アクセスキー以外の認証情報については、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、サポートされているサービスを使用して、暗号化されたシークレットを簡単に管理、ローテーション、安全に保存できます。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

関連するドキュメント:

- 一時的なセキュリティ認証情報
- 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
- IAM Identity Center
- ID プロバイダーとフェデレーション
- アクセスキーのローテーション
- Security Partner Solutions: Access and Access Control (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)
- AWS アカウントのルートユーザー

関連動画:

- Managing user permissions at scale with AWS IAM Identity Center (AWS SSO を使用した大規模なユーザー権限の管理) (AWS IAM Identity Center の後継サービス)
- Mastering identity at every layer of the cake (すべての層での ID の把握)

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する:

- 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考える) 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が確立されてい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。たとえば、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。

Credential type	Description	Suggested strategy
IAM access keys	AWS IAM access and secret keys used to assume IAM roles inside of a workload	Replace: Use IAM ロール assigned to the compute instances (such as Amazon EC2 or AWS Lambda) instead. For interoperability with third parties that require access to resources in your AWS ア

Credential type	Description	Suggested strategy
		カウント, ask if they support AWS クロスアカウントアクセス. For mobile apps, consider using temporary credentials through Amazon Cognito ID プール (フェデレーティッドアイデンティティ). For workloads running outside of AWS, consider IAM Roles Anywhere or AWS Systems Manager ハイブリッドアクティベーション.
SSH keys	Secure Shell private keys used to log into Linux EC2 instances, manually or as part of an automated process	Replace: Use AWS Systems Manager or EC2 Instance Connect to provide programmatic and human access to EC2 instances using IAM roles.
Application and database credentials	Passwords – plain text string	Rotate: Store credentials in AWS Secrets Manager and establish automated rotation if possible.
Amazon RDS and Aurora Admin Database credentials	Passwords – plain text string	Replace: Use the Amazon RDS との Secrets Manager 統合 or Amazon Aurora. In addition, some RDS database types can use IAM roles instead of passwords for some use cases (for more detail, see IAM データベース認証).

Credential type	Description	Suggested strategy
OAuth tokens	Secret tokens – plain text string	Rotate: Store tokens in AWS Secrets Manager and configure automated rotation.
API tokens and keys	Secret tokens – plain text string	Rotate: Store in AWS Secrets Manager and establish automated rotation if possible.

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。IAM アクセスキーが AWS サービスと通信する必要がある場合、一時的 (短期的) セキュリティ認証情報を使用します。これらの短期的な認証情報は、EC2 インスタンス用の 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のための Cognito IAM ロール、および IoT デバイス用の IoT Core ポリシーを通して提供できます。サードパーティー向けの場合は、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 は、機密性の高い認証情報を確実かつ安全に保管して取扱うための主な機能を 5 つ提供しています: 保管時の暗号化、転送中の暗号化、総合的な監査、きめ細やかなアクセスコントロール、および拡張可能な認証情報のローテーション。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。

実装手順

1. Amazon CodeGuru などの自動化ツールを使用して、ハードコード化された認証情報を含むコードパスを特定します。
   - 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` をフィルターして、問題のあるコードの行を突き留めます。
2. 削除または置換できる認証情報を特定します。
   a. すで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 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。

      - サーバーレス Lambda 関数では、言語に依存しない Lambda 拡張子を使用できます。
      - 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。

   - 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

関連するドキュメント:

- Getting started with AWS Secrets Manager (AWS シークレットマネージャーの開始方法)
- ID プロバイダーとフェデレーション

- Amazon CodeGuru Introduces Secrets Detector (Amazon CodeGuru がシークレットディテクターを提供)
- How AWS Secrets Manager uses AWS Key Management Service (AWS Secrets Manager が AWS Key Management Service を使用する方法について)
- Secret encryption and decryption in Secrets Manager (Secrets Manager におけるシークレット暗号化と復号化)
- 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 (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)
- Find Hard-Coded Secrets Using Amazon CodeGuru Secrets Detector (Amazon CodeGuru Reviewer Secrets Detector を使ってハードコード化されたシークレットを見つける)
- Securing Secrets for Hybrid Workloads Using AWS Secrets Manager (AWS re:Inforce 2022 - AWS Secrets Manager を使用したハイブリッドワークロードのシークレットの保護)

関連ワークショップ:

- Store, retrieve, and manage sensitive credentials in AWS Secrets Manager (AWS Secrets Manager で機密性の高い認証情報を保存、取得、管理する)
- AWS Systems Manager 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 Management Console 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 Management Console 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、ユーザが AWS Management Console 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、AWS ロールを引き受けることで、 AWS CLI の AWS や AWS STS SDK で使用する 一時的な IAM 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 組織内の複数のアカウントにフェデレーションするには、 AWS IAM Identity Center 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 AWS 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 Studio や AWS IoT Sitewise Monitor ポータル) 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 users およびグループを使用する必要がなくなります。代わりに、ユーザーとグループは AWS 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 users、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て、 該当する IAM users 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 users 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。

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、ウェブアプリやモバイルアプリの認証、承認、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 users 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 Well-Architected の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します
- SEC03-BP06 ライフサイクルに基づいてアクセスを管理する

関連するドキュメント:

- Identity federation in AWS
- IAM でのセキュリティのベストプラクティス
- AWS Identity and Access Management Best practices
- 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:

- Workshop: Using AWS IAM Identity Center to achieve strong identity management
- Workshop: Serverless identity

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、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により軽減できます。

期待される成果: 認証情報のローテーションを実装することにより、長期的認証情報の使用に関連するリスクを軽減します。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う。
- 長期的認証情報を使用して、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確立しない場合のリスクレベル: 中

実装のガイダンス

一時的な認証情報に頼れず、長期的な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、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。

正しい制御が実施されていることを確認するには、定期的な検証、できれば自動化されたツールによる検証が必要です。ユーザー ID の場合、ユーザーにはパスワードの定期的な変更と、一時的な認証情報を優先したアクセスキーの廃止を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。

また、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をお勧めします。AWS Config ルール を設定するか、またはAWS Security Hub セキュリティスタンダードを使って、ユーザーの MFA が有効になっているかどうかをモニタリングできます。IAM Roles Anywhere を使って、マシン ID の一時的な認証情報を提供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の使用が不可能なときは、アクセスキーの監査および更新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、承認された 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。こういった ID には、IAM ユーザー、AWS IAM Identity Center 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、これらに限定されません。たとえば、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 を使って、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します。Amazon GuardDuty は、想定外のアクティビティがあるとアラートを発動します。これは、IAM 認証情報に対する過度に寛容なアクセスまたは意図しないアクセスを示している可能性があります。

- 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、長期的 IAM アクセスキーを定期的にローテーションしてください (最大 90 日ごと)。知らない間にアクセスキーが開始された場合でも、これによりその認証情報を使ってリソースにアクセスされる期間を制限できます。IAM ユーザーのアクセスキー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。
- 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準拠していることを確認します。
- IAM リソース作成および更新の自動化を検討する: IAM Identity Center は、ロールやポリシー管理など多くの IAM タスクを自動化します。または、AWS CloudFormation を使用すると、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減らすことができます。
- IAM Roles Anywhere を使用して、マシン ID の IAM ユーザーを置換する: IAM Roles Anywhere を使用すると、オンプレミスサーバーなど、従来は不可能であった領域でロールを使用できるようになります。IAM Roles Anywhere は、信頼された X.509 証明書を使って AWS を認証し、一時的な認証情報を受け取ります。IAM Roles Anywhere を使用することにより、長期的認証情報がオンプレミス環境に保管されなくなるため、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

関連するドキュメント:

- Getting started with AWS Secrets Manager (Amazon SQS 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
- Security Partner Solutions: Access and Access Control (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)
- 一時的なセキュリティ認証情報

- AWS アカウント アカウントの認証情報レポートの取得

関連動画:

- Best Practices for Managing, Retrieving, and Rotating Secrets at Scale (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 (AWS SSO を使用した大規模なユーザー権限の管理)
- Mastering identity at every layer of the cake (すべての層での ID の把握)

関連する例:

- Well-Architected ラボ - IAM ユーザーの自動クリーンアップ
- Well-Architected ラボ - IAM グループおよびロールの自動デプロイ

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。 部門や場所などの属性は、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: 職務に基づくアクセス許可の変更を、その職務を果たすユーザー全員に適用できます。 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。

このベストプラクティスが確立されていない場合のリスクレベル: 中

実装のガイダンス

AWS のアクセス許可は、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た、ポリシーと呼ばれるドキュメントで定義されます。 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、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をアタッチできます。 `AutomationDeveloper` グループが持つ一部の CloudFront アクセス許可が、`WebAppDeveloper` グループと共通しているとしましょう。この場合、両方の職務を担うユーザーを `CloudFrontAccess` グループに所属させるのではなく、共通のアクセス許可を個別のポリシーで定義し、両方のグループに関連付けることができます。

グループに加えて、属性を使用してアクセス権限をさらに限定できます。例えば、`WebAppDeveloper` グループのユーザーに `Project` 属性を適用し、担当するプロジェクトに固有のリソースにアクセスを限定できます。 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、そのソースによって異なります (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のか、カスタム SAML アサーションとして定義されているのか、または IAM Identity Center 内で設定されているのか)。

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、グループと属性を ID プロバイダー内で定義する必要があるのか、IAM Identity Center 内で定義する必要があるのか、または特定のアカウント内の IAM user グループを使用する必要があるのかを判断できます。
2. グループを定義します。
   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。
   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 との間でグループの作成、メンバーシップ、削除を同期します。

3. 属性を定義します。
   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。 SAML アサーションで `https://aws.amazon.com/SAML/Attributes/PrincipalTag` 属性名を指定して、追加の属性を定義して渡すことができます。
   b. IAM Identity Center 内で定義する場合は、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。
4. 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。

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します
- COST02-BP04 グループとロールを実装する

関連するドキュメント:

- IAM ベストプラクティス
- Manage Identities in IAM Identity Center
- AWS の ABAC とは
- ABAC In IAM Identity Center

関連動画:

- Managing user permissions at scale with AWS IAM Identity Center

- すべての層での ID の把握

Permissions management

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。権限を分けることで、どのような条件で誰が何にアクセスできるかを制御します。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します。例えば、特定のリージョンのみで新しい Lambda 関数を作成することをデベロッパーに許可できます。大規模な AWS 環境を管理する場合、以下のベストプラクティスに従って、それぞれのアイデンティティに必要なアクセスのみを許可し、必要以上に設定しないようにします。

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 での アイデンティティベースのポリシーは、マネージドまたはインラインで、ユーザー、グループ、ロールなどの IAM アイデンティティにアタッチされます。これらのポリシーでは、そのアイデンティティができる内容 (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。

マネージドポリシー – 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、AWS アカウントで複数のユーザー、グループ、およびロールにアタッチできます。マネージドポリシーには 2 つのタイプがあります。

- AWS マネージドポリシー – AWS によって作成および管理されるマネージドポリシー。
- 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マネージドポリシー。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では、ポリシーとアイデンティティ間に厳格な 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は、リソースにアタッチされます。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。リソースベースのポリシーをサポートするサービスの一覧については、「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。

属性ベースのアクセスコントロール (ABAC) では、属性に基づいてアクセス許可を付与することができます。AWS では、これをタグと呼びます。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。IAM ポリシーを使うと、管理者は再利用可能なポリシーを作成して IAM プリンシパルの属性に基づいたアクセス許可を適用できます。例えば、管理者は 1 つの IAM ポリシーを使用して、開発者の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開発者に付与できます。開発者チームがプロジェクトにリソースを追加すると、属性に基づきそれに対するアクセス許可が自動的に適用されます。このため、リソースが追加されるたびにポリシーを更新する必要はありません。

Organizations サービスコントロールポリシー (SCP) は、組織または組織単位 (OU) のアカウントメンバーの最大許容を定義します。SCP は、アイデンティティベースのポリシーやリソースベースのポリシーがアカウント内のエンティティ (ユーザーやロール) に付与する許可を制限するものですが、許可を付与するものではありません。

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。これらのポリシーは、作成されたセッション許可を制限するものですが、許可を付与するものではありません。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またはリソースには、管理者、エンドユーザー、またはその他のコンポーネントからアクセスする必要があります。各コンポーネントにアクセスできるユーザーや内容を明確に定義し、適切な ID タイプと認証および承認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が確立されてい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またはリソースには、管理者、エンドユーザー、またはその他のコンポーネントからアクセスする必要があります。各コンポーネントにアクセスできるユーザーや内容を明確に定義し、適切な ID タイプと認証および承認の方法を選択します。

組織内の AWS アカウントへの通常のアクセスは、 フェデレーションアクセス または一元化された 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。お勧めのアプローチは、最小特権アクセスモデルで構築された IAM 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定義済みの IAM ポリシーを提供します。

AWS のサービス (AWS Secrets Manager や AWS Systems Manager パラメータストア) を使用すると、IAM ロールの使用が不可能なケースで、シークレットをアプリケーションやワークロードから安全に切り離すことができます。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。Systems 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、オートメーションワークフロー内のパラメータを参照できます。

AWS Identity and Access Management Roles Anywhere を使用すると、 IAM 内の一時的なセキュリティ認証情報を取得して、 AWS の外部で実行されるワークロードに使用できます。ワークロードに使用できる IAM ポリシー および IAM ロール は、AWS アプリケーションが AWS リソースにアクセスするために使用するものと同じです。

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。IAM ユーザーに、プログラムによるアクセスと長期の認証情報を付与する必要がある場合は、 最後に使用されたアクセスキーの情報を使用し、 アクセスキーのローテーションと削除を行います。

リソース

関連するドキュメント:

- Attribute-based access control (ABAC)
- AWS IAM Identity Center
- IAM Roles Anywhere
- AWS Managed policies for IAM Identity Center (IAM アイデンティティセンター用の AWS マネージドポリシー)
- AWS IAM policy conditions (AWS IAM ポリシー条件)
- IAM ユースケース
- 必要でない認証情報を削除する
- 「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 (AWS アカウント、OU、または組織に基づいて AWS リソースへのアクセスを制御する方法)
- Identify, arrange, and manage secrets easily using enhanced search in AWS Secrets Manager (AWS Secrets Manager の拡張検索を使用してシークレットを容易に特定、調整、管理する)

関連動画:

- Become an IAM Policy Master in 60 Minutes or Less (60 分以内に IAM ポリシーマスターになる)
- Separation of Duties, Least Privilege, Delegation, and CI/CD (職務分離、最小特権、委任、および CI/CD)

- Streamlining identity and access management for innovation (アイデンティティとアクセスの管理を合理化してイノベーションを実現)

SEC03-BP02 最小特権のアクセスを付与します

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 ID が必要とするアクセス許可のみを付与するのがベストプラクティスです。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、開発者のグループに、扱うプロジェクトのリソースのみを管理することを許可できます。これにより、開発者がプロジェクトから離れると、基盤となるアクセスポリシーに変更を加えることなく、その開発者のアクセスは自動的に取り消されます。

期待される成果: ユーザーは、ジョブの実行に必要なアクセス許可のみを持つ必要があります。ユーザーには、限られた時間内に特定のタスクを実行するためだけに本番環境へのアクセスが与えられ、タスクが完了したらアクセスを取り消す必要があります。アクセス許可は、ユーザーが別のプロジェクトまたは職務に移った場合を含め、不要になったときに取り消す必要があります。管理者権限は、信頼できる管理者の少数のグループのみに付与する必要があります。アクセス許可の変化を避けるため、アクセス許可は定期的にレビューする必要があります。マシンまたはシステムアカウントには、タスクを完了するために必要な最小セットのアクセス許可を付与する必要があります。

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
- ルートユーザーを日常業務に使用する
- 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。
- アクセス許可をレビューして、最小特権アクセスを許可するかどうかを把握しない。

このベストプラクティスが確立されていない場合のリスクレベル: 高

実装のガイダンス

最小特権の原則には、特定のタスクの遂行に必要な最小セットのアクションを実行する許可のみを ID に付与する必要があると記載されています。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。この原則の下で運用すると、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 ユーザーとロールには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、厳格に制御、監視する必要があり、ルートアクセスが必要な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されます。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 ポリシーの作成では、AWS がアクセスを許可または拒否するために必要なサービスアクション、リソース、条件を指定できます。AWS では、アクセスを最小限にするために役立つさまざまな条件を用意しています。例えば、依頼者が AWS 組織に属していない場合、`PrincipalOrgID` 条件キーを使用して、アクションを拒否できます。

