PCI DSS コンプライアンスのための Amazon EKS の設計

2021 年 *12* 月 *4* 日

注意

お客様は、本書に記載されている情報を独自に評価する責任を負うものとします。本書は、(a) 情報提供のみを目的としており、(b) AWS の現行製品とプラクティスを表したものであり、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義務や確約を意味する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、いかなる保証、表明、条件を伴うことなく「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 とお客様の間で行われるいかなる契約の一部でもなく、そのような契約の内容を変更するものでもありません。

目次

はじめに 6

AWS のサービスにおける PCI DSS コンプライアンスステータス 8

AWS 責任共有モデル 8

PCI DSS の適用範囲の決定と検証 10

Amazon EKS デプロイの保護 12

ネットワークセグメンテーション 12

ホストとイメージの堅牢化 17

データ保護 20

アクセスの追跡とモニタリング 25

まとめ 30

寄稿者 31

ドキュメントの改訂 31

要約

AWS 内でマイクロサービスやコンテナを使用して機密データのワークロードをサポートしようとする企業が増えています。本書では、AWS Fargate 用の Amazon Elastic Kubernetes Service (Amazon EKS) または Payment Card Industry Data Security Standard (PCI DSS) バージョン 3.2.1 準拠の Amazon Elastic Compute Cloud (EC2) 起動タイプを設定する際にお客様が考慮すべきベストプラクティスの概要を説明します。本書は、PCI DSS のすべての統制を網羅しているわけではありません。一部の統制はコンテナに該当せず、お客様間で環境は異なる場合があるためです。

本書は、PCI DSS コンプライアンスに向けて AWS クラウド環境のアーキテクチャの設計に関心があるシステムアーキテクト、デベロッパー、セキュリティ担当者、リスクおよびコンプライアンス担当者を対象としています。

本書は、アマゾン ウェブ サービスの全額出資子会社である AWS Security Assurance Services, LLC (AWS SAS) が作成したものです。AWS SAS は、独立した PCI 認定セキュリティ評価 (QSA) 機関 (QSAC) であり、AWS のお客様とパートナーに対して、AWS クラウド上で PCI DSS コンプライアンスを達成するための手引きとなる具体的な情報を提供します。

はじめに

ペイメントカード業界のデータセキュリティ基準 (PCI DSS) は、人、プロセス、テクノロジーに適用される、ペイメントカード処理環境の保護に関する技術および運用上のガイダンスを提供します。カード会員データ (CHD) を保存、処理、または伝送する事業体は、PCI DSS に照らしてカード会員データ環境 (CDE) 統制のコンプライアンスを検証する必要があります。このような事業体の例として、加盟店、ペイメントプロセサー、サービスプロバイダーがあります。

AWS では、PCI DSS コンプライアンスに準拠していることが証明された多くのサービスを提供しています。詳細については、AWS Artifact を参照してください。AWS Artifact は、AWS のセキュリティおよびコンプライアンスレポートと一部のオンライン契約にオンデマンドでアクセスできる、コンプライアンス関連情報の中心的なリソースです。企業はこれらのサービスを使用して、コンプライアンスの労力を減らすことができます。継続的に成長している 1 つの分野は、AWS のコンテナ化されたソリューションの使用です。

PCI DSS に準拠しているサービスとは、それを使用するとお客様の環境がデフォルトで準拠するという意味ではありません。PCI DSS 要件を満たすようにそのサービスを設定できるという意味です。パラメータがアクセス可能で、お客様の側で設定できる場合、該当するコンプライアンス要件を満たすようにこれらのパラメータを確実に設定することはお客様の責任となります。AWS PCI DSS 評価に含まれていないその他の AWS のサービスでも、PCI DSS の統制を満たすために使用できる場合があります。

AWS のコンテナソリューションには、マネージドサービスである Amazon Elastic Container Service Amazon (Amazon ECS) と Amazon Elastic Kubernetes Service (Amazon EKS) が含まれます。どちらのサービスもコンテナを AWS Fargate

または Amazon Elastic Compute Cloud (Amazon EC2) へデプロイできます。AWS Fargate はサーバーレスコンピューティングエンジンであり、Amazon EC2 インスタンスのプロビジョニングと管理は不要です。Amazon EC2 へコンテナをデプロイの場合は、お客様はコンテナをホストする基盤となる Amazon EC2 インスタンスを管理します。