AWS のサービスがユーザーに代わって行うリクエスト (AWS CloudFormation による AWS Lambda 関数の作成など) を制御するには、`CalledVia` 条件キーを使用します。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する必要があります。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、アプリケーションチームが独自の IAM ポリシーを作成することは許可できますが、アクセス許可の境界 を適用して、チームが受け取る最大のアクセス許可を制限する必要もあります。

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します。
  - API 使用状況に関するベースポリシー: 必要なアクセス許可を判断する 1 つの方法は、AWS CloudTrail ログをレビューすることです。このレビューでは、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されたアクセス許可を作成できます。IAM Access Analyzer は、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。IAM Access Advisor を組織またはアカウントレベルで使用して、特定のポリシーの最終アクセスの情報を追跡できます。
- 職務に応じた AWS マネージドポリシーの使用を検討してください。 きめ細かいアクセス許可ポリシーの作成を開始するとき、どこから始めればよいかわからない場合があります。AWS には、例えば請求、データベース管理者、データサイエンティストなど、一般的な職種に対するマネージドポリシーがあります。これらのポリシーは、最小特権ポリシーの実装方法を判断している間、ユーザーの持つアクセスを絞り込むことができます。
- 必要でないアクセス許可を削除する: 必要でないアクセス許可を削除し、過度に寛容なポリシーを削ります。IAM Access Analyzer ポリシー生成は、アクセス許可ポリシーの微調整に役立ちます。
- ユーザーの本番環境へのアクセスが制限されるようにする: ユーザーは、有効なユースケースのある本番環境にのみアクセスできるように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管理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では、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。
- アクセス許可のリソースタグを検討する: リソースタグを使用する属性ベースのアクセスコントロールモデルでは、リソースの目的、所有者、環境、またはその他の基準に基づいてアクセスを付与できます。例えば、リソースタグを使用して、開発と本番環境を区別することができます。これらのタグを使用して、開発者を開発環境に制限することができます。タグ付けとアクセス許可ポリシーを組み合わせることで、きめ細かいリソースアクセスを達成でき、すべての職務に複雑な、カスタムポリシーを定義する必要がなくなります。
- AWS Organizations のサービスコントロールポリシーを使用します。 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では、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 にオンボードされたとき、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。アクセス許可レビューは、ユーザーのライフサイクルの各ステップで実行し、アクセス許可が適切に制限されていることを検証して、アクセス許可の変化を回避します。
- アクセス許可をレビューする定期的スケジュールを確立し、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レビューして、過度に寛容なアクセスがないことを検証する必要があります。AWS Config および IAM Access Analyzer は、ユーザーアクセス許可を監査するときに役立ちます。
- 職種マトリックスを確立する: 職種マトリックスは、AWS フットプリント内で必要なさまざまなロールとアクセスレベルを可視化します。職種マトリックスを使用して、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代わりに、グループを使用します。

リソース

関連するドキュメント:

- 最小特権を付与する
- IAM エンティティのアクセス許可の境界
- Techniques for writing least privilege IAM policies (最小特権の IAM ポリシーを作成するテクニック)
- IAM Access Analyzer makes it easier to implement least privilege permissions by generating IAM policies based on access activity (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)
- IAM アクセス許可の境界を使用して開発者にアクセス許可管理を委任する
- 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 でのポリシータイプと使用する状況
- 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
- Zero Trust architectures: An AWS perspective (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 (CloudFormation StackSets を使用して最小特権の原則を実装する方法)
- Attribute-based access control (ABAC)
- ユーザーアクティビティを確認してポリシーの範囲を削減する
- ロールアクセスを表示する
- タグ付けを使用して環境を整理しアカウンタビリティを促進
- AWS のタグ付け戦略
- AWS リソースのタグ付け

関連動画:

- Next-generation permissions management (次世代のアクセス許可管理)
- Zero Trust: An AWS perspective (ゼロトラスト: AWS の視点)
- How can I use permissions boundaries to limit users and roles to prevent privilege escalation? (アクセス許可の境界を使用して IAM ユーザーとロールの範囲を限定し、権限の昇格を防ぐにはどうすればよいですか?)

関連する例:

- ラボ: ロールの作成を委任する IAM アクセス許可の境界
- ラボ: EC2 の IAM タグベースのアクセスコントロール

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、従業員ユーザーはクラウドへのフェデレーションに一元化された ID プロバイダー (SEC02-BP04) を使用して、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:

- 緊急事態と見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。障害モードを特定し、障害の可能性を最小限に抑える高い回復力を備えたシステムを構築するうえで役立つ質問とベストプラクティスは、 信頼性の柱 で確認できます。
- 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 users 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しておくことで、緊急時に利用できなくなる可能性のある 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) に依存する必要がなくなります。さらに、IAM リソースを事前に作成しておくと、 結果整合性のための遅延の可能性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。
- 緊急アクセスに成功した場合と失敗した場合の両方について、詳細な監査ログとイベントが生成されることを確認します。リクエストプロセスと緊急アクセスメカニズムの両方を監視して、誤用や不正アクセスを検出します。インシデント管理システムから進行中の緊急事態とアクティビティを関連付け、想定時間外に障害が発生した場合にアラートを発します。例えば、緊急アクセス AWS アカウント でのアクティビティを監視してアラートする必要があります。こうしたアクティビティは通常の操作では使用されるべきではありません。
- 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。緊急アクセスプロセスは、インシデント対応シミュレーション (SEC10-BP07) およびディザスタリカバリテスト (REL13-BP03) の一環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 ID プロバイダーを利用する」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 プロバイダーを利用することが推奨されます。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 プロバイダーが 4 時間利用できない場合、その間、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。

緊急アクセスプロセスでは、事前に作成されている緊急アクセス用 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、緊急アクセスプロセスに対応するための AWS リソース (IAM ロールや IAM users 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)。

緊急アクセス用アカウントには、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ています。

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、緊急管理者用の強力なパスワードと MFA トークンを設定した一連の IAM users を事前に作成できます。これらの IAM users 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用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 user が 1 人の緊急管理者に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 users と、それぞれに関連付けられた永続的なパスワードと MFA トークンを管理しなければならないことです。

- 緊急アクセス用 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

ワークフォースユーザーが 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 の中断が発生した場合に備えて、AWS Management Console 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が推奨されます。

緊急アクセスプロセスでは、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 Management Console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 ロールや IAM users、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、ワークロード AWS アカウント でクロスアカウントの IAM ロールを事前に作成します。この場合、 AWS CloudFormation StackSets と 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。

- AWS Organizations サービスコントロールポリシー (SCP) を作成して、メンバー AWS アカウント のクロスアカウント IAM ロールの削除と変更を拒否します。
- 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はデフォルトで CloudTrail が有効になっており、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」、「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」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 users を使用する: 強力なパスワードと関連付けられた MFA デバイスを持つ IAM users 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」の追加手順

- 「Set up emergency access to the AWS Management Console」で説明されているとおり、緊急アクセス AWS アカウント で IAM の ID プロバイダーを作成して、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します
- 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

関連するドキュメント:

- Set up emergency access to the AWS Management Console
- SAML 2.0 フェデレーティッドユーザーが AWS Management Console 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、最小特権原則に準拠する必要があります。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確立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。たとえば、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。Attribute-based access control (ABAC) は、各ユーザーに直接権限ポリシーをアタッチするのではなく、ユーザーの属性を利用してアクセス許可を与えることができるため、アクセス権限管理の簡素化に役立ちます。

実装手順

- AWS Identity and Access Management Access Analyzerを使用する: IAM Access Analyzer は、組織内のリソースや、Amazon Simple Storage Service (Amazon S3) バケットや IAM ロールなど、外部エンティティと共有しているアカウントを特定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、きめ細やかなアクセス許可ポリシーを作成するのに役立ちます。
- IAM ユーザーとロールに対して許容可能な期間と使用ポリシーを決定する: 最終アクセスタイムスタンプを使って、使用されていないユーザーとロールを特定し、それを削除します。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、特定のユーザーおよびロールのアクセス許可を特定してスコープを決定できます。たとえば、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 Management Console 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 にデータイベントを口グ記録することを検討する: デフォルトで、CloudTrail は Amazon S3 オブジェクトレベルアクティビティ (たとえば、`GetObject` および `DeleteObject`) または Amazon DynamoDB テーブルアクティビティ (たとえば、`PutItem` および `DeleteItem`) などのデータイベントを口グ記録しません。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

関連するドキュメント:

- 最小特権を付与する
- 必要でない認証情報を削除する
- AWS CloudTrailとは?

- IAM ポリシーを管理する
- DynamoDB のログ記録とモニタリング
- Amazon S3 バケットとオブジェクトの CloudTrail イベントロギングの有効化
- AWS アカウント アカウントの認証情報レポートの取得

関連動画:

- Become an IAM Policy Master in 60 Minutes or Less (60 分以内に IAM ポリシーマスターになる)
- Separation of Duties, Least Privilege, Delegation, and CI/CD (職務分離、最小特権、委任、および CI/CD)
- AWS re:Inforce 2022 - AWS Identity and Access Management (IAM) deep dive (ディープダイブ)

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ガードレールを組み込み、承認の決定を下す際に、プリンシパルの有効なアクセス許可を判断します。 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: 複数の環境が別々の AWS アカウント で明確に分離されています。 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 内にメンバー AWS アカウント を作成しているが、ルート認証情報に対する使用やアクセス許可を制限するために SCP を使用していない。
- 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 の暗黙的な拒否の原則に基づいてアクセス許可を制限しており、明示的に許可された望ましくないアクセス許可をポリシーが付与することはないと安心している。

- 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を設けると、望ましくないアクセス許可をアクセス許可ポリシーが付与しようとした場合でも、実際には付与されないという信頼感が得られます。 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が確立されてい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。 あるアカウントのプリンシパルは、別のアカウントのリソースにアクセスできません。両方のアカウントが同じ AWS 組織や同じ 組織単位 (OU) に属している場合でも、明示的に許可されていない限りはアクセスできません。 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。 サービスコントロールポリシー (SCP) をこの用途で使い、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。

さらに、IAM リソースポリシーを使用して、適用対象のリソースに対して実行できるアクションの範囲と、アクションを実行するプリンシパルが満たす必要のある条件を指定します。 範囲を広くして、プリンシパルが組織の一員である限り (PrincipalOrgId 条件キーを使用して) すべてのアクションを許可することも、範囲を絞って、特定の IAM ロールによる特定のアクションだけを許可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 アクセス許可境界内で付与できるアクセス許可の最大範囲を定義し、このドキュメントを IAM ロールに関連付けます。チームはそのロールを使用して、各自の IAM ロールとアクセス許可を管理できるようになります。 このアプローチにより、業務の完遂に必要な自由をチームに与えつつ、IAM の管理権限を持つことのリスクを軽減できます。

さらに細かく管理するには、特権アクセス管理 (PAM) と一時的な昇格アクセス管理 (TEAM) の手法を実装します。 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。 別の方法としては、通常の運用では、IAM ロールによってプリンシパルに付与されるアクセス許可の範囲をセッションポリシーを使用して制限し、承認された時間枠ではこの制限を一時的に解除します。AWS と一部のパートナーによる検証済みの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 の作成には、許可リスト方式を採用することをお勧めします。許可したアクションを所定の条件下でのみ認め、それ以外のアクションはすべて拒否します。まず、制御したいリソース (Resources) を定義し、Effect を Deny に設定します。NotAction 要素を使用して、指定したアクション以外のすべてのアクションを拒否します。これを NotLike 条件と組み合わせて、指定したアクションを許可する状況を適宜定義します (StringNotLike や ArnNotLike など)。
   b. サービスコントロールポリシーの例を参照してください。
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

関連するドキュメント:

- 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 ソリューション: Temporary Elevated Access Management
- Validated security partner solutions for TEAM

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全ライフサイクルにわたりアクセス許可が監視および調整され、不要な権限が付与されるリスクが軽減されます。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:

- 初期段階で必要とされる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。
- アイデンティティライフサイクルの管理が自動化されていない。

このベストプラクティスが確立されてい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 users 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 Identity Center 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 users 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 Identity Center 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 IAM の適切な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 のロール、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 Identity Center 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 Identity Center の ID ストアに同期します。
5. 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 IAM Identity Center で作成します。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

関連するドキュメント:

- Manage your identity source
- Manage identities in IAM Identity Center
- AWS Identity and Access Management Access Analyzer を使用する
- 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: AWS リソースのうちどれが、誰と共有されるのかを把握します。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が確立されてい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。直接クロスアカウントアクセスを付与するには、Amazon Simple Storage Service (Amazon S3) バケットポリシーなどのリソースベースのポリシーを使用するか、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します。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。IAM Access Analyzer を AWS Organizations で設定して、すべてのアカウントが表示されることを確認します。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を設計する際、信頼ポリシーを使用して、ロールを引き受けるケースを制御できます。たとえば、`PrincipalOrgId` 条件キーを使用して、AWS Organizations 外からのロールを引き受けようとするのを拒否できます。

AWS Config は、設定が誤っているリソースを報告し、AWS Config ポリシーチェックを通して、パブリックアクセスが設定されたリソースを検出できます。AWS Control TowerやAWS Security Hub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、公開されたリソースを特定および修復します。たとえば、AWS Control Tower にはマネージド型のガードレールが含まれており、Amazon EBS スナップショットのうち AWS アカウント によって復元できるものがあるかどうかが検出されます。

実装手順

- AWS Organizations に対して AWS Config を有効化することを検討する: AWS Config では、AWS Organizations 内の複数アカウントからの検出結果を、委任された管理者アカウントに集計することができます。これにより、全体像が把握でき、アカウント全体に 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特定できます。
- AWS Identity and Access Management Access AnalyzerIAM Access Analyzer を設定すると、外部エンティティと共有している、組織やアカウント内のリソース (Amazon S3 バケットや IAM ロールなど) を特定するのに役立ちます。
- AWS Config で自動修復を使用し、Amazon S3 バケットのパブリックアクセス設定の変更に対応します: Amazon S3 バケットに対して、パブリックアクセスのブロック設定を自動的に再有効化することができます。
- モニタリングを実装して、Amazon S3 バケットがパブリックになった場合はアラートを発動する: Amazon S3 パグリックアクセスブロックが無効な場合と、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ために、モニタリングとアラートを設定しておく必要があります。さらに、AWS Organizations を使用している場合、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する必要があります。AWS Trusted Advisor は、オープンアクセス権限がある 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、「AWS Config を使って、パブリックアクセスを許可する Amazon S3 バケットをモニタリングおよび対応する」を参照してください。アクセスをレビューする間、どのようなタイプのデータが Amazon S3 バケットに含まれているかを考慮することが重要です。Amazon Macie は、PII、PHI などの機密性の高いデータ、およびプライベートまたは AWS キーなどの認証情報を検出して保護します。

リソース

関連するドキュメント:

- AWS Identity and Access Management Access Analyzer を使用する
- AWS Control Tower コントロールライブラリ
- AWS Foundational Security Best Practices 標準
- AWS Config マネージドルール
- AWS Trusted Advisor チェックリファレンス
- Amazon EventBridge を使って AWS Trusted Advisor チェック結果をモニタリングする
- 組織内のすべてのアカウントで AWS Config ルールを管理する
- AWS Config および AWS Organizations

関連動画:

- Best Practices for securing your multi-account environment (マルチアカウント環境を守るためのベストプラクティス)
- Dive Deep into IAM Access Analyzer(IAM Access Analyzer を深堀りする)

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のリソースを共有することにより、意図しないアクセスを最小限に抑え、データ損失防止イニシアチブに役立てます。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時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的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確立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を分析するをレビューして、外部からのアクセスを必要なリソースのみに限定し、継続的にモニタリングし、自動的にアラートを出すプロセスを確立するためのガバナンスを確立するのに役立ちます。

AWS Organizations 内のクロスアカウント共有は、多数の AWS サービス (AWS Security Hub、Amazon GuardDuty、および AWS Backup) によって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 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) を使用して、VPC サブネットおよび Transit Gateway 添付ファイル、AWS Network Firewall、または Amazon SageMaker パイプラインなど他の一般的なリソースを共有できます。