ワークロードをコンテナサービスに移行することのメリットには、プラットフォームの独立性、デプロイの速度、リソースのより効率的な利用などがあります。クラウドワークロードと同様に、コンテナの導入時にセキュリティを設計する方法を理解することが重要です。コンテナ環境の一時的で動的な性質により、継続的に変化する CDE が作成され、評価が困難になる可能性があります。

コンテナ化されたアプリケーションの攻撃ベクトルは、コンテナベースでないアプリケーションのデプロイで発生するものと似ていますが、これに「コントロールプレーン」と呼ばれるコンテナオーケストレーションレイヤーが追加されています。他のアプリケーションのデプロイと同様に、Open Web Application Security Project (OWASP) の懸念事項への対応を含めて、ベストプラクティス内で運用を継続することをお勧めします。

コンテナワークロードは、通常、主要なタスクを実行するように設計されており、それによって分散環境が作成されます。コンテナによって実装されるサービスは、ネットワークとの相互依存性が高くなり、スケジューリング、スケーリング、リソース管理が必要になります。コンテナは、仮想マシンとは異なり、オペレーティングシステムのカーネルを共有します。

この設定により、特定のホストのすべてのコンテナにアクセスするために使用できる共通の攻撃ポイントが生まれる可能性があります。1 つのオペレーティングシステムで複数のコンテナを実行する場合、すべてのコンテナが共通のネットワークインターフェイ

スを共有できます。本書では、このセキュリティリスクを軽減するために、AWS のサービスに関連して構築できるさまざまなソリューションについて説明します。

AWS のサービスにおける PCI DSS コンプライアンスステータス

AWS は、お客様がコンプライアンス要件をより簡単に満たせるようにする、レベル 1 の PCI DSS サービスプロバイダーです。PCI DSS 評価の各サービスの適用範囲では、お客様から提供されたすべてのデータにプライマリアカウント番号 (PAN) や重要な認証データ (SAD) が含まれている可能性があること、およびこのようなデータのセキュリティが影響を受けることを前提としています。この評価には、PCI DSS の適用範囲内のサービスをサポートする AWS データセンターに適用されるすべての物理的セキュリティ要件も含まれます。

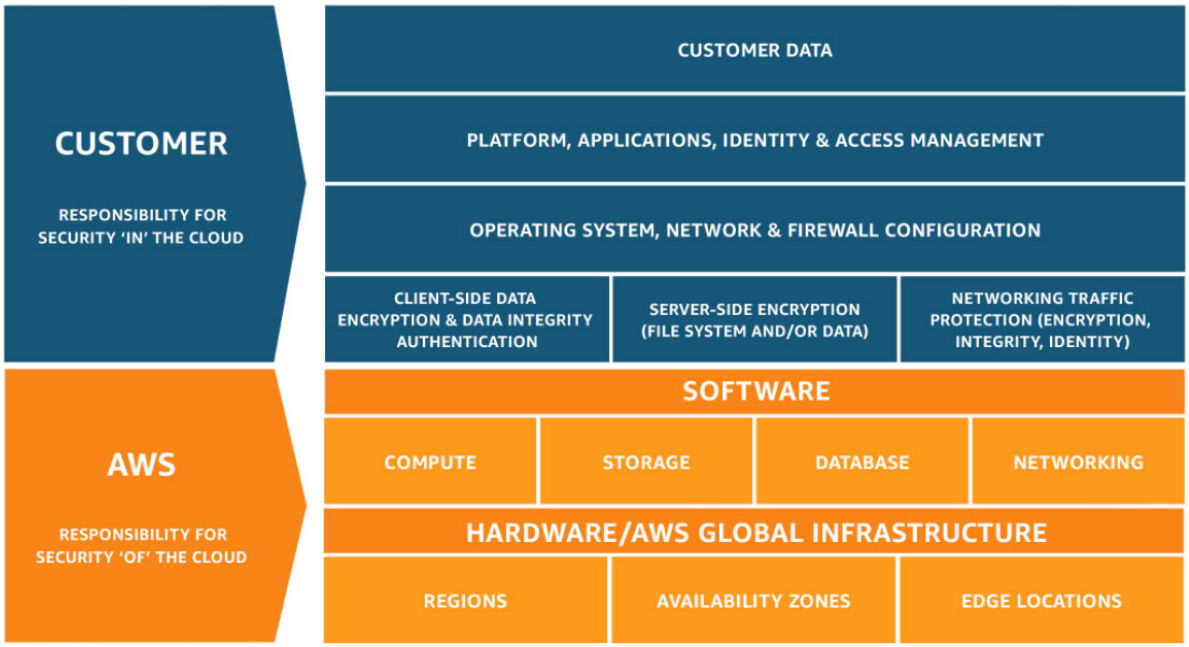
「コンプライアンスプログラムによる AWS 対象範囲内のサービス」ウェブサイトには、PCI DSS 年次評価に含まれた AWS のサービスとその他コンプライアンスプログラムの対象となるすべてのサービスを一覧表示しています。AWS のサービスのコンプライアンスは継続的に維持され、新機能の評価では、元のコンプライアンスステータスを継承できるかどうかが判断されます。新しいサービスが定期的に AWS ポートフォリオに追加されていますが、サービスが評価に追加されると、このリストにも表示されます。お客様は、AWS Artifact を使用して AWS マネジメントコンソールから AWS コンプライアンスのドキュメントにアクセスできます。