リソースを組織内のリソースのみと共有するよう制限するには、サービスコントロールポリシー (SCP) を使って、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際、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すると、意図しないアクセスから保護するのに役立ちます。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` 条件を使用することで、Amazon S3 バケットにアクセスするアイデンティティが組織に確実に属するようにできます。SCP は、サービスにリンクされたロール (LSR) または AWS サービスプリンシパルに適用されないことに注意してください。

Amazon S3 を使用する際、 は Amazon S3 バケットに対して ACL を無効化し、IAM ポリシーをし応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。これは、AWS Key Management Service のサーバー側暗号化を含む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。リソースを外部で共有する権限管理に関する規定的なガイダンスについては、「Permissions management」を参照してください。

実装手順

1. AWS Organizations を使用します。

   AWS Organizations は、組織を作成して一元管理するときに、複数の AWS アカウント を統合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 を AWS サービスと統合します。

   組織のメンバーアカウントを代理してタスクを実行する AWS サービスを有効にすると、AWS Organizations が各メンバーアカウントでそのサービスに対して IAM サービスがリンクされたロールを作成します。AWS Management Console、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可能にするための規範的ガイダンスについては、「AWS Organizations を他の AWS サービスおよび Organizations と併用できる AWS サービスの使用」を参照してください。
3. データ境界を確立します。

   AWS 境界は、AWS Organizations によって管理される組織として表現されるのが普通です。オンプレミスネットワークとシステムとともに、AWS リソースへのアクセスは、My AWS の境界としてみなしているものです。この境界の目標は、アイデンティティが信頼され、リソースが信頼され、そして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にあります。

a. 境界を定義および実装します。

各認証条件について、AWS ホワイトペーパーの「境界の構築」の境界の実装に記載されたステップに従います。ネットワーク層に関する規範的ガイダンスについては、「ネットワークの保護」を参照してください。

b. 継続的にモニタリングとアラートを行います。

AWS Identity and Access Management Access Analyzer は、組織内のリソースや、外部エンティティと共有しているアカウントを特定するのに役立ちます。IAM Access Analyzer を AWS Security Hub と統合して、IAM Access Analyzer から Security Hub へリソースの検出結果を送信および集計し、環境のセキュリティ体制を分析するのに役立てます。統合を有効にするには、各アカウントの各リージョンでIAM Access Analyzer と Security Hub の両方を有効にします。また、AWS Config ルール を使用して、設定を監査し、AWS Chatbot と AWS Security Hub を併用する適切な当事者にアラートを出すことができます。次に、AWS Systems Manager の自動化ドキュメントを使用して、非準拠のリソースを修復できます。

c. 外部で共有するリソースを継続的にモニタリングおよびアラート通知するための規範的ガイダンスについては、「パブリックおよびクロスアカウントアクセスを分析する」を参照してください。

4. AWS サービスのリソース共有を使用し、それに従って制限します。

多くの AWS サービスを使用すると、別のアカウントとリソースを共有したり、別アカウントにある Amazon マシンイメージ (AMI) および AWS Resource Access Manager (AWS RAM) などのリソースをターゲットとすることができます。`ModifyImageAttribute` API を制限して、AMI を共有する信頼できるアカウントを指定します。AWS RAM を使用している場合、`ram:RequestedAllowsExternalPrincipals` 条件を指定し、共有を自組織にのみ限定して、信頼されていないアイデンティティからのアクセスを防止します。規範的ガイダンスおよび注意事項については、「リソース共有および外部ターゲット」を参照してください。

5. AWS RAM を使用して、自分のアカウント内または他の AWS アカウント と安全に共有します。

AWS RAM は、自分のアカウントおよび他の AWS アカウント アカウントのロールやユーザーで作成したリソースを安全に共有するのに役立ちます。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。

リソースベースポリシーを使って過去に共有したリソースがある場合、`PromoteResourceShareCreatedFromPolicy` API または同等のコマンドを使って、リソース共有を完全な AWS RAM リソース共有に昇格させることができます。

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。たとえば、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

関連するドキュメント:

- 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 (IAM ロールと信頼ポリシーを使用する方法)
- Building Data Perimeter on AWS (AWS でのデータ境界の構築)
- 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 Organizations と使用できる AWS サービス
- AWS でデータ境界を確立する: 信頼できるアイデンティティのみが会社データにアクセスできるようにします

関連動画:

- Granular Access with AWS Resource Access Manager (AWS Resource Access Manager を使用したきめ細かいアクセス)
- Securing your data perimeter with VPC endpoints (VPC エンドポイントを使用したデータ境界の保護)
- Establishing a data perimeter on AWS (AWS でのデータ境界の確立)

関連ツール:

- Data Perimeter Policy Examples (データ境界ポリシーの例)

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: ユーザーと関連付けられた長期的AWS Identity and Access Management (IAM) 認証情報、IAM アクセスキー、およびシークレットキーは、認証情報が有効かつアクティブである限り、誰でも使用できます。IAM ロールと一時的な認証情報を使うと、そういった機密性の高い詳細の管理と運用間接費など、長期的認証情報を維持するための業務を減らすことにより、総合的なセキュリティスタンスが改善されます。IAM 信頼ポリシーの外部 ID に対して汎用一意識別子 (UUID) を使用し、IAM ロールにアタッチされた IAM ポリシーを制御かに置くことにより、サードパーティーに付与されたアクセスを監査して、過度に寛容でないことを確認できます。外部共有リソースの分析に関する規範的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確立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。たとえば、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 ID 条件を使っ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 を用いる AWS 外のアプリケーションに対して 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。共有をサポートする他の AWS サービスを継続的に把握しておきます。たとえば、AWS Well-Architected Tool では ワークロードを他の AWS アカウント と共有でき、AWS Resource Access Manager は所有する AWS リソースを安全に他のアカウントと共有するのに役立ちます。

実装手順

1. クロスアカウントロールを使って、外部アカウントへのアクセスを提供します。

   クロスアカウントロールを使うと、顧客にサービスを提供するために外部アカウントとサードパーティーによって保存されている機密情報の量が減ります。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 Partner 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、またはオフサイトロケーションにデータをプル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が AWS ワークロードと安全に対話し、長期的認証情報のニーズをさらに軽減することができます。

   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. サードパーティーに外部 ID を使用します。

   外部 ID を使用することにより、IAM 信頼ポリシーで誰がロールを担うかを指定できます。信頼ポリシーでは、ロールを引き受けるユーザーが、操作を行う条件とターゲットを実施する必要があります。またこの方法により、アカウントの所有者が特定の状況下でのみロールを引き受けることを許可する方法も提供できます。外部 ID の主な機能は、代理人の混乱問題に対処して防止することにあります。

   外部 ID は、AWS アカウント 所有者であり、自分のものに加えて他の AWS アカウント にアクセスするサードパーティーに対してロールを設定した場合、または異なる顧客に代わってロールを引き受けるポジションにある場合に使用します。サードパーティーまたは AWS Partner と連携して、IAM 信頼ポリシーに含める外部 ID 条件を確立します。

3. 汎用一意識別子を使用します。

   汎用一意識別子 (UUID) など、外部 ID に対してランダムな一意の値を生成するプロセスを実装します。サードパーティーが異なる顧客間で外部 ID を再使用しても、「代理人の混乱」問題に対処できません。これは、顧客 A が、顧客 B のロール ARN と重複した外部 ID を使用することで、顧客 B のデータを表示できる可能性があるためです。マルチテナント環境では、サードパーティーが異なる AWS アカウント で複数の顧客をサポートするため、サードパーティーが各 AWS アカウント に対する外部 ID として異なる一意の ID を使用する必要があります。サードパーティーは、重複した外部 ID を検出して、各顧客をそれぞれの外部 ID に安全にマッピングする責任があります。サードパーティーは、外部 ID を指定する際にのみロールを引き受けることができることをテストして検証する必要があります。サードパーティーは、外部 ID が必要となるまで、顧客ロール ARN と外部 ID を保存することを控える必要があります。

   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。

4. 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。キーの管理に関する詳細については、「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に対する反応と軽減の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。

5. セットアップに規範的ガイダンスがあるか、または自動化されていることを確認します。

   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、least-privilege principle 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たクロスアカウントロール作成の詳細については、「クロスアカウントロール」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。

6. 変更するアカウント。

   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。

   検出方法に関する規範的ガイダンスについては、「検出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 の検出

関連するドキュメント:

- 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 (IAM ロールと信頼ポリシーを使用する方法)
- AWS アカウント 間の IAM ロールを使用したアクセスの委任
- 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 (カスタムリソースで外部アカウントに作成された AWS CloudFormation のリソースから情報を収集する)
- Securely Using External ID for Accessing AWS Accounts Owned by Others (他人が所有する AWS アカウントへのアクセスに外部 ID を安全に使用する)
- Extend IAM roles to workloads outside of IAM with IAM Roles Anywhere (IAM Roles Anywhere で AWS IAM ロールを AWS 外のワークロードに拡張する)

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? (別の AWS アカウントのユーザーやロールに、私の AWS アカウントへのアクセスを許可するにはどうすればよいですか?)
- AWSre:Invent 2018: Become an IAM Policy Master in 60 Minutes or Less (60 分以内に IAM ポリシーマスターになる)
- AWS Knowledge Center Live: IAM Best Practices and Design Decisions (ベストプラクティスと設計の決定)

関連する例:

- Well-Architected Lab - Lambda cross account IAM role assumption (Lambda クロスアカウント IAM ロール引き受け) (レベル 300)
- Configure cross-account access to Amazon DynamoDB (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 (AWS STS ネットワーククエリツール)

検知

検知は、予期しないか望ましくない設定の変更の検知と、不要な動作の検知という 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複数個所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 Partner 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、または特定のタイプの API コールの増加の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。たとえば、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から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、内部プロセスまたは義務を遂行する必要がある場合、AWS サービスおよびアプリケーションからのセキュリティイベントログを、確実かつ一貫して迅速に取得できるようにする必要があります。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) メカニズムを導入し、ガバナンス、リスク、コンプライアンス義務のための証拠資料とします。

このベストプラクティスが確立されてい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。クエリと取得メカニズムとアラートを選択、有効化、保存、設定することが非常に重要となります。

実装手順

- ログのソースを選択して有効にします。 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て、有効化します。

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。AWS CloudTrail または AWS Organizations 証跡を使って各 AWS アカウント に証跡を確立し、そのための 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 Management Console、AWS CLI、または AWS 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および確認するには、CloudTrail 証跡を作成して、Amazon S3 バケットと、そしてオプションで Amazon CloudWatch ロググループと関連付けます。または、CloudTrail Lake を作成できます。これは、CloudTrail ログを最長 7 年間保持し、SQL ベースのクエリ施設を提供します。

AWS は、VPC を使用している顧客は、VPC フローログとAmazon Route 53 リゾルバー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 ログを有効化し、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、および Route 53 リゾルバーのクエリログは、AWS でセキュリティ調査をサポートするための基本的なログ記録ソースです。また、Amazon Security Lake を使用して、このログデータを収集、正規化、そして Apache Parquet フォーマットと Open Cybersecurity Schema Framework (OCSF) に保存することもできます。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 検出結果、Amazon Elastic Kubernetes Service (Amazon EKS) 監査ログ、および Amazon EC2 インスタンスのオペレーティングシステムおよびアプリケーションログなど、基本ログソースによって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詳しいリストについては、『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は、ログストレージのオプションを提供し、それぞれが独自の保持とライフサイクル機能が備わっています。2 つの最も良く使用する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。プライマリログ記録オプション

が Amazon CloudWatch ログの場合、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択する: ログストレージの選択肢は、通常、使用するクエリツール、保持機能、精通度、そしてコストに関連しています。ログストレージの主なオプションは、Amazon S3 バケットまたは CloudWatch 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。

- 適切なログ保持を識別する: 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場合、各ログソースに対して適切なライフサイクルを確立して、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 ヶ月 ~ 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有効にする: 組織の各 AWS サービスまたはアプリケーションについて、特定のログ設定ガイダンスを確認してください。

  - AWS CloudTrail 証跡を設定する
  - VPC フローログを設定する
  - Amazon GuardDuty 検出結果エクスポートを設定する
  - 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対して 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 ログ分析情報を 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。

たとえば、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を使ってアラートを提供しています:
  - AWS Config では、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 は脅威検出サービスです。悪意のある動作や不正な動作を継続的にモニタリングし、AWS アカウント とワークロードを保護できるようにします。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、および Amazon EKS 監査ログなどのソースからの情報をインジェスト、集計、および分析します。GuardDuty は、CloudTrail、VPC フローログ、DNS クエリログ、および Amazon EKS から直接独立したデータストリームをプルします。Amazon S3 バケットポリシーを管理したり、ログを収集および保存する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 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指示を構築するたの情報については、『AWS セキュリティインシデント対応ガイド』の「検出」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

関連するドキュメント:

- AWS Security Incident Response Guide (AWS セキュリティインシデント対応ガイド)
- Amazon Security Lake の開始方法
- 開始方法: Amazon CloudWatch Logs
- Security Partner Solutions: Logging and Monitoring (セキュリティパートナーのソリューション: ログ記録とモニタリング)

関連動画:

- AWS re:Invent 2022 - Introducing Amazon Security Lake (Amazon Security Lake の紹介)

関連する例:

- Assisted Log Enabler for AWS (AWS 向けの Assisted Log Enabler)
- AWS Security Hub Findings Historical Export (AWS セキュリティハブ検出結果履歴レポート)

関連ツール:

- Snowflake for Cybersecurity

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、視覚化のアプローチが標準化されています。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: ログデータやイベントを収集して照会するログ記録ソリューションが標準化されていて、ログに含まれる情報からより良いインサイトを引き出せます。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が確立されてい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。

こうした課題を克服するために、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てください。詳細については、「Organizing Your AWS Environment Using Multiple Accounts」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や、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該当します。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。

ログや検出結果を簡単に取り込んで標準化できるように、ログアーカイブアカウントの Amazon Security Lake を評価してください。CloudTrail、Route 53、Amazon EKS、VPC フローログなどの一般的なソースからデータを自動的に取り込むように Security Lake を設定できます。AWS Security Hub を Security Lake へのデータソースとして設定して、Amazon GuardDuty、Amazon Inspector などの他の AWS サービスからの検出結果をログデータと相関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。すべての統合で、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 処理の必要がなくなります。

標準化された場所にセキュリティデータを保存することで、高度な分析機能を使用できます。AWS では、AWS 環境で動作するセキュリティ分析用のツールを、ログアーカイブ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相関付けるオンデマンドクエリを実行することを検討してください。Amazon QuickSight などの視覚化ツールを統合することもできます。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用に標準化した保存場所を設定する
    a. セキュリティデータ用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現できます。AWS では、アカウントでオプトインしている AWS リージョン についても、積極的に使われていなくてもセキュリティデータを収集することを推奨します。
3. 標準化した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 プロセスを開発する必要がある形式ではなく、希望する形式でデータを自動的にエクスポートするオプションを検討してください。 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、Amazon QuickSight、または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、標準化し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 で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 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
- 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

関連するドキュメント:

- AWS Whitepapers: Organizing Your AWS Environment Using Multiple Accounts
- AWS 規範ガイダンス: AWS Security Reference Architecture (AWS SRA)
- AWS 規範ガイダンス: Logging and monitoring guide for application owners

関連する例:

- Aggregating, searching, and visualizing log data from distributed sources with Amazon Athena and Amazon QuickSight
- How to visualize Amazon Security Lake findings with Amazon QuickSight
- Generate AI powered insights for Amazon Security Lake using Amazon SageMaker Studio and Amazon Bedrock
- Identify cybersecurity anomalies in your Amazon Security Lake data using Amazon SageMaker
- Ingest, transform, and deliver events published by Amazon Security Lake to Amazon OpenSearch Service
- How to use AWS Security Hub and Amazon OpenSearch Service for SIEM

関連ツール:

- Amazon Security Lake
- Amazon Security Lake パートナー統合
- Open Cybersecurity Schema Framework (OCSF)
- Amazon Athena
- Amazon QuickSight
- 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複数のセキュリティアラートを生成する場合があります。文脈を完全に理解するには、それらを相互に相関付けて情報を充実 (エンリッチメント) させる必要があります。 セキュリティアラートの相関付けとエンリッチメントを自動化することで、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相関付け、そのデータに情報を補足します。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: アラートの相関付けやエンリッチメントを自動化することで、調査員に求められる認知的な負担と手作業によるデータ準備が全体的に軽減されます。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が確立されてい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、Amazon Macie、Amazon Inspector、AWS Config、AWS Identity and Access Management Access Analyzer、Network Access Analyzer などのサービス
- AWS のサービス、インフラストラクチャ、アプリケーションの口グの自動分析 (Amazon OpenSearch Service のセキュリティ分析など) によるアラート
- 請求アクティビティの変化に応じた、Amazon CloudWatch、Amazon EventBridge、AWS Budgets などのソースからのアラーム
- サードパーティーのソース (脅威インテリジェンスフィードや AWS Partner Network のセキュリティパートナーソリューションなど)
- AWS Trust & Safety またはその他のソース (顧客や従業員など) からの連絡

最も基本的な形式のアラートは、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)、何に対して (影響を受けるリソース) 実行しているかという情報を含みます。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。

相関付けを代行するサービスの例としては、Amazon Detective があります。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外のドメイン名をクエリしていることを警告したとしましょう。このアラート単体では、GuardDuty は「低」重要度を割り当てる可能性があります。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、GuardDuty は、相関付けられたイベントの文脈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。Security Hub、EventBridge、CloudWatch などのサービスをこの用途で利用することを検討してください。
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、CloudTrail、VPC フローログ、Amazon Security Lake、インフラストラクチャログとアプリケーションログなどがあります。
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 に公開できます。

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)

関連するドキュメント:

- 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 findings with account metadata
- How to use AWS Security Hub and Amazon OpenSearch Service for SIEM

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るなど、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が確立されてい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 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は、クラウドセキュリティ体制の管理 (CSPM) ソリューション (AWS Security Hub など) にアラートを送信するように設定し、修正に役立てることをお勧めします。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。

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 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 でカスタムアクションを自動または手動で実行するように設定できます。

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 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 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 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

関連するドキュメント:

- AWS セキュリティインシデント対応ガイド - Detection

関連する例:

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

インフラストラクチャ保護

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。たとえば、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

リージョン、アベイラビリティーゾーン、AWS ローカルゾーン、および AWS Outpostsは AWS セキュアグローバルインフラストラクチャのコンポーネントですが、必ず精通しておいてください。

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 から相当の距離、つまり何キロメートルも離れていますが、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、およびその他の選択された AWS のサービスを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

アエ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。AWS Local 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 により、ネイティブの AWS のサービス、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るようになります。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。AWS Outposts 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
- コンピューティングの保護

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に従えば、ゼロトラスト アプローチを採用することになります。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。ウェブアプリケーションのバックエンドに関する推奨事項については、「AWS Well-Architected サーバーレスアプリケーションレンズ」で 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が確立されてい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。最小特権の原則に則った設計の一部として、可能な限り、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が確立されてい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。これらのサービスに関する詳細なガイダンスについては、「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

関連するドキュメント:

- 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

関連する例:

- Lab: CloudFront for Web Application

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が確立されていない場合のリスクレベル: 中

実装のガイダンス

AWS Network Firewall や、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 Best Practice」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

関連するドキュメント:

- 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:

- Best practices for deploying Gateway Load Balancer
- 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が確立されてい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

関連するドキュメント:

- AWS Security Reference Architecture - 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 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。たしかし、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 ハードニングし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: 脆弱性管理プログラムを作成して維持する。Amazon EC2 インスタンス、Amazon Elastic Container Service (Amazon ECS) コンテナ、および Amazon Elastic Kubernetes Service (Amazon EKS) ワークロードなどのリソースを定期的にスキャンしてパッチを適用する。Amazon Relational Database Service (Amazon RDS) データベースなど、AWS マネージドリソースのメンテナンスウィンドウを設定する。静的コードスキャンを使って、アプリケーションソースコードに一般的な問題がないかどうか検査する。組織に必要なスキルがあるかどうか、または外部のアシスタンスを雇用できるかどうか調べるために、Web アプリケーションペネトレーションテストを検討します。

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するメリット:

このベストプラクティスが確立されていない場合のリスクレベル: 高

実装のガイダンス

脆弱性管理プログラムには、セキュリティ評価、問題の特定、優先順位付け、問題解決の一環としてのパッチ適用の実施などが含まれます。オートメーションは、ワークロードの問題や意図しないネットワークへの露出を継続的にスキャンし、修復を実行するための鍵となります。リソースの作成と更新を自動化することにより時間の節約となり、それ以上の問題を生じさせる設定エラーのリスクを低減します。優れた設計の脆弱性管理プログラムでは、ソフトウェアライフサイクルの開発およびデプロイ段階における脆弱性テストも考慮する必要があります。開発とデプロイ中に脆弱性管理を実装することにより、脆弱性が本番環境に入り込む可能性を低減させます。

脆弱性管理プログラムを実装するには、AWS 責任共有モデルと、それが特定のワークロードにどのように関連するかを理解する必要があります。責任共有モデルでは、AWS に AWS クラウド のインフラストラクチャを保護する責任があります。このインフラストラクチャは、AWS クラウド 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、および施設で構成されています。ユーザーには、実績データ、セキュリティ設定、Amazon EC2 インスタンスの管理タスクなどクラウド内のセキュリティ、さらにはAmazon S3 オブジェクトが適切に分類・設定されていることを確認する責任があります。脆弱性管理へのアプローチは、利用するサービスによっても異なります。たとえば、AWS はマネージド型のリレーショナルデータベースサービス Amazon RDS に対するパッチ適用を管理しますが、自己ホスト型データベースのパッチ適用はユーザーの責任となります。

AWS には、脆弱性管理プログラムに役立つ様々なサービスがあります。Amazon Inspector は、ソフトウェアの問題と意図しないネットワークアクセスを検出するために、継続的に AWS ワークロードをスキャンします。AWS Systems Manager Patch Manager を使うと、Amazon EC2 インスタンス全体のパッチ適用を管理できます。Amazon Inspector と Systems Manager は、AWS Security Hub で表示できます。これは、AWS セキュリティチェックを自動化して、セキュリティアラートを一元化するのに役立つクラウドセキュリティ体制管理サービスです。

Amazon CodeGuru を使うと、静的コード分析を使って、Java および Python アプリケーションの潜在的問題を特定できます。

実装手順

- Amazon Inspector を設定する: Amazon Inspector は新たに起動された Amazon EC2 インスタンス、Lambda 関数、および Amazon ECR にプッシュされた適格なコンテナイメージを自動的に検出し、ソフトウェア問題、潜在的な欠陥、および意図しないネットワーク露出がないかスキャンします。
- ソースコードをスキャンする: ライブラリと依存関係をスキャンして、問題と欠陥がないか調べます。Amazon CodeGuru は、Java と Python アプリケーションの両方について 一般的なセキュリティ問題を修復するための推奨事項を伝えます。The OWASP Foundation は、Source Code Analysis Tools (SAST ツール) を公開しています。
- 既存環境をスキャンしてパッチを適用するメカニズム、さらには CI/CD パイプラインのビルドプロセスの一環としてスキャンするメカニズムを導入する: 新しい脅威からの保護を強化するため、依存関係や OS の問題をスキャンしてパッチを適用するメカニズムを実装します。そのメカニズムを定期的に実行します。ソフトウェア脆弱性管理は、パッチを適用したりソフトウェア問題に対処したりする状況を理解するのに不可欠です。継続的インテグレーション/継続的デリバリー (CI/CD) パイプラインの早期に脆弱性評価を組み込むことで、潜在的なセキュリティ脆弱性の問題修復の優先度を決定します。アプローチは、利用する AWS サービスによっても異なります。Amazon EC2 インスタンスで実行するソフトウェアの潜在的問題をチェックするには、Amazon Inspector をパイプラインに追加して、問題や潜在的欠陥が検出されたらアラートを発動して、ビルドプロセスを停止します。Amazon Inspector は継続的にリソースをモニタリングします。脆弱性管理には、OWASP Dependency-Check、Snyk、OpenVAS、パッケージマネージャー、および AWS Partner ツールを使うこともできます。
- AWS Systems Manager を使用する: Amazon Elastic Compute Cloud (Amazon EC2) インスタンス、Amazon マシンイメージ (AMI)、およびその他多くのコンピューティングリソースなど、AWS リソースのパッチ管理を行う責任があります。AWS Systems Manager Patch Manager は、セキュリティ関連および他のタイプ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 は、Microsoft アプリケーション、Windows sellable サービスパック、およびLinux ベースインスタンスのマイナーアップグレードなど、オペレーティングシステムとアプリケーション両方の Amazon EC2 インスタンスに対するパッチ適用に使用できます。Amazon EC2 に加え、Patch Manager はオンプレミスサーバーへのパッチ適用にも使用できます。

  サポート対象であるオペレーティングシステムの一覧については、Systems Manager ユーザーガイドの「Supported operating systems (サポートされるオペレーティングシステム) 」で確認してください。インスタンスをスキャンして、不足しているパッチのレポートのみを表示したり、不足しているすべてのパッチをスキャンして自動的にインストールしたりできます。
- AWS Security Hub を使用する: Security Hub AWS の総合的なセキュリティ状態を把握できます。複数の AWS サービス全体のセキュリティデータを収集して、標準化されたフォーマットで

それらの検出結果を提供し、AWS サービス全体のセキュリティ検出結果の優先順位を決定できます。

- AWS CloudFormation を使用する: AWS CloudFormation は、複数のアカウントや環境にまたがるリソースデプロイの自動化やリソースアーキテクチャの標準化により、脆弱性管理を支援する Infrastructure as code (IaC) サービスです。

リソース

関連するドキュメント:

- AWS Systems Manager
- Security Overview of AWS Lambda (AWS Lambda のセキュリティ概要)
- Amazon CodeGuru
- Improved, Automated Vulnerability Management for Cloud Workloads with a New Amazon Inspector (改善、自動化された新しい Amazon Inspector のクラウドワークロードの脆弱性管理)
- Automate vulnerability management and remediation in AWS using Amazon Inspector and AWS Systems Manager – Part 1 (Amazon Inspector と AWS Systems Manager を使って AWS の脆弱性管理と修復を自動化する - パート 1)

関連動画:

- Securing Serverless and Container Services (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 (Amazon EC2 インスタンスメタデータサービスにおけるセキュリティベストプラクティス)

SEC06-BP02 ハードニングし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

ハードニングし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、その署名を検証してください。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、ハードニングし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。
- イメージをハードニングしているが、新しい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必要のない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、時間が経つにつれ、コンピューティングリソースがハードニング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: イメージをハードニングすることで、権限のないユーザーやサービスによる不正アクセスを許しかねない、ランタイム環境内の経路の数を減らすことができます。また、不正アクセスを万一受けた場合でも、影響の範囲を小さくすることができます。

このベストプラクティスが確立されていない場合のリスクレベル: 高

実装のガイダンス

システムをハードニングするには、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、その他の認証情報は削除し、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できます。

システムをハードニングする負担を軽減するため、Center for Internet Security (CIS) や国防情報システム局 (DISA) Security Technical Implementation Guide (STIG) など、信頼できる情報源が提供するガイダンスを参考にすることができます。まず、AWS や APN パートナーが公開している Amazon

マシンイメージ (AMI) を使い、AWS EC2 Image Builder を使用して、CIS や STIG の統制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 の推奨事項を適用するハードニング済みのイメージや 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、ハードニング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できます。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 でよりきめ細かなハードニングの推奨事項を実装できるかテストしてください。正常に適用できるさまざまな 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 のハードニングのレシピを 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 ハードニング済みイメージを Amazon Elastic Container Registry (ECR) パブリックリポジトリで入手できます。EC2 Image Builder を使用して、AMI と共にコンテナイメージをハードニングできます。

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 などのツールを使用して、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) を入手できます。コードパッケージを管理するには、プライベートリポジトリ (AWS CodeArtifact 内部など) を信頼できるパブリックリポジトリと統合することをお勧めします。この統合により、パッケージを取得および保存し、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、ソフトウェアコンポジション分析 (SCA)、静的アプリケーションセキュリティテスト (SAST)、動的アプリケーションセキュリティテスト (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てテストできます。

AWS Lambda を使用するサーバーレスワークロードの場合は、Lambda レイヤーを使用してパッケージの依存関係の管理を簡素化します。 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整理します。独自のビルドプロセスでレイヤーを作成および管理できるため、一元的に関数を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ハードニングします。信頼できるソースから取得したベースイメージを、ハードニングした AMI を構築するための基盤として使用してください。EC2 Image Builder を使用して、イメージにインストールされている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ハードニングします。セキュリティのベストプラクティスを満たすよう、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実行して、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。信頼できるコードのみが Lambda 関数で実行されるように、Lambda のコード署名を設定します。

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
- Building end-to-end AWS DevSecOps CI/CD pipeline with open source SCA, SAST and DAST tools

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や自動化ドキュメント (ランブック) に、コンピューティングリソースに対して承認されるアクションが定義されています。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、個々のユーザーログインを管理している。
- 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: アクションの実行を自動化することで、意図しない変更や構成ミスによる運用上の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が確立されてい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 Agent (SSM Agent) 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待ち受け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 Agent 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 では、インスタンスにインタラクティブにアクセスしなくても、パッチの適用、ソフトウェアのインストール、設定の変更を行うことができます。また、Systems Manager でリモートシェルへのアクセスを確立し、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、Systems Manager API の呼び出しを検査のために記録します。

実装手順

1. Amazon EC2 インスタンスに AWS Systems Manager Agent (SSM Agent) 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ている IAM ロールに、`AmazonSSMManagedInstanceCore` マネージド IAM ポリシーが紐付けら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。そのためには、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 を使用して、インスタンスへのインタラクティブアクセスを適宜確立します。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 に監査証跡を残します。

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用いて、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、暗号化技術を用いて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入手されます。ベンダーのウェブサイトの証明書が検証されています。 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用いて検証されます。独自のソフトウェアは暗号化技術を用いて署名され、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わない場合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ことで、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が確立されてい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。 こうしたチェックでは、ペイロードが改ざんされていないことは確認できますが、ペイロードが元のソース (出所) からのものであるかどうかは検証されません。 出所を確認するには、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公開鍵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公開鍵と検証手順を提供しています。その一例を以下に紹介します。

- EC2 Image Builder: Verify the signature of the AWSTOE installation download
- AWS Systems Manager: SSM Agent の署名の確認
- Amazon CloudWatch: CloudWatch エージェントパッケージの署名の検証

イメージの取得とハードニングに使用するプロセスにデジタル署名検証を組み込んでください。詳細については、「SEC06-BP02 ハードニングし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」を参照してください。

AWS Signer を使用すると、署名の検証だけでなく、独自のソフトウェアやアーティファクトのコード署名ライフサイクルも管理できます。 AWS Lambda と Amazon Elastic Container Registry は両方とも Signer に統合でき、コードやイメージの署名を検証できます。 「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 を組み込んで、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

関連するドキュメント:

- 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: コンピューティングリソースのスキャンとパッチ適用をすべて自動化システムで実行します。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、信頼できるソースを出所とし、改ざんされていないことを確認します。 ワークロードは最新の依存関係がないか自動的にチェックされ、AWS コンピューティング環境における信頼性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: 自動化により、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のリスクを軽減できます。 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が確立されてい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、Amazon Inspector を CI/CD パイプラインで使用する方法と、既知の共通脆弱性識別子 (CVE) がないかランタイム環境を継続的にスキャンするために使用する方法の両方を解説しています。 AWS Systems Manager を使用して、自動化したランブックでパッチの適用や新しいイメージからの再デプロイを行い、コンピューティングフリートに最新のソフトウェアやライブラリを適用し、常に最新の状態に保つことができます。 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、信頼できるワークロードのデプロイの一翼も担います。詳細については、「SEC06-BP02 ハードニングし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」および「SEC06-BP04 ソフトウェアの整合性を検証する」を参照してください。 EC2 Image Builder、AWS Signer、AWS CodeArtifact、Amazon Elastic Container Registry (ECR) などのサービスを使用して、ハードニングした承認済みイメージやコードの依存関係をダウンロード、検証、構築、保管できます。 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。 また、ワークロードは署名されているため、AWS Lambda や Amazon Elastic Kubernetes Service (EKS) などの AWS コンピューティング環境では、改ざんされていないことを検証してから実行を許可できます。