AWS 責任共有モデル

セキュリティとコンプライアンスは、AWS とお客様の責任共有です。責任共有モデルは、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、サービスを運用する施設

の物理的なセキュリティに至るまで、AWS が該当するコンポーネントの運用、管理、制御を引き受けるため、お客様の運用上の負担が軽減します。

次の図に、責任共有モデルの概要を示します。責任範囲は、実装した AWS のサービスによって異なる場合があります。



責任共有モデル

AWS は、AWS クラウド内で提供するすべてのサービスを実行するインフラストラクチャである**クラウドの**セキュリティとコンプライアンスについて責任を負います。クラウドセキュリティは AWS の最優先事項です。AWS のお客様は、セキュリティ要件が最も厳しい組織の要件を満たすように構築されたデータセンターやネットワークアーキテクチャの恩恵を享受します。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。

お客様は、お客様が設定したシステムと AWS 上でプロビジョニングされるサービスで構成される**クラウド内の**セキュリティとコンプライアンスについて責任を負います。PCI DSS コンプライアンスの場合、CDE に含まれているか、接続している AWS リソースを含むすべてのシステムコンポーネントがお客様の責任となります。Amazon Simple Storage Service (Amazon S3) や Amazon DynamoDB などの抽象化されたサービスの場合、アクセス制御、ログ設定、データライフサイクルポリシー、暗号化設定など、お客様が設定できる統制は責任に含まれます。

責任分担は、お客様が選択した AWS のサービスと実装によって異なります。Amazon EKS が良い例です。このサービスを使用すると、お客様は AWS Fargate でコンテナのサーバーレスデプロイを選択したり、お客様がアクセスできる Amazon EC2 インフラストラクチャでコンテナを実行したりできます。

AWS Fargate では、お客様は基盤となるホストから抽象化され、ホストシステムの更新やパッチ適用の責任を負いません。これは、より高いレベルの責任 (ホストアクセスの制御やセキュリティパッチの適用など) をお客様が負う必要がある、Amazon EC2 ホストを使用した Amazon EKS のデプロイとは対照的です。

お客様がサービスパラメータを制御できる場合、お客様は PCI DSS 要件を満たすようにパラメータを確実に設定する責任を負います。

PCI DSS の適用範囲の決定と検証

環境内のカード会員データ (CHD) の完全なフローを理解することが重要です。CHD フローは、PCI DSS の適否を決定し、CDE の境界とコンポーネントを定義します。また、これに伴って PCI DSS 評価の適用範囲を定義します。PCI DSS の適用範囲を正確に決定することは、評価されたワークロードのセキュリティ体制を定義し、最終的に評価を成功させるための鍵となります。お客様は、適用範囲の決定に関する手順を用意して、この手順に従って適用範囲の完全性を確保し、適用範囲の変更や逸脱を検出する