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 では NIST 800-53 Rev. 5 標準が提供されていますが、これには「[EC2.8] EC2 instances should use Instance Metadata Service Version 2 (IMDSv2)」などのチェックが含まれています。 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。EC2 インスタンスが IMDSv1 を使用している場合は、Security Hub 内のこのチェックでそのことが検知され、自動修復を開始できます。自動検知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、安全でコンプライアンスに準拠し、ハードニングした 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 を使用した構成管理の自動化
   b. AWS Security Hub を使用したセキュリティおよびコンプライアンス体制の管理の自動化
3. Amazon Elastic Compute Cloud (Amazon EC2) インスタンスのパッチ適用または交換を自動化します。AWS Systems Manager Patch Manager は、セキュリティ関連の更新と他のタイプ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. マルウェアと脅威を自動的に検知してコンピューティングリソースを保護するため、Amazon GuardDuty の使用を検討します。GuardDuty は、AWS Lambda 関数が AWS 環境内で呼び出されたときに、潜在的な問題を検出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します。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

関連するドキュメント:

- 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。たとえば、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。これらは、規制義務への対応ミスを回避したり、規制義務を順守したりする目的の達成に役立つ重要な方法です。

AWS にはデータ保護対策として使用できる多くの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、データの保管場所、データの所有者について把握します。 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。データを把握することが、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、文書化されています。 適切な統制が効いていて、機密データがその分類に基づいて保護されています。 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ています。

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則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: ワークロード内のデータの適切な取り扱いについて、曖昧さが解消されます。 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、法的規制やその他のサイバーセキュリティに関する証明書や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、データのタイプ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が確立されてい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見なし、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、一切アクセスできないというように、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。そのうえで、適切な統制が効いているか、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。

ワークロード内のどこに機密データがあるか分類しやすくするため、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、保護対象医療情報 (PHI) にはタグキーが `Classification` でタグ値が `PHI` のタグと、タグキーが `Sensitivity` でタグ値が `High` の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、これらのリソースが変更されていないか監視し、保護要件に違反する方法で変更された場合 (暗号化設定の変更など) にアラートを送信できます。 タグキーと許容値の標準的な定義は、AWS Organizations の機能であるタグポリシーで指定できます。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推奨されません。

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. 機密データがポリシーに従ってワークロード内に保存され、保護されていることを確認します。 自動テストなどの手法を用いて、統制の有効性を監査します。
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ち得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

関連するドキュメント:

- 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 Tag Policy Syntax and Examples

関連ツール

- AWS Tag Editor

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: 組織内のデータに対し、さまざまな機密レベルが分類ポリシーで定義されています。 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。

このベストプラクティスを活用するメリット: データの分類レベルに併せて統制を行うことで、組織は必要に応じてより高いレベルの統制に投資できます。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が確立されてい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

の機密レベルごとに許容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 アカウントに分割す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、アクセス制御、監査などの関連統制を実装するには、「保管中のデータの保護」および「伝送中のデータの保護」の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

関連するドキュメント:

- 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 Best Practices
- 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: 分類と取り扱いの方針に基づいて適切な統制が効い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: データの識別と分類を自動化することで、データ保護統制の一貫性と正確性が向上し、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が確立されてい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、他の機能を使用して PII を検出し、緩和措置を講じることもできます。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. 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。
2. 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。
3. 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。

4. 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。

リソース

関連するドキュメント:

- AWS Glue: Detect and process sensitive data
- Using managed data identifiers in Amazon SNS
- Amazon CloudWatch Logs: Help protect sensitive log data with masking

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: データは極力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: スケーラブルなデータライフサイクル管理戦略を明確に定義しておくことで、規制コンプライアンスの遵守、データセキュリティの強化、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実現しつつ、適切な統制を維持できます。

このベストプラクティスが確立されてい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採用し、データの保持、アーカイブ、有効期限のプロセスを設定してください。

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、複数の AWS サービスにまたがって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 スナップショットを使用すると、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 の機能を使用して、EBS ボリュームをコピーして保存できます。そうし

たメカニズムのうちの 2 つが、S3 Object Lock と AWS Backup Vault Lock です。バックアップのセキュリティと統制を強化できます。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 つの側面は、ワークロードをデータが進行する過程の履歴を記録することです。これは、データ出所追跡と呼ばれます。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、変換に関するメタデータは Amazon DynamoDB テーブルに記録できます。 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の情報は 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。

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

関連するドキュメント:

- 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。たとえば、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。

暗号化とトークン分割は、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。

トークン分割 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。トークン分割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。たとえば、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分割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。たとえば、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 サービスが、毎回の使用を AWS CloudTrail に記録します。その後、Amazon

CloudWatch 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 保管時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権を強制する能力を提供し、キーの可用性、機密性、完全性の適切なバランスを実現できるものでなければなりません。キーへのアクセスは監視され、キーマテリアルは自動化されたプロセスでローテーションされる必要があります。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:

- 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

多くの規制要件やベストプラクティスには、基本的なセキュリティ制御として保管中のデータの暗号化が含まれています。この制御に準拠するには、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。 AWS の多くのサービスがデータの暗号化に対応するため AWS KMS と統合しています。AWS KMS は、FIPS 140-2 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、 ベストプラクティス として、AWS KMS キーを、そのキーを使用するワークロードと同じアカウントに保持することが考えられます。この分散型モデルでは、AWS KMS キーの管理責任はアプリケーションチームにあります。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 キーポリシーと IAM ポリシーを使用して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。時間を取り、 AWS KMS キーへのアクセス制御に関するベストプラクティ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のがベストプラクティスです。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監視には特に注意する必要があります。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 でキーを削除した場合に、デフォルトで 30 日間の 待機期間が設けられおり、管理者はこれらの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 を使用する必要がある場合は、データを保護するため エンベロープ暗号化 を使用するのがベストプラクティスです。この 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. キーに適切な キー管理オプション を決定します (AWS マネージドまたはカスタマーマネージド)。
   - 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。
   - 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 や 外部キーストアでカスタムキーストアを使用する必要があるかもしれません。
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。
   - AWS のサービスで AWS KMS を使用する方法
   - 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、サービスのユーザーガイドまたは開発者ガイドの「保管時の暗号化」トピック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。
   - 開始方法: AWS Key Management Service (AWS KMS)
   - AWS KMS キーのアクセス制御については、 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。
   - AWS Encryption SDK
5. を 有効にし、 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。
6. Security Hub を 有効にし、 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 キーに適したロ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) はログに記録されるため、AWS KMS に関連する CloudTrail ログが膨大になる可能性があります。

- 組織によっては、AWS KMS のロ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 による AWS KMS API コールのログ記録 セクションを参照してください

リソース

関連するドキュメント:

- AWS Key Management Service
- AWS 暗号化サービスとツール
- 暗号化を使用して Amazon S3 データを保護する
- エンベロープ暗号化
- デジタル主権に関する誓約
- Demystifying AWS KMS key operations, bring your own key, custom key store, and ciphertext portability
- AWS Key Management Service 暗号化の詳細説明

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:

- Implement advanced access control mechanisms using AWS KMS

SEC08-BP02 保管中に暗号化を適用する

保管中のデータには暗号化の使用を適用する必要があります。暗号化は、不正なアクセスや偶発的な開示が発生した場合、機密性の高いデータの機密を保持します。

期待される成果: プライベートデータが保管中にデフォルトで暗号化される。暗号化を行うと、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

できます。暗号化されたデータは、まずそれを解除しないと読み出すこともアクセスすることもできません。暗号化されずに保管されたデータは、インベントリに入れて制御する必要があります。

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が確立されてい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、データに単一の暗号化キーまたは非常にわずかな暗号化キーを使用する場合、過度に寛容な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 を参照してください)。

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 Simple Storage Service (Amazon S3) では、新しいオブジェクトが自動的に暗号化されるように、バケットの暗号化をデフォルトで設定できます。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 Elastic Compute Cloud (Amazon EC2) および Amazon S3 は、デフォルト暗号化を設定することにより、暗号化の適用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、および Amazon S3 バケットなどに対して暗号化を使用していることを自動的に確認します。

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ってデータを暗号化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 ではまた、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

- Amazon S3 に対して保管中に暗号化を適用する: Amazon S3 バケットのデフォルト暗号化を実施します。

  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 Machine Image (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使用して、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。

  各データ分類に対する適切なプリンシパルへのアクセスを制限するポリシーを使って AWS KMS キーを作成および設定する: 例えば、本番環境データの暗号化のために AWS KMS キーを 1 つ、開発またはテストデータの暗号化のためにもう 1 つ作成します。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: 他の AWS サービスを使用する場合は、サービスの暗号化オプションを決定するために、そのサービスの「セキュリティドキュメント」を参照してください。

リソース

関連するドキュメント:

- AWS Crypto Tools
- AWS ドキュメント
- AWS Encryption SDK
- AWS KMS Cryptographic Details Whitepaper (AWS KMS 暗号化の詳細についてのホワイトペーパー)
- AWS Key Management Service
- AWS cryptographic services and tools (AWS 暗号化サービスとツール)
- Amazon EBS Encryption (Amazon EBS 暗号化)
- Default encryption for Amazon EBS volumes (Amazon EBS ボリュームのデフォルトの暗号化)
- Encrypting Amazon RDS Resources (Amazon RDS リソースの暗号化)
- Amazon S3 バケットに対してデフォルトの暗号化を有効にするにはどうすればよいですか。
- 暗号化を使用して Amazon S3 データを保護する

関連動画:

- How Encryption Works in AWS (AWS の暗号化の仕組み)
- Securing Your Block Storage on AWS (AWS でブロックストレージをセキュリティ保護する)

SEC08-BP03 保管時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: 自動システムがデータの保存先 (ストレージ) をスキャンおよび監視し、統制の設定ミス、不正アクセス、予期しない使用を検出します。 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: 自動化のおかげで、データの保存先 (ストレージ) の設定ミスが起きるリスクを防ぐことができます。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が確立されてい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、リソースの設定を Infrastructure as Code (IaC) テンプレートに取り込む方法を説明しています (AWS CloudFormation を使用するなど)。 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。 CloudFormation やその他の IaC ツールではまだ利用できない設定も、AWS Config を使用して設定ミスがないか監視できます。 設定ミスがあった場合に Config が生成するアラートは、自動で修正できます。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。

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」と「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を設定し、AWS Identity and Access Management Access Analyzer で継続的に監視して、アクセス許可を削減できる場合に検出結果を生成する方法を説明しています。 アクセス許可の監視を自動化する以外に、Amazon GuardDuty を設定して、EBS ボリューム (EC2 インスタンス経由)、S3 バケット、サポート対象の Amazon Relational Database Service データベースに対する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 で S3 バケットを監視し、予期しない機密データを検知した場合にアラートを生成する方法について説明しています。アラートをきっかけに自動対応を開始できます。

「REL09 データをバックアップする」のプラクティスに従って、データのバックアップと復旧の自動化戦略を策定してください。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. <ulink type="marketing" url="cloudformation">&CFN;</ulink> を独自の IaC テンプレートに使用し、CloudFormation Guard でテンプレートに設定ミスがないかチェックすることができます。
   b. AWS Config を使用して、事前対応の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. データストレージリソースを監視して統制設定の変更がないかチェックし、設定ミスを検知した場合はアラートを生成して修正アクションを呼び出すように、AWS Config を設定します。
   b. 自動修復に関する詳細なガイダンスについては、「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可能性があ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、既知の脅威シグネチャと、データストレージリソース (EBS ボリューム、S3 バケット、RDS データベースなど) に対するベースラインのアクセス動作からの逸脱を両方とも監視します。
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 バケットを継続的にスキャンし、機密データを検知します。
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 のさまざまなデータソースの暗号化した安全なバックアップを作成す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) で継続的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します
- 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

関連するドキュメント:

- AWS 規範的ガイダンス: 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ってアクセス制御を実施し、最小特権の原則を適用してください。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:「知る必要」に基づき、認証されたユーザーのみがデータへ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が確立されていない場合のリスクレベル: 低

実装のガイダンス

アクセス (最小特権を使用)、分離、バージョニングなど、複数のコントロールによって保管中のデータを保護できます。データへのアクセスは、AWS CloudTrail などの探査メカニズムと、Amazon Simple Storage Service (Amazon S3) アクセスログなどのサービスレベルログを使用して監査する必要があります。パブリックにアクセス可能なデータをインベントリし、時間の経過とともにパブリックで利用可能なデータ量の削減します。

Amazon S3 Glacier のボールトロックと Amazon S3 オブジェクトロックは、Amazon S3 のオブジェクトに対して必須のアクセス制御を提供します。ボールトポリシーがコンプライアンスオプショ

ンを使用してロックされると、ロックの有効期限が切れるまではルートユーザーでも変更できません。

実装手順

- アクセスコントロールを適用する: 暗号キーへのアクセスを含め、最小特権を用いたアクセスコントロールを適用します。
- さまざまな分類レベルに基づいてデータを分離する: データ分類レベルには異なる AWS アカウント を使用し、それらのアカウントの管理には AWS Organizations を使用します。
- AWS Key Management Service (AWS KMS) ポリシーをレビューする: AWS KMS ポリシーで付与されるアクセスのレベルを確認します。
- Amazon S3 バケットとオブジェクトアクセス許可をレビューする: 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確認します。ベストプラクティスは、バケットを公開で読み取ったり書き込んだりできないようにすることです。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。デフォルトでは、すべての S3 バケットはプライベートであり、明示的にアクセスが許可されたユーザーのみがアクセスできます。
- AWS IAM Access Analyzer を有効にする: IAM Access Analyzer は、Amazon S3 バケットを分析して、S3 ポリシーが外部エンティティにアクセスを付与した時点で検出結果を生成します。
- Amazon S3 バージョニングとオブジェクトロックを有効にします (該当する場合)。
- Amazon S3 インベントリを使用する: Amazon S3 インベントリは、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とレポートに使用できます。
- Amazon EBS および AMI 共有アクセス許可をレビューする: 共有アクセス許可は、イメージとボリュームをワークロード外の AWS アカウント に共有することを可能にします。
- AWS Resource Access Manager Shares を定期的にレビューして、リソースを共有し続けるかどうかを決定します。 Resource Access Manager では、AWS Network Firewall ポリシー、Amazon Route 53 リゾルバールール、およびサブネットなど、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

関連するドキュメント:

- AWS KMS Cryptographic Details Whitepaper (AWS KMS 暗号化の詳細についてのホワイトペーパー)
- Introduction to Managing Access Permissions to Your Amazon S3 Resources (Amazon S3 リソースへのアクセス許可の管理の導入)
- Overview of managing access to your AWS KMS resources (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 (理想的な組み合わせ)
- Using versioning (バージョニングの使用)
- Locking Objects Using Amazon S3 Object Lock (Amazon S3 Object Lock を使ってオブジェクトをロックする)
- Sharing an Amazon EBS Snapshot (Amazon EBS スナップショットの共有)
- 共有 AMI
- Hosting a single-page application on Amazon S3 (Amazon S3 でのシングルページアプリケーションのホスティング)

関連動画:

- Securing Your Block Storage on AWS (AWS でブロックストレージをセキュリティ保護する)

伝送中のデータの保護

送信中のデータ 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ロケーション間でデータを安全に保護します。 専用のインフラストラクチャで AWS PrivateLink を使用して、Amazon Virtual Private Cloud (Amazon VPC) またはオンプレミス接続と AWS でホストされるサービスとの間にセキュアなプライベートネットワーク接続を作成します。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 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 および AWS Private Certificate Authority (AWS Private CA) を提供しています。ACM は、パブリックとプライベートの AWS ワークロードの両方で使用するための証明書のプロビジョニング、管理、およびデプロイに使用できる主要なサービスです。ACM は AWS Private CA を使用して証明書を発行し、他の多くの AWS マネージドサービスと統合して、 ワークロード用の安全な TLS 証明書を提供します。

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、 および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 を使用して、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、IoT デバイスへの PKI 証明書のプロビジョニング、管理、およびデプロイに特化したサポートを提供します。AWS IoT Core は、 公開鍵基盤に大規模に IoT デバイスをオンボーディングするための 特殊なメカニズムを提供します。

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できます。ACM は、ウェブサーバー、ロードバランサー、または一般的に信頼されている証明書を使うその他の用途に TLS 証明書をデプロイするために使用できます。
   - 独自のプライベート認証機関階層を確立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 を検討してください。これにより、ACM を、AWS Private CA を使用するさまざまな種類のエンドエンティティ証明書の発行に使用できます。
   - 組み込み型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 マネージド型更新と 統合された AWS のマネージドサービスを使用します。
3. 認証機関を保有するアカウントへの
   - アクセスを追跡するための CloudTrail ログ を有効にします。CloudTrail でログファイルの整合性検証を設定して、ログデータの信頼性を検証することを検討してください。
   - プライベート CA が発行または取り消した 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場合のガイダンスについては、「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

関連するドキュメント:

- AWS でプライベート証明書インフラストラクチャをホストおよび管理する方法
- 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 (ワークショップ)

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伝送中も、データの機密性を保持できます。

期待される成果: すべてのデータは、安全な TLS プロトコルと暗号スイートを使用して伝送中に暗号化する必要があります。データへの不正なアクセスを軽減するためには、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する必要があります。内部 AWS 環境内にのみあるネットワークトラフィックは、可能な場合に TLS を使って暗号化する必要があります。AWS 内部ネットワークはデフォルトで暗号化され、VPC 内のネットワークトラフィックは、トラフィック (Amazon EC2 インスタンス、Amazon ECS コンテナなど) を生成しているリソースに権限のない人がアク

セスしない限り、なりすましや盗聴を行うことはできませんIPsec 仮想プライベートネットワーク (VPN) を使ってネットワーク間のトラフィックを保護することを検討してください。

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が確立されてい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。HTTP など安全でないプロトコルは、セキュリティグループを使用して VPC で監査およびブロックできます。HTTP リクエストは、Amazon CloudFront または Application Load Balancer でHTTPS に自動的にリダイレクトすることもできます。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 からお使いの VPC に VPN で接続して、トラフィックの暗号化を促進できます。クライアントが AWS API に電話かける際に、最低でも TLS 1.2 を使用していることを確認してください。AWS は、2023 年 6 月に TLS 1.0 と 1.1 の使用を廃止予定です。特別な要件がある場合は、AWS Marketplace でサードパーティーのソリューションを入手できます。

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。たとえば、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: HTTPS を Amazon CloudFront と設定して、自分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てください。

- ロードバランサーで安全なプロトコルを設定する: リスナーに接続し、クライアントがサポートするなかで最強の暗号スイートを提供するセキュリティポリシーを選択します。Application Load Balancer に HTTPS リスナーを作成します。
- Amazon Redshift で安全なプロトコルを設定する: クラスターで Secure Socket Layer (SSL) または Transport Layer Security (TLS) 接続が必要となるよう設定します。
- 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 バケットにアップロードする際、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、データに対して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 では、AWS サービスで使用するためのパブリック TLS 証明書をプロビジョニング、管理、およびデプロイできます。
- プライベート PKI ニーズに対してAWS Private Certificate Authority の使用を検討する: AWS Private CA では、プライベート認証局 (CA) 階層を作成し、暗号化された TLS チャネルの作成に使用できる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。

リソース

関連するドキュメント:

- AWS ドキュメント
- Using HTTPS with CloudFront (CloudFront で HTTPS を使う)
- Connect your VPC to remote networks using AWS Virtual Private Network (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 (アプリケーションロードバランサーの HTTPS リスナーを作成する)
- チュートリアル: Amazon Linux 2 で SSL/TLS を設定する
- Using SSL/TLS to encrypt a connection to a DB instance (SSL/TLS を使用した DB インスタンスへの接続の暗号化)
- Configuring security options for connections (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。認証と承認をサポートするネットワークプロトコルを使用すれば、ネットワークフローの制御を強化し、不正アクセスによる影響を軽減できます。

期待される成果: 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ワークロードで明確に定義されている。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承認の統制を組み合わせて、ワークロードの多層防御を実現します。

このベストプラクティスが確立されてい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、ワークロードを構成す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の中にある場合でも、サービス間の通信を暗号化、認証、承認することがベストプラクティスです。

一例として、AWS サービス API は AWS Signature Version 4 (Sigv4) 署名プロトコルを使用して、リクエストの発信元のネットワークに関係なく、呼び出し元を認証します。この認証を通じて AWS API はアクションの要求元の ID を確認することができ、その ID を承認決定の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承認を追加できます。AWS 環境の外のリソースが、SigV4 ベースの認証と承認を必要とするサービスと通信する必要がある場合は、その非 AWS リソースで AWS Identity and Access Management (IAM) Roles Anywhere を使用して、一時的な AWS 認証情報を取得できます。この認証情報を使用して、アクセス権の承認に SigV4 を使用するサービスへのリクエストに署名できます。

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 (B2B) 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 や AWS App Mesh 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。mTLS は TLS 通信の両側に認証情報を提供しますが、承認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一般的に使用されていますが、最近ではサービス間トラフィックで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および文書化する: 多層防御戦略を実装するためには、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: AWS の機能を使用して、データフローに応じたネットワーク統制を確立することを検討してください。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。
  - セキュリティグループを使用して、リソース間のデータフローを確立、定義、制限します。
  - AWS サービスとサードパーティーの AWS PrivateLink 対応サービスの両方との通信に、AWS PrivateLink を使用することを検討してください。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。
- ワークロードのサービス全体に認証と承認を実装する: ワークロードのトラフィックフローを認証および暗号化するために最適な一連の AWS サービスを選択してください。
  - Amazon VPC Lattice でサービス間通信のセキュリティを確保することを検討してください。VPC Lattice では、SigV4 認証を認証ポリシーと組み合わせて使用して、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 App Mesh を検討してください。AWS Private CA を使用して、mTLS で使用する証明書を発行可能なプライベート CA 階層を確立できます。
  - OAuth 2.0 または OIDC を使用するサービスと統合する場合は、API Gateway で JWT オーソライザーを使用することを検討してください。
  - ワークロードと IoT デバイス間の通信については、AWS IoT Core を検討してください。IoT Core は、ネットワークトラフィックの暗号化と認証のオプションをいくつか提供します。
- 不正アクセスを監視する: 意図しない通信チャネル、保護されたリソースにアクセスしようとする非承認のプリンシパル、その他の不適切なアクセスパターンを継続的に監視します。

- サービスへのアクセスの管理に VPC Lattice を使用する場合は、VPC Lattice アクセスログを有効にし、監視することを検討してください。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。
- VPC フローログを有効にして、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確認することを検討してください。
- セキュリティインシデントの計画、シミュレーション、対応に関する詳細なガイダンスについては、「AWS セキュリティインシデント対応ガイド」および「AWS Well-Architected フレームワーク」の「セキュリティの柱」の「インシデント対応」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

関連するドキュメント:

- 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
- AWS IoT Core 認証情報プロバイダーを使用して、AWS サービスへの直接呼び出しを認証
- 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

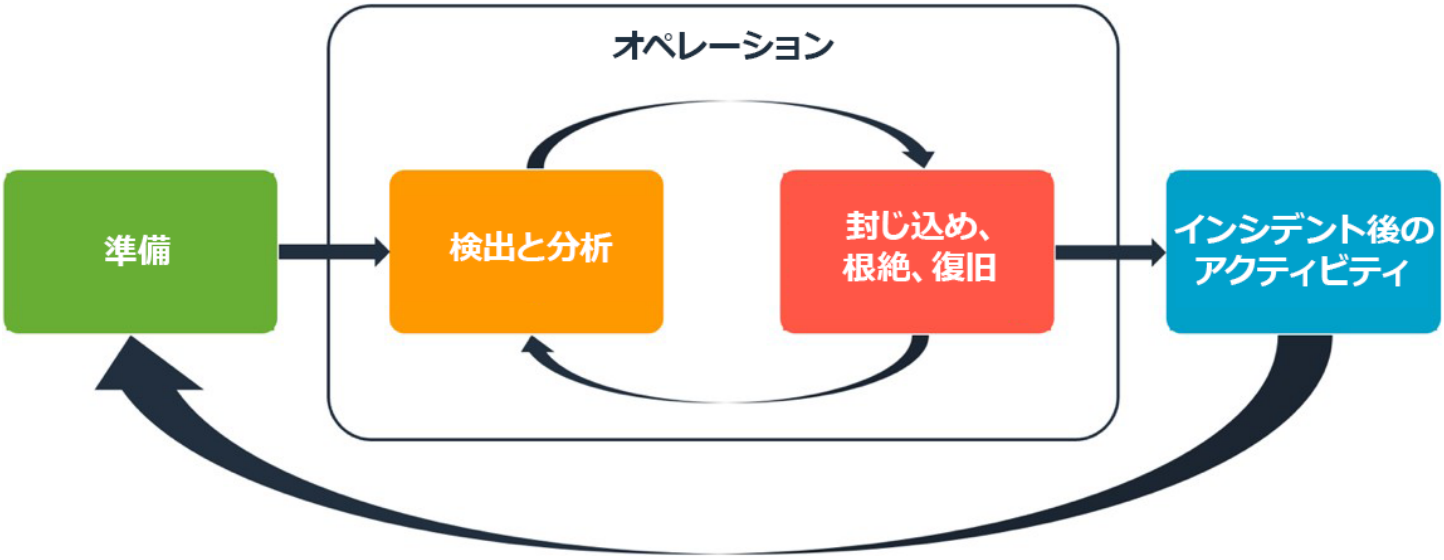
組織内のすべての AWS ユーザーは、セキュリティインシデント対応プロセスの基本を理解している必要があり、セキュリティ担当者はセキュリティ問題への対応方法を理解している必要があります。教育、トレーニング、経験は、クラウドインシデント対応プログラムを成功させるために不可欠であり、起こり得るセキュリティインシデントに対処する前に十分な余裕を持って実施するのが理想的です。クラウドでのインシデント対応プログラムの成功の基盤は、 準備、 オペレーション、 インシデント後のアクティビティです。

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。

- インシデント後のアクティビティ: 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

ただし、NIST SP 800-61 Computer Security Incident Handling Guide などで定義されているインシデント対応の一般的なプロセスと メカニズムは、引き続き正しいものですが、クラウド環境におけるセキュリティインシデントへの対応に関連する、以下の具体的な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 つのドメインにわたって行われます。

- 人材: 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: 主要担当者、その連絡先情報、セキュリティイベント対応時の担当者の役割をまとめたリストが用意されています。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: イベント発生時のトリアージと対応において、適切な人員とその役割を決めるために割く時間が短縮されます。主要担当者とその役割の最新リストが用意してあれば、適切な人材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が確立されていない場合のリスクレベル: 高

実装のガイダンス

組織内の主要な人員を特定する: インシデント対応に動員する必要がある組織内の人員の連絡先リストを用意しておきます。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。インシデントマネージャー、インシデントレスポンダー、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。

- インシデントマネージャー: イベント対応時に全体を統率する権限を持ちます。
- インシデントレスポンダー: 調査および修復の活動を担当します。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: 特に公的機関、規制当局、顧客との組織内外のコミュニケーションを担当します。
- 対象分野のエキスパート (SME): 分散型で自律的なチームでは、ミッションクリティカルなワークロードのために SME を特定しておくことをお勧めします。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。

AWS Systems Manager Incident Manager 機能を主要連絡先の指定、対応計画の定義、オンコールスケジュールの自動化、エスカレーション計画の立案に役立てることを検討してください。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: 企業は独立系ソフトウェアベンダー (ISV)、パートナー、業務委託先が構築したツールを使用して、顧客向けに差別化ソリューションを構築しています。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 AWS Support 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

関連するドキュメント:

- 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、米国の FedRAMP 準拠のワークロードを AWS で運用している場合は、『NIST SP 800-61 Computer Security Handling Guide』を遵守することが役に立ちます。同様に、ヨーロッパの個人を特定できる情報 (PII) データを含むワークロードを運用している場合は、EU 一般データ保護規則 (GDPR) で義務付けられているようにデータレジデンシー関連の問題に対してどのように防御、対応するかといったシナリオを考慮します。

AWS のワークロードについてインシデント管理計画を策定する際は、AWS 責任共有モデル から始め、インシデント対応に向けた多層防御アプローチを構築することを目指します。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 Security Incident Response Guide』(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重要なコンセプトと基本的な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

役割と責任の定義

セキュリティイベントに対処するためには、組織横断的な規律と行動力が必要です。組織内には、人事 (HR)、経営陣、法務部など、インシデント発生時に責任、説明責任、相談、情報提供の役割を持つ担当者が多くいるはずです。これらの役割と責任、および第三者が関与する必要があるかどうかを検討してください。多くの地域には、義務や禁止事項を規定する現地の法律があることに注意してください。セキュリティ対応計画のために責任、説明責任、相談、情報提供 (RACI) チャートを作成するのはマニュアル的に思われるかもしれませんが、作成することで、迅速かつ直接的なコミュニケーションを促進し、イベントのさまざまな段階のリーダーシップを明確に説明できます。

インシデントが発生した場合、影響の測定に役立つ情報や背景を提供できる対象分野のエキスパート (SME) である、影響を受けるアプリケーションやリソースの所有者と開発者を巻き込むことが重要です。インシデント対応について開発者やアプリケーション所有者の専門知識に頼る際は、事前にやり取りを行い、関係を構築してください。アプリケーション所有者や SME (クラウド管理者やエンジニアなど) は、不慣れまたは複雑な環境、対応者がアクセスできない状況下で対応することが必要な場合もあります。

最後に、信頼できるパートナーは、さらなる専門知識や価値のある調査を提供できるため、調査や対応に関与する可能性があります。自分のチームにこれらのスキルがない場合は、外部の人材に支援を依頼するということも検討できます。

AWS 対応チームとサポートを理解する

- AWS Support
  - AWS Support には、AWS ソリューションの成功と運用上の健全性をサポートするツールや専門知識を利用できるさまざまなプランが用意されています。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。
  - お客様 の AWS リソースに影響を与える問題についてのサポートを受けるには、AWS Management Console (サインインが必要) のサポートセンターが中心的な窓口になりま

す。AWS Support へのアクセスは AWS Identity and Access Management によって制御されます。AWS Support 機能へのアクセスの詳細については、 Getting started with AWS Support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、24 時間 365 日の体制で専門のグローバル AWS チームであり、 AWS 責任共有モデルのカスタマーサイドでのアクティブなセキュリティイベント中にカスタマーをサポートします。
  - 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、AWS CIRT と AWS Support ケースで連携できます。
- DDoS 対応のサポート
  - AWS オファー 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マネージドで分散型のサービス拒否 (DDoS) 保護サービスを提供します。Shield は、常時検出と自動インライン緩和により、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るため、DDoS 保護のメリットを得るために AWS Support と連携する必要はありません。Shield のティアには、AWS Shield Standard および AWS Shield Advanced の 2 つのティアがあります。これら 2 つのティアの違いについては、 Shield 機能のドキュメント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。インフラストラクチャを維持するためのベストプラクティスを実行することで、AMS によって運用上のオーバーヘッドとリスクの軽減を支援します。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、サポートするためにライフサイクル全体にわたるサービスを利用できます。
  - 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 のお客様と共有され、AMS との対応策を練り、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を説明します。
- ロールと責任: インシデントに対応する利害関係者を一覧で表示し、インシデントが発生時のそれぞれの役割を詳しく説明します。
- コミュニケーション計画: 連絡先情報とインシデント発生時の連絡方法を詳しく説明します。
- バックアップの通信方法: インシデント時の通信のバックアップとして帯域外通信を行う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段階と取るべき措置: インシデント対応のフェーズ (検出、分析、根絶、封じ込め、復旧など) を列挙し、それらのフェーズで取るべき大まかなアクションも含めます。
- インシデントの重要度と優先順位の定義: インシデントの重大度を分類する方法、インシデントの優先順位付け方法、および重要度の定義が エスカレーション手順にどのように影響するかを詳しく説明します。

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 (セキュリティイベントは、どのように検出して調査するのですか?)