必要があります。通常、PCI DSS の適用範囲の識別は、次のステップで構成されます。

1. **CHD フローを識別する。**CHD のライフサイクルを定義します。これには、環境での CHD の消費またはエントリのパス、それに続く CHD の処理と保存、最終的には環境からの CHD のセキュアな破棄、取り扱いの変更、または終了が含まれます。
2. **環境内の対象範囲内のリソースをすべて識別する。**CHD フローを構成する CHD の受信、処理、保存、送信に関連するさまざまな種類の AWS リソースを識別します。
3. **システムを分類する。**システムを抽象化されたサービスとインフラストラクチャサービスに分類します。これらのリソースの対象範囲の識別とセグメンテーションは、さまざまな種類の接続に基づいています。つまり、インフラストラクチャサービス (OSI レイヤー 3～4) 接続と抽象化されたサービス (OSI レイヤー 7) です。
4. **セグメンテーションの境界を設計する。**CHD フローに関連しない他のすべての AWS リソースが CDE から隔離 (セグメント化) され、PCI DSS の適用範囲から除外されるようにセグメンテーションの境界を設計します。

コンテナはエフェメラルであり、動的にスコープが変わり続けることを考えると、さらに複雑になります。その結果、お客様は、コンテナのライフサイクルのすべての段階でコンプライアンス要件に確実に対処するために、すべてのコンテナ設定パラメータに常に注意する必要があります。これについては、次のセクションで示します。

Amazon EKS デプロイの保護

次のセクションでは、PCI DSS コンプライアンス向けにコンテナベースの環境を設計する際に考慮すべき重要なトピックに関するガイダンスを提供します。これらのセクションは、次のカテゴリで構成されます。

- ネットワークセグメンテーション
- ホストおよびコンテナイメージの強化
- データ保護
- ユーザーアクセスの制限
- イベントのログ記録
- 脆弱性のスキャンと侵入テスト

セクションごとに、要件の概要と、コンプライアンスを達成するためのベストプラクティスを示します。お客様ごとに環境は異なるため、ガイダンスは包括的なものではありません。次のレコメンデーションは、コンテナベースの環境を保護するための多層防御アプローチを提供します。

ネットワークセグメンテーション

PCI DSS の要件 1 の統制では、カード会員データを保護するためにファイアウォールをインストールして維持することを定め、システムを不正アクセスから保護することを義務付けています。このコンテキストにおけるファイアウォールは、インバウンドおよびアウトバウンドのアクセスを、承認されたポートとサービスのみに制限する必要があることを意味します。ネットワークセグメンテーションの使用は PCI DSS の要件では

ありませんが、お客様の環境の適用範囲を狭めるために、非常に関連性の高い手段です。

専用 AWS アカウントは、AWS プラットフォームで達成できる最高レベルのセグメンテーション境界を提供します。設計上、AWS アカウント内でプロビジョニングされたすべてのリソースは、お客様自身の AWS Organizations 内であっても、他の AWS アカウントでプロビジョニングされたリソースから論理的に分離されます。PCI DSS ワークロード用に分離されたアカウントを使用すると、AWS クラウドで実行する PCI アプリケーションを設計する際に、強力なアクセスのセグメンテーションを確立できます。

Amazon Virtual Private Cloud (VPC) とサブネットは、CDE 関連のリソースをさらに論理的に分離します。 手動設定を使用するか、`eksctl` または Amazon EKS が提供する AWS CloudFormation テンプレートを使用して VPC とサブネットをデプロイすることで、Amazon EKS の要件を満たす VPC とサブネットをデプロイできます。`eksctl` と AWS CloudFormation テンプレートはいずれも、必要な設定で VPC とサブネットを作成します。詳細については、「Amazon EKS クラスターの VPC を作成する」を参照してください。

Kubernetes クラスター内では、デフォルトですべての Pod 間通信が許可されています。Kubernetes の Network Policy では、Pod 間だけでなく、Pod と外部サービス間のネットワークトラフィックを制限するメカニズムを提供しています。Kubernetes の Network Policy は、開放型システム間相互接続 (OSI) モデルのレイヤー 3 およびレイヤー 4 で動作します。Network Policy は、Pod セレクターとラベルを使用して送信元 Pod と送信先 Pod を識別しますが、IP アドレス、ポート番号、プロトコル番号、またはこれらの組み合わせを含めることもできます。Calico は Tigera のオープンソースのポリシーエンジンで、Network Policy の適用を支援し、Amazon EKS と連携します。

Calico は、Kubernetes の Network Policy 機能のフルセットの実装に加えて、レイヤー 7 のルール (例えば、AWS AppMesh や Istio などのサービスメッシュと統合されている場合は HTTP) のサポートを含む、より豊富な機能セットで拡張された Network Policy をサポートしています。