関連するドキュメント:

- AWS Security Incident Response Guide (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) を使用してアカウントをグループ化し、1 つのユニット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。

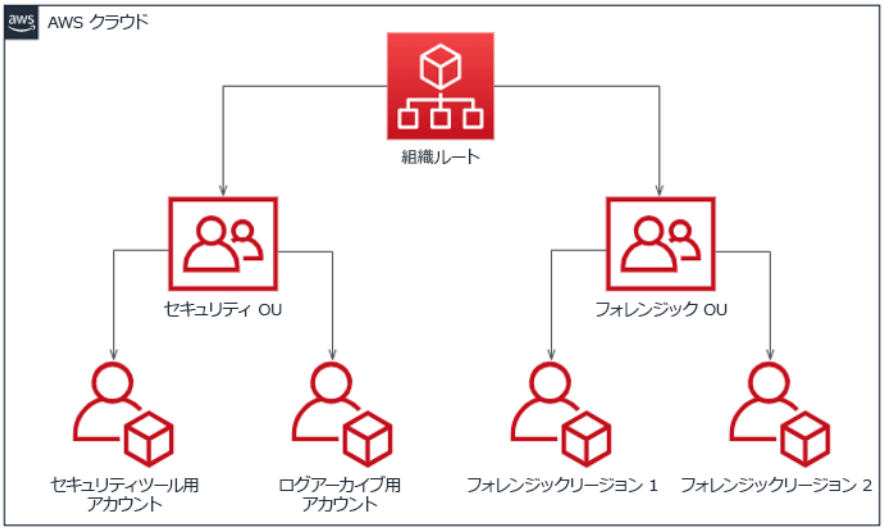
- ログアーカイブ: 限られたアクセス許可を持つログアーカイブ用の AWS アカウント にログを集約します。
- セキュリティツール: セキュリティサービスをセキュリティツール用の AWS アカウント に一元化します。このアカウントは、セキュリティサービスの委任管理者として機能します。

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、US East (N. Virginia) Region (us-east-1) および US West (Oregon) (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 で、もう 1 つは us-west-2 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、AWS 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

関連するドキュメント:

- 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 Prescriptive Guidance - Automate incident response and forensics

関連動画:

- インシデント対応とフォレンジックの自動化

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一連の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: プレイブックは、既知のセキュリティ上の検出結果とアラート (GuardDuty の検出結果など) に基づいて作成する必要があります。GuardDuty の検出結果を受け取っても、どうすればよいかわからないといったことがあるかもしれません。 そこで、GuardDuty の検出結果を誤って処理したり無視したりすることがないように、GuardDuty で検出される可能性のある問題ごとにプレイブックを作成しておきます。修正に関する詳細とガイダンスについては、 GuardDuty ドキュメントを参照してください。なお、GuardDuty はデフォルトでは有効になっておらず、コストがかかりますので注意してください。GuardDuty の詳細については、 「Appendix A: Cloud capability definitions - Visibility and alerting」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。

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

関連するドキュメント:

- 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
- 既存のユーザー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が確立されてい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。AWS は、 適切なアクセス許可を持つ IAM ユーザー を使用することをお勧めします。IAM ユーザーがタスクを実行し AWS のリソースにアクセスするための適切な許可を付与します。ルート認証情報は、 ルートユーザーアクセス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、専用のユーザーアカウントへの事前プロビジョニングをお勧めします。このユーザー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 AWS Identity and Access Management 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 および多要素認証 (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 Management Console 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』(AWS Organizations SCP のための AWS セキュリティベストプラクティス) 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 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、ルートユーザーのパスワードと 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ための IAM ポリシーを設定するには、 IAM Access Analyzer を使用し AWS CloudTrail ログに基づいてポリシーを生成することを検討します。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、プレイブックを一通り実行します。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。

インシデント対応ユーザーアカウントを使用して、 別の AWS アカウント 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。これらのロールに対するすべての `AssumeRole` リクエストが CloudTrail ログに記録され、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 ユーザーアカウントと 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 `AssumeRole` イベント (インシデント対応 IAM ロールに関連する) 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、速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 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

関連するドキュメント:

- Managing temporary elevated access to your AWS environment (AWS 環境へのアクセスの一時的な昇格の管理)
- AWS Security Incident Response Guide (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
- Configuring Cross-Account Access with MFA (MFA を使用したクロスアカウントアクセスの設定)
- Using IAM Access Analyzer to generate IAM policies (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 (マルチアカウント環境の AWS Organizations サービスコントロールポリシーのためのベストプラクティス)
- How to Receive Notifications When Your AWS Account's Root Access Keys Are Used (AWS アカウントのルートアクセスキーを使用した場合の通知の受信方法)
- Create fine-grained session permissions using IAM managed policies(AWS マネージドポリシーを使用して、きめ細かいセッション許可を作成する)

関連動画:

- AWS のインシデント対応とフォレンジックの自動化
- ランブック、インシデントレポート、インシデント対応の DIY ガイド
- AWS 環境のセキュリティインシデントの準備と対応

関連サンプル:

- ラボ: AWS Account Setup and Root User (AWS アカウントのセットアップとルートユーザー)
- ラボ: Incident Response with AWS Console and CLI (AWS コンソールと CLI を使用したインシデント対応)

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、すで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、対応のネイティブ機能備えているほか、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、 Cloud capability definitions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関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

関連するドキュメント:

- Logging strategies for security incident response
- Incident response cloud capability definitions

関連する例:

- Threat Detection and Response with Amazon GuardDuty and Amazon Detective
- 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確立するメリット: 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が確立されてい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ネトレーショ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の目的を定義します。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

関連するドキュメント:

- AWS Incident Response Guide

関連動画:

- AWS GameDay - Security Edition

オペレーション

インシデント対応の実施では、運用が中核となります。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。運用には次の 5 つのフェーズが含まれます。 検知、分析、 封じ込め、根絶、復旧。これらのフェーズと目標の説明は、次の表にあります。

フェーズ	目標
検知	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。
封じ込め	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。
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。封じ込めおよび根絶後は、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、準備のセクションの 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てください。

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運用の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、安全な運用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つことがあります。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

次の要点を広範囲で確立して達成する 教訓 フレームワークを導入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、また関わり方
- 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。
  - 人材
    - 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

関連するドキュメント:

- AWS Security Incident Response Guide - Establish a framework for learning from incidents
- NCSC CAF guidance - Lessons learned

アプリケーションのセキュリティ

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の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、本稼働環境でのアプリケーションのセキュリティ問題の識別、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必要があります。その中で、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、設計フェーズから適切な成果にフォーカスすることができます。AppSec プログラムが成熟するにつれて、自動化を使っ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、本稼働への機能の実装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、パイプラインでのセキュリティ、および依存関係管理の 4 つの領域にフォーカスしています。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、構築、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 定期的にペネトレーションテストを実施する
- SEC11-BP04 手動のコードレビュー

- 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

組織のビルダーに対して、アプリケーションのセキュアな開発と運用のための一般的な手法に関するトレーニングを実施します。セキュリティを重視した開発手法を導入することは、セキュリティのレビューステージでしか検知されない問題が発生する可能性を減らすうえで役立ちます。

期待される成果: セキュリティを考慮したソフトウェアの設計と構築脅威モデルを起点とするセキュアな開発プラクティスについて組織のビルダーをトレーニングすることで、開発されるソフトウェアの全体的な品質とセキュリティを向上させることができます。このアプローチによって、セキュリティレビューステージ後に必要な再作業を削減でき、ソフトウェアや機能をリリースするまでの時間を短縮できます。

このベストプラクティスでは、セキュアな開発は、開発されるソフトウェア、およびソフトウェア開発ライフサイクル (SDLC) をサポートするツールまたはシステムを指します。

一般的なアンチパターン:

- セキュリティレビューを待ち、その後、システムのセキュリティ特性を考慮する。
- セキュリティ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。
- SDLC での意思決定方法に関するコミュニケーションの欠如が、全体的なセキュリティの期待や組織のポリシーに影響を与える。
- セキュリティレビュープロセスが遅延する。

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が確立されていない場合のリスクレベル: 中

実装のガイダンス

組織のビルダーに対してトレーニングを実施します。セキュリティに関するトレーニングについては、脅威モデリングのコースから始めることを推奨します。理想的には、ビルダーは、ワークロードに関する情報に自分たちでアクセスできることが望まれます。このアクセスによって、ビルダーは他のチームに尋ねることなく、開発するシステムのセキュリティ特性に関する十分な情報に基づいた意思決定を行えます。レビューにおけるセキュリティチームの関与プロセスは、明確に定義され、容易に実行できる必要があります。レビュープロセスの各ステップは、セキュリティトレーニングに含める必要があります。既存の実装パターンやテンプレートを利用できる場合、それらは容易に見つけることができ、全体的なセキュリティ要件にリンクされている必要があります。AWS CloudFormation、AWS Cloud Development Kit (AWS CDK) Constructs、Service Catalog、または他のテンプレートツールの使用を検討し、カスタム構成に必要な時間を短縮します。

実装手順

- 良い基盤を築くため、ビルダーのトレーニングを脅威モデリングのコースから始め、セキュリティをどのように考慮すべきかについてのトレーニングを実施します。
- AWS トレーニング と認定、業種、または AWS パートナートレーニングへのアクセスを提供します。
- セキュリティチーム、ワークロードチーム、および他のステークホルダー間の責任分担を明確にするために、組織のセキュリティレビュープロセスについてのトレーニングを実施します。
- 利用可能な場合、コードの例やテンプレートを含め、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を提供します。
- セキュリティレビュープロセスとトレーニングについて、ビルダーチームからフィードバックを定期的に取得し、得られたフィードバックをもとに改善を行います。
- ゲームデーやバグバッシュキャンペーンを活用して、問題数の低減やビルダーのスキル向上に役立てます。

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

関連するドキュメント:

- AWS トレーニング と認定
- How to think about cloud security governance (クラウドのセキュリティガバナンスをどのように考えるか)
- How to approach threat modeling (脅威モデリングにアプローチする方法)
- Accelerating training – The AWS Skills Guild (トレーニングの加速化 – AWS Skills Guild)

関連動画:

- Proactive security: Considerations and approaches (プロアクティブなセキュリティ: 考慮事項とアプローチ)

関連する例:

- Workshop on threat modeling (脅威モデリングについてのワークショップ)
- Industry awareness for developers (開発者向けの業界認識)

関連サービス:

- AWS CloudFormation
- AWS Cloud Development Kit (AWS CDK) (AWS CDK) Constructs
- Service Catalog
- AWS BugBust

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確認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におけるセキュリティ問題のリスクが減ります。

期待される成果: 自動化テストの目標は、開発の初期段階に、また多くの場合開発ライフサイクルをとおして、潜在的な問題を検知するプログラムを使用した手段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、すでに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識別し修正できます。

テストの自動化では、アプリケーションの要件と期待され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に対応する期待される結果の比較に基づき、全体的なテストライフサイクルを促進します。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、自動化に最も適しています。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得ることができます。静的または動的なコード分析の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での潜在的なソフトウェアの問題の検知に役立ちます。

一般的なアンチパターン:

- テストケースおよび自動化テストの結果のコミュニケーションの欠如。
- リリース直前のみでの自動化テストの実施。
- 頻繁に変更される要件に関する自動化テストケース。
- セキュリティテストの結果への対処方法に関するガイダンスの欠如。

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の低減。
- 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る一貫性の向上。
- 本稼働ソフトウェアでのセキュリティ問題の低減。
- ソフトウェアの問題を早期に検知することにより、検知から修正までの時間を短縮。
- 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の向上により、組織全体での改善を促進。

このベストプラクティスが確立されてい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、セキュリティにフォーカスし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、脆弱性のあるコードを検知します。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 の品質グループは、バイトコードスキャナーやバイナリコードスキャナーのオープンソースツールを含むソースコードセキュリティアナライザーの比較結果を提供しています。

潜在的な予期していない動作を識別するため、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) によって静的テストを補完します。動的テストは、静的分析では検知できない潜在的な問題の検知に使用できます。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タイプの潜在的な問題をチェックすることができます。Amazon CodeWhisperer は、ビルダーの IDE 内で、セキュリティスキャンを含むコードのレコメンデーションを提供します。Amazon CodeGuru Reviewer は、アプリケーション開発中の重大な問題、セキュリティの問題、検知が難しいバグを識別し、コード品質の改善のためのレコメンデーションを提供します。

開発者のためのセキュリティワークショップでは、SAST と DAST のテスト手法を含むリリースパイプライン自動化のための AWS CodeBuild、AWS CodeCommit、AWS CodePipeline などの AWS 開発者ツールを使用します。

SDLC を進める中で、セキュリティチームと一緒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部として、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でき、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。
- 単に修正が必要な問題を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。
- 開発者のためのセキュリティワークショップは、リリースパイプラインでの静的および動的テストの統合の例を提供します。
- 開発者の IDE と統合された Amazon CodeWhisperer、コミットのコードスキャン用の Amazon CodeGuru Reviewer などの自動化ツールを使用したテストまたはコード分析の実施は、適切なタイミングでビルダーにフィードバックを提供するのに役立ちます。
- AWS Lambda を使用した構築では、Amazon Inspector を使用して機能内のアプリケーションコードをスキャンできます。

- AWS CI/CD ワークショップは、AWS での CI/CD パイプライン構築の出発点を提供します。
- 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検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の改善を支援します。
- 自動化ツールからの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。

リソース

関連するドキュメント:

- 継続的デリバリーと継続的なデプロイ

- AWS DevOps コンピテンシーパートナー
- アプリケーションセキュリティの AWS セキュリティコンピテンシーパートナー
- Choosing a Well-Architected CI/CD approach (Well-Architected CI/CD アプローチの選択)
- Monitoring CodeCommit events in Amazon EventBridge and Amazon CloudWatch Events (Amazon EventBridge と Amazon CloudWatch Events での CodeCommit イベントの監視)
- Secrets detection in Amazon CodeGuru Review (Amazon CodeGuru Review でのシークレット検知)
- Accelerate deployments on AWS with effective governance (効果的なガバナンスによる AWS でのデプロイの加速)
- AWS での安全なハンズオフデプロイメントの自動化

関連動画:

- Hands-off: Automating continuous delivery pipelines at Amazon (ハンズオフ: Amazon での継続的デリバリーパイプラインの自動化)
- Automating cross-account CI/CD pipelines (クロスアカウント CI/CD パイプラインの自動化)

関連する例:

- Industry awareness for developers (開発者向けの業界認識)

- AWS CodePipeline Governance (AWS CodePipeline ガバナンス) (GitHub)
- Security for Developers workshop (開発者のためのセキュリティワークショップ)
- AWS CI/CD Workshop (AWS CI/CD ワークショップ)

SEC11-BP03 定期的にペネトレーションテストを実施する

定期的にソフトウェアのペネトレーショ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知できない、ソフトウェアの潜在的な問題を識別するうえで役立ちます。また、発見的コントロールの効率について把握するうえでも有効です。ペネトレーションテストでは、保護する必要があるデータを公開する、または予期した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みます。

期待される成果: ペネトレーションテストは、アプリケーションのセキュリティ特性の検知、修正、検証に使用されます。ソフトウェア開発ライフサイクル (SDLC) の一部として、定期的かつ計画的にペネトレーションテストを実施します。ペネトレーショ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。ペネトレーションテストでの検出結果を分析し、自動化によって検知できる問題があるかどうかを識別します。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ネトレーショ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ネトレーションテストを実施する。
- 依存するサードパーティツールやライブラリを除いてアプリケーションのペネトレーショ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ネトレーショ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ついての信頼性の向上。
- 好ましいアプリケーションパターンを識別する機会を創出することによる、ソフトウェアのさらなる品質の向上。
- 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識別するフィードバックループの確立。

このベストプラクティスが確立されていない場合のリスクレベル: 高

実装のガイダンス

ペネトレーションテストは、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知、修正、検証する、構造化されたセキュリティテストです。ペネトレーションテストは、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が作成され実施されます。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の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する際は、必ずペネトレーションテストを実施する必要があります。

ペネトレーショ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に十分に近く、さらに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

- コンテキストを維持するために、脅威モデルに基づいて、ペネトレーションテストのスコープを決定する構造化されたプロセスを確立します。
- ペネトレーショ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。
- ペネトレーショ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、ペネトレーションテストプロセスの時間を短縮します。
- ペネトレーショ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識別し、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

関連するドキュメント:

- AWS ペネトレーションテストでは、AWS でのペネトレーションテストの詳細なガイダンスを提供しています。
- Accelerate deployments on AWS with effective governance (効果的なガバナンスによる AWS でのデプロイの加速)
- 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 Fargate でのペネトレーションテストアーキテクチャのモダナイゼーション)
- AWS Fault injection Simulator

関連する例:

- Automate API testing with AWS CodePipeline (AWS CodePipeline での API テストの自動化) (GitHub)
- Automated security helper (セキュリティヘルパーの自動化) (GitHub)