Calico ポリシーの範囲は、Namespace、Pod、Service Account、またはグローバルに設定できます。ポリシーの範囲が Service Account に限定されると、その Service Account にインバウンド/アウトバウンドルールのセットが関連付けられます。適切な Kubernetes のロールベースアクセスコントロール (RBAC) ルールが用意されているので、チームがこれらのルールを上書きするのを防ぐことができ、IT セキュリティ専門家は Namespace の管理を安全に委任できます。Amazon EKS クラスターを初めてプロビジョニングする場合、Calico ポリシーエンジンはデフォルトではインストールされません。Calico をインストールするためのマニフェストは、VPC CNI リポジトリにあります。

AWS 内では、セキュリティグループが仮想ファイアウォールとして機能し、ステートフルな検査を行います。セキュリティグループを使用して、IP アドレス、ポート、プロトコルに基づいて通信を制限できます。デフォルトでは、セキュリティグループはすべてのアウトバウンド通信を許可することに注意してください。そのため、PCI DSS 要件を満たすようにアウトバウンド接続ルールを設定する必要があります。これは、セキュリティグループまたは AWS Network Firewall などのインラインセキュリティアプライアンスを使用して実行できます。

AWS Network Firewall は、高度にスケーラブルなフルマネージド型のインラインファイアウォールで、CDE を出入りするトラフィックを保護するために使用できます。標準のオープンソースルール形式を使用して、ステートレスでステートフルなファイアウォールルールがサポートされます。

また、パートナー統合の多様なエコシステムを使用して、脅威インテリジェンス、一元管理、マネージドルールセットを提供することもできます。AWS Network Firewall は、East-West (VPC から VPC) トラフィックと、North-South (VPC からインターネット、インターネットから VPC) トラフィックを保護するために、さまざまな方法でデプロイできます。

Amazon EKS は Amazon VPC のセキュリティグループを使用して、Kubernetes のコントロールプレーンとクラスターのワーカーノード間のトラフィックを制御します。セキュリティグループは、ワーカーノード間、外部 IP アドレス間、および他の VPC リソース間のトラフィックを制御するためにも使用されます。コントロールプレーンごとに専用のセキュリティグループを使用することを強くお勧めします (クラスターごとに 1 つ)。コントロールプレーンとノードグループのセキュリティグループの最小ルールと推奨ルールは、「Amazon EKS セキュリティグループに関する考慮事項」に記載されています。

クラスター内で実行されるサービスとクラスター外で実行されるサービス間の通信を制御するには、Amazon EC2 のセキュリティグループを Kubernetes の Pod と統合する、Pod のセキュリティグループを検討してください。

Amazon EC2 のセキュリティグループを使用して、多くの Amazon EC2 インスタンスタイプまたは AWS Fargate で実行されているノードにデプロイする Pod 間のインバウンドおよびアウトバウンドネットワークトラフィックを許可するルールを定義できます。サポートされているインスタンスの完全なリストについては、「Amazon EC2 でサポートされるインスタンスとブランチのネットワークインターフェイス」を参照してください。ノードは、サポートされているインスタンスタイプの 1 つである必要があります。Pod のセキュリティグループをデプロイする前に、Amazon EKS ユーザーガイドの「Pod のセキュリティグループ」で説明されている制限と条件について検討してください。

AWS Fargate は、各 Pod を専用のカーネルランタイム環境で実行し、CPU、メモリ、ストレージ、またはネットワークリソースを他の Pod と共有しないため、ワークロードの分離とセキュリティの向上に役立ちます。ただし、Kubernetes はシングルテナントのオーケストレーターであるため、Pod レベルの相互通信の可能性はまだ存在します。完全なセキュリティ分離が必要な機密性の高いワークロードの場合は、専用の Amazon EKS クラスターにグループ化するべきでしょう。

コンテナ化されたワークロードは、ネットワークセグメンテーションを容易にするために、データの機密性レベルに基づいてグループ化するのが最適です。さらに、Kubernetes の Namespace では、相互に論理的に分離された Kubernetes クラスター内でリソースのセグメンテーションが可能です。Namespace は、クラスター内の Pod、Service、Deployment の適用範囲を提供するため、1 つの Namespace を操作するユーザーには別の Namespace のコンテンツは表示されません。ただし、同じクラスター内の Namespace では、それらの間の通信は制限されません。きめ細かな制御を行ったり、Namespace の間でこのような通信を制限したりするには、Network Policy が必要になります。

結論として、コンテナ化されたアプリケーションの通信を分離する場合は、次を検討します。

- Service の機密性に基づいて Pod を個別のノードに分離し、専用のセキュリティグループを使用して別のクラスターに CDE ワークロードを分離します。
- AWS セキュリティグループを使用して、ノード間の通信およびコントロールプレーンと外部の通信を制限します。
- Kubernetes の Network Policy でマイクロセグメンテーションを実装し、サービスメッシュ、ネットワークおよび暗号化ライブラリ (NaCI) の暗号化、コンテ

ナネットワークインターフェイス (CNI) の使用を検討して通信を制限し、セキュアにします。

- Network Policy を利用して、ネットワークセグメンテーションとテナント分離の分離を実装します。Network Policy は、ネットワークの進入ルールと送信ルールを作成できる点で AWS セキュリティグループと似ています。インスタンスをセキュリティグループに割り当てる代わりに、Pod セレクターとラベルを使用して Pod に Network Policy を割り当てます。詳細については、「Amazon EKS への Calico のインストール」を参照してください。
- CDE の入出力がセキュリティで保護されていること、および許可されたフローが文書化されていることを確認します。

ホストとイメージの堅牢化

PCI DSS の要件 2 は、ベンダーが提供するシステムパスワードやその他のセキュリティパラメータに関して、ベンダーが提供するデフォルト値のサポートを無効にする必要性を強調しています。Amazon EKS などの Amazon のコンテナサービスは、コンテナに最適化された Amazon マシンイメージ (AMI) で実行されます。これらのオペレーティングシステムは、コンテナのデプロイに必須となるライブラリのみが含まれているため、攻撃ベクトルを最小限に抑えるのに役立ちます。

お客様は、オペレーティングシステム、ネットワーク、アプリケーションの各レイヤーで、すべての設定と機能のコンプライアンスを引き続き維持する責任があります。オペレーティングシステムには AWS Systems Manager を使用して定期的にパッチを適用する必要があります。一方、重要でないサービスやライブラリは無効化または削除する必要があります。Amazon EC2 インスタンスタイプの Center for Internet Security(CIS) のベンチマークなど、業界で認知されたシステム強化ガイドラインに準

拠した標準設定を確立する必要があります。AWS セキュリティの学習ウェブサイトには、AWS の安全な標準設定に関する追加のサポートがあります。

Bottlerocket などの特定の目的に特化したオペレーティングシステム (OS) の使用を検討します。これには、アタックサーフェスの縮小、起動時に検証されるディスクイメージ、SELinux を使用した許可の境界の適用が含まれます。

コンテナのビルドは、必要なリソースのみを含めるようにし、マイクロサービスのモデルを採用してコンテナが 1 つの主要な機能を提供するようにします。ソフトウェアアーキテクトは、イメージが古いソフトウェアライブラリやアプリケーションに依存していないことを確認する必要があります。これらには、既知の脆弱性が含まれている可能性があります。ベストプラクティスは、コンテナレジストリで定期的にコンテナイメージを再構築し、最新のアプリケーションバージョンを確実に使用することです。脆弱なライブラリを使用すると、見逃されがちな、悪意のあるアクティビティの侵入経路が生まれる可能性があります。

コンテナを管理する場合、コンテナはイミュータブルであること、およびインプレースでパッチを適用しないことが必要です。お客様は、信頼できるベースコンテナイメージ、つまりアセスメントを受け、パッチが適用されたライブラリとアプリケーションが利用されていることが確認されたベースイメージを作成する必要があります。コンテナイメージを保護するには、Amazon Elastic Container Registry (Amazon ECR) のような信頼されたレジストリを使用します。Amazon ECR は、Common Vulnerabilities and Exposures (CVE) データベースに基づくイメージスキャンを提供し、一般的なソフトウェアの脆弱性を特定できます。

Amazon Inspector は、AWS にデプロイされたアプリケーションのセキュリティとコンプライアンスを向上させるための、自動化されたセキュリティ評価サービスとして使用できます。Amazon Inspector では、自動的にアプリケーションを評価し、露出、脆弱性、ベストプラクティスからの逸脱がないかどうかを確認できます。