SEC11-BP04 手動のコードレビュー

作成するソフトウェアについて、手動のコードレビューを実施します。このプロセスは、コードを記述した人物が、コードの品質を確認する唯一のユーザーでないことを検証するうえで役立ちます。

期待される成果: 開発に手動のコードレビューステップを含めることで、開発中のソフトウェアの品質を改善したり、経験の浅いチームメンバーのスキルアップを支援したり、自動化を使用できる領域を識別したりすることができます。手動のコードレビューは、自動化ツールやテストによってサポートすることができます。

一般的なアンチパターン:

- デプロイ前にコードレビューを実施していない。
- 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。
- ペネトレーションテストや後工程において検知される問題数の低減。
- チーム内での知識の移転の改善。

このベストプラクティスが確立されていない場合のリスクレベル: 中

実装のガイダンス

全体的なコード管理フローの一部として、レビューステップを実装します。詳細は、分岐、プルリクエスト、マージで使用するアプローチによって異なります。それらのアプローチでは AWS CodeCommit、または GitHub、GitLab、Bitbucket のようなサードパーティソリューションを使用する場合があります。使用する手法にかかわらず、プロセスを本稼働環境にデプロイする前に、プロセスのレビューが必要であることを認識することが重要です。Amazon CodeGuru Reviewer などのツールを使用すると、コードレビュープロセスを簡単に調整することができます。

実装手順

- コード管理フローの一部として手動のレビューステップを実装し、次に進む前にこのレビューを実施します。
- コードレビューの管理と支援のために Amazon CodeGuru Reviewer の使用を検討します。
- コードを次のステージに進める前にコードレビューの完了を必須とする承認フローを実装します。
- 手動のコードレビューで発見された問題を自動的に検知するプロセスがないか確認します。
- コード開発プラクティスに沿って手動のコードレビューを統合します。

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

関連するドキュメント:

- Working with pull requests in AWS CodeCommit repositories (AWS CodeCommit でのプルリクエストの使用)

- Working with approval rule templates in AWS CodeCommit (AWS CodeCommit での承認ルールテンプレートの使用)
- About pull requests in GitHub (GitHub でのプルリクエストについて)
- Automate code reviews with Amazon CodeGuru Reviewer (Amazon CodeGuru Reviewer を使用したコードレビューの自動化)
- Automating detection of security vulnerabilities and bugs in CI/CD pipelines using Amazon CodeGuru Reviewer CLI (Amazon CodeGuru Reviewer CLI を使用した CI/CD パイプラインでのセキュリティ脆弱性とバグの検知の自動化)

関連動画:

- Continuous improvement of code quality with Amazon CodeGuru (Amazon CodeGuru を使用したコード品質の継続的な改善)

関連する例:

- Security for Developers workshop (開発者のためのセキュリティワークショップ)

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: ソフトウェアは、作成されたコードと、一連の他のソフトウェアパッケージによって構成されます。このため、JSON パーサーや暗号化ライブラリなどの繰り返し使用される機能を容易に実装することができます。これらのパッケージのソースと依存関係を論理的に一元化することで、パッケージが使用される前に、そのパッケージのプロパティを検証するためのメカニズムをセキュリティチームに提供することができます。またこのアプローチによって、既存のパッケージの変更による予期しない問題のリスクや、インターネット上の任意のパッケージをビルダーチームが使用することによるリスクを低減できます。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で、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のより良い理解。
- 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の低減。

このベストプラクティスが確立されてい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。更新が必要な場合や新しい要件が発生した際に新しいパッケージをリポジトリに追加する効率的な方法を実装します。AWS CodeArtifact または類似する AWS パートナーソリューションなどの AWS サービスは、このような機能を提供します。

実装手順:

- ソフトウェアが開発され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。
- 最も大きな変更が発生する頻繁に使用されるパッケージ、言語、チームに関する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識別します。

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

関連するドキュメント:

- Accelerate deployments on AWS with effective governance (効果的なガバナンスによる AWS でのデプロイの加速)
- Tighten your package security with CodeArtifact Package Origin Control toolkit (CodeArtifact Package Origin Control ツールキットを使用してパッケージのセキュリティを高める)
- Detecting security issues in logging with Amazon CodeGuru Reviewer (Amazon CodeGuru Reviewer を使用してセキュリティの問題をログで検知する)
- Supply chain Levels for Software Artifacts (SLSA) (ソフトウェアアーティファクトのサプライチェーンレベル)

関連動画:

- Proactive security: Considerations and approaches (プロアクティブなセキュリティ: 考慮事項とアプローチ)
- The AWS Philosophy of Security (re:Invent 2017) (AWS のセキュリティ哲学、re:Invent 2017)
- When security, safety, and urgency all matter: Handling Log4Shell (セキュリティ、安全性、緊急性のすべてが問題になるとき: Log4Shell への対応)

関連する例:

- Multi Region Package Publishing Pipeline (マルチリージョンパッケージ公開パイプライン) (GitHub)
- Publishing Node.js Modules on AWS CodeArtifact using AWS CodePipeline (AWS CodePipeline を使用した Node.js モジュールの AWS CodeArtifact への公開) (GitHub)
- AWS CDK Java CodeArtifact Pipeline Sample (AWS CDK Java CodeArtifact パイプラインの例) (GitHub)
- Distribute private .NET NuGet packages with AWS CodeArtifact (AWS CodeArtifact を使用したプライベート .NET NuGet パッケージの配布) (GitHub)

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: AWS クラウド でセキュアに構築するためには、原則としてデータへの人的関与を排除します。この原則には、ソフトウェアのデプロイ方法も含まれます。

ソフトウェアのデプロイにおいて人的な依存を排除することで、テスト済みのソフトウェアがデプロイされ、デプロイが常に一貫して実行されるという信頼性を大幅に高めることができます。異なる環境で動作させるために、ソフトウェアを変更する必要はありません。12 要素のアプリケーション開発の原則、具体的には構成の外部化の原則を用いることで、変更を必要とすることなく、複数の環境に同じコードをデプロイすることができます。暗号化を用いたソフトウェアパッケージの署名は、環境間で変更がないことを証明するための便利な手法です。このアプローチによって、変更プロセスでのリスクを低減し、ソフトウェアリリースの一貫性を向上させることができます。

一般的なアンチパターン:

- 本稼働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の向上。
- 変更の失敗がビジネスの機能性に与えるリスクの低減。
- 変更リスクの低減によるリリース頻度の向上。
- デプロイ中に予期しないイベントが発生した場合の自動ロールバック機能。
- 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する能力。

このベストプラクティスが確立されていない場合のリスクレベル: 高

実装のガイダンス

持続的な人的アクセスを環境から排除するために AWS アカウント 構造を構築し、CI/CD ツールを使用してデプロイを実施します。環境固有のデータを AWS Systems Manager Parameter Store な

どの外部ソースから取得するようにアプリケーションを設計します。テスト後にパッケージに署名し、デプロイ中にこれらの署名を検証します。アプリケーションコードをプッシュするように CI/CD パイプラインを設定し、Canary を使用してデプロイの成功を確認します。AWS CloudFormation または AWS CDK などのツールを使用してインフラストラクチャを定義し、AWS CodeBuild と AWS CodePipeline を使用して CI/CD のオペレーションを実行します。

実装手順

- 明確に定義された CI/CD パイプラインを構築して、デプロイプロセスを合理化します。
- AWS CodeBuild と AWS Code Pipeline を使用して、パイプラインへのセキュリティテストの統合を容易にする CI/CD 機能を提供します。
- ホワイトペーパー「Organizing Your AWS Environment Using Multiple Accounts (複数のアカウントを使用した AWS 環境の組織化)」で説明している環境の分離に関するガイダンスに従います。
- 本稼働ワークロードが実行されている環境への持続的な人的アクセスがないことを確認します。
- 構成データの外部化をサポートするようアプリケーションを設計します。
- ブルー/グリーンデプロイモデルを使用したデプロイを検討します。
- Canary を実装してソフトウェアのデプロイの成功を検証します。
- AWS Signer または AWS Key Management Service (AWS KMS) などの暗号化ツールを使用して、デプロイするソフトウェアパッケージの署名と検証を行います。

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

関連するドキュメント:

- AWS CI/CD Workshop (AWS CI/CD ワークショップ)
- Accelerate deployments on AWS with effective governance (効果的なガバナンスによる AWS でのデプロイの加速)
- 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 (AWS Certificate Manager Private CA および AWS Key Management Service 非対称キーを使用したコードの署名)

- Code Signing, a Trust and Integrity Control for AWS Lambda (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 (ハンズオフ: Amazon での継続的デリバリーパイプラインの自動化)

関連する例:

- Blue/Green deployments with AWS Fargate (AWS Fargate を使用したブルー/グリーンデプロイ)

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により、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するパイプラインは、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う必要があります。パイプラインに実装されるテストは、テストを使用するビルダーによって編集できないようにする必要があります。パイプラインへのアクセス許可は、実施するデプロイに必要なものだけに制限し、誤った環境へのデプロイを防ぐための安全対策を実装する必要があります。パイプラインは長期的な認証情報に依存せず、構築環境の整合性を検証できるようにステータスを送信する必要があります。

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ていない。
- 長期的な認証情報、またはハードコード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:

- パイプラインをとおし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の向上。
- 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。

このベストプラクティスが確立されていない場合のリスクレベル: 高

実装のガイダンス

IAM ロールをサポートするマネージド CI/CD サービスから始めることで、認証情報の流出リスクを低減することができます。セキュリティの柱の原則を CI/CD パイプラインインフラストラクチャに適用すると、セキュリティの改善が可能な領域を判断するのに役立ちます。AWS デプロイパイプラインリファレンスアーキテクチャに沿うことは、CI/CD 環境の構築の良い起点となります。パイプラインの実装を定期的にレビューし、予期しない動作のログを分析すると、ソフトウェアのデプロイで使用されているパイプラインの使用パターンの理解に役立ちます。

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を起点とします。
- AWSIAM Access Analyzer を使用して、プログラム的にパイプラインの最小特権の IAM ポリシーを生成することを検討します。
- 予期しないアクティビティや異常なアクティビティを検知するために、監視と警告をパイプラインに実装します。AWS のマネージドサービスである Amazon EventBridge を使用すると、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することができます。

リソース

関連するドキュメント:

- AWS Deployment Pipelines Reference Architecture (AWS デプロイパイプラインリファレンスアーキテクチャ)
- Monitoring AWS CodePipeline (AWS CodePipeline のモニタリング)
- Security best practices for AWS CodePipeline (AWS CodePipeline のセキュリティのベストプラクティス)

関連する例:

- DevOps monitoring dashboard (DevOps モニタリングダッシュボード) (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。また、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: セキュリティのオーナーシップと意思決定をビルダーチームに根付かせるには、セキュリティについての考え方についてビルダーにトレーニングを実施したり、ビルダーチームにセキュリティスタッフを配置または連携させてトレーニングを補強したりします。いずれのアプローチも適切で、チームは開発サイクルの初期にセキュリティに関する質の高い意思決定を行えるようになります。このオーナーシップモデルは、アプリケーションセキュリティのトレーニングを前提にしています。特定のワークロードの脅威モデルから始めると、適切なコンテキストでの設計思考にフォーカスするのに役立ちます。セキュリティにフォーカスしたビルダーコミュニティの構築、またはセキュリティエンジニアグループとビルダーチームを連携させることの別の利点として、ソフトウェアの作成についてより深い理解を得られることが挙げられます。この理解は自動化に関する次の改善領域の特定に役立ちます。

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の短縮。
- セキュリティレビューのステージでようやく検知されるセキュリティ問題の削減。
- 作成されるソフトウェアの全体的な品質の向上。
- システム的な問題、または価値の高い改善領域を識別し、理解する機会の創出。

- セキュリティレビューでの結果に基づく再作業量の削減。
- セキュリティ機能に関する認識の改善。

このベストプラクティスが確立されてい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、組織に最も適切と思われるプログラムの運用モデルを識別します。主な 2 つのパターンは、ビルダー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計測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有益です。ベストプラクティスは、主に組織の変化と文化にフォーカスしています。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。
- ビルダーの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。
- プログラムへの参加人数、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計測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

関連するドキュメント:

- How to approach threat modeling (脅威モデリングにアプローチする方法)
- How to think about cloud security governance (クラウドのセキュリティガバナンスをどのように考えるか)

関連動画:

- Proactive security: Considerations and approaches (プロアクティブなセキュリティ: 考慮事項とアプローチ)

まとめ

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対応の自動化、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

本ドキュメントは、次の人物および組織が寄稿しました。

- Sarita Dharankar、セキュリティピラーリード、Well-Architected、Amazon Web Services
- Adam Cerini、シニアソリューションアーキテクト、Amazon Web Services
- Bill Shinn、シニアプリンシパル、CISO オフィス、Amazon Web Services
- Brigid Johnson、シニアソフトウェア開発マネージャー、AWS Identity、Amazon Web Services
- Byron Pogson、シニアソリューションアーキテクト、Amazon Web Services
- Charlie Hammell、プリンシパルエンタープライズアーキテクト、Amazon Web Services
- Darran Boyd、プリンシパルセキュリティソリューションアーキテクト、金融サービス、Amazon Web Services
- Dave Walker、プリンシパルスペシャリストソリューションアーキテクト、セキュリティとコンプライアンス、Amazon Web Services
- John Formento、シニアソリューションアーキテクト、Amazon Web Services
- Paul Hawkins、プリンシパル、CISO オフィス、Amazon Web Services
- Sam Elmalak、シニアテクノロジーリーダー、Amazon Web Services
- Pat Gaw、プリンシパルセキュリティコンサルタント、Amazon Web Services
- Daniel Begimher、シニアコンサルタント、セキュリティ、Amazon Web Services
- Danny Cortegaca、シニアセキュリティソリューションアーキテクト、Amazon Web Services
- Ana Malhotra、セキュリティソリューションアーキテクト、Amazon Web Services
- Debashis Das、プリンシパル、CISO オフィス、Amazon Web Services
- Reef Dsouza、プリンシパルソリューションアーキテクト、Amazon Web Services
- Brad Burnett、アイデンティティ担当セキュリティソリューションアーキテクト、Amazon Web Services
- Anna McAbee、脅威検出およびインシデント対応担当シニアセキュリティソリューションアーキテクト、Amazon Web Services
- Jason Garman、プリンシパルセキュリティソリューションアーキテクト、Amazon Web Services

その他の資料

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 Framework のホワイトペーパー
- AWS アーキテクチャセンター

改訂履歴

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 フィードをサブスクライブしてください。

変更	説明	日付
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。	June 27, 2024
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を更新し、以下の領域に関するガイダンスを追加。「ワークロードを安全に運用する」、「転送中のデータの保護」。	December 6, 2023
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」の複数のベストプラクティスを更新。「インシデント対応」に 2 つの新しい領域、「オペレーション」、「インシデント後のアクティビティ」を追加。新しい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」を追加。	October 3, 2023
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を更新し、以下の領域に関する新しいガイダンスを追加。「準備」、「シミュレーション」。	July 13, 2023

新しいフレームワークの更新	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。	April 10, 2023
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。	December 15, 2022
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。	October 20, 2022
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。	June 28, 2022
マイナーな更新	AWS PrivateLink 情報を追加し、壊れたリンクを修正。	May 19, 2022
マイナーな更新	AWS PrivateLink を追加。	May 6, 2022
マイナーな更新	インクルーシブでない表現を削除。	April 22, 2022
マイナーな更新	VPC Network Access Analyzer に関する情報を追加。	February 2, 2022
マイナーな更新	イントロダクションに持続可能性の柱を追加。	December 2, 2021
マイナーな更新	壊れたリンクを修正。	May 27, 2021
マイナーな更新	全体を通した編集の変更。	May 17, 2021

メジャーアップデート	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。	May 7, 2021
マイナーな更新	リンクを更新。	March 10, 2021
マイナーな更新	壊れたリンクを修正。	July 15, 2020
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。	July 8, 2020
新しいフレームワークの更新	すべての分野、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。	April 30, 2020
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ための更新とリファレンスの更新。	July 1, 2018
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。	May 1, 2017
初版発行	信頼性の柱 - AWS Well-Architected フレームワークを公開。	November 1, 2016

注意

お客様は、この文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、明示または暗示を問わず、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 契約により規定されます。本書は、AWS とお客様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。