お客様は、Amazon EKS の Amazon EC2 の起動タイプを選択する際に、Amazon EC2 インスタンスが適切なアンチウイルスおよびファイル整合性モニタリングソフトウェアを実行していることを確認する責任があります。多くのコンテナベンダーは、これらの要件に対処するために、コンテナの使用に最適化したソリューションを提供しています。

または、コンテナ用にオンデマンドで適切なサイズのコンピューティング性能を提供する AWS Fargate 起動タイプの使用を検討してください。AWS Fargate では、コンテナを実行するために仮想マシンのグループをプロビジョニング、設定、またはスケールする必要がありません。このオプションにより、サーバータイプの選択、ノードグループをスケールするタイミングの決定、クラスターのパッキング (コンテナの詰め方) の最適化を行う必要がなくなります。AWS Fargate を使用する利点は、ホストシステムの強化やパッチ適用、監視、およびそれらによるワーカーノードの管理が AWS により行われることです。AWS Fargate の選択に関するその他の考慮事項については、Amazon EKS ユーザーガイドを参照してください。

Gatekeeper、Open Policy Agent (OPA)、dockerfile-lint などのツールを使用して、セキュリティとコンプライアンスのポリシーを適用できるポリシーガバナンスの実装を検討することをお勧めします。

結論として、ホストとイメージの堅牢化に関しては次を検討します。

- コンテナの実行に最適化された OS を使用します。
- ワーカーノードへのアクセスを最小限に抑え、プライベートサブネットにワーカーノードをデプロイします。
- Amazon Inspector を実行して、ホストの露出、脆弱性、逸脱を評価します。

- 最小限のコンテナイメージを使用し、定期的にイメージをスキャンして脆弱性を検出します。

データ保護

PCI DSS の要件 3 および 4 の統制は、保管中および伝送中の機密データを保護する必要性に重点を置いています。AWS は、PCI DSS に準拠しているサービスや機能を数多く提供しており、これらのコンプライアンスの取り組みを支援します。

カード会員データなどの機密データを含むワークロードでは、すべての保存データをセキュリティで保護する必要があります。データの保存は、基盤となるコンテナホストではなく、安全なファイルストアまたはデータベースで行う必要があります。システムアーキテクトは、ホストファイルシステムや一時ストレージなどのコンテナへのボリュームマウント、およびコンテナ間でのデータの共有に注意する必要があります。

コンテナのビルドファイル内のデータベース接続文字列などの機密データや環境変数は、必ずセキュリティで保護します。AWS の多くのサービスは AWS Key Management Service (AWS KMS) と統合されています。AWS KMS は、セキュアな暗号化キーストレージ、アクセスコントロール、年次のローテーションなどの暗号化キー管理機能を提供する、PCI DSS に準拠したサービスです。

AWS Secrets Manager と AWS Systems Manager Parameter Store は、コンテナのビルドファイル内の機密データを保護するために使用できる 2 つのサービスです。AWS Systems Manager Parameter Store は、データの安全な階層ストレージを提供し、そのためのサーバーの管理も不要にします。きめ細かなアクセスコントロールと監査統制を確立して、コンプライアンス要件を満たすために適切な制限が実施されていることを確認できます。AWS Systems Manager Parameter Store 内に保存されたデータは、AWS KMS を使用して暗号化できます。

AWS Systems Manager Parameter Store と同様に、AWS Secrets Manager 内の保護されたデータでも AWS KMS を使用します。AWS Secrets Manager には、ランダムなパスワード生成や自動パスワードローテーションなどの追加機能もあります。AWS KMS は PCI DSS に準拠しているサービスであり、多くの AWS プラットフォームサービスに統合されています。ユーザーは、暗号化キーマテリアルを作成および管理し、誰が暗号化キーにアクセスして使用できるかを制御できます。

伝送中のデータに関しては、PCI DSS ではオープンなパブリックネットワーク経由で伝送する機密情報は暗号化する必要があります。 お客様は、強力な暗号化およびセキュリティの統制を設定する責任があります。

AWS では、Transport Layer Security (TLS) の使用をサポートするために Amazon API Gateway や Application Load Balancer などの複数のサービスを提供しています。ポリシーをサービスに適用して、TLS 1.1 以上のみをサポートするように強制できます。

また、Amazon API Gateway と Application Load Balancer は、統合された AWS WAF (ウェブアプリケーションファイアウォール) の使用をサポートしており、アプリケーションレイヤーでの通信を保護します。AWS WAF は、OWASP Top 10 で特定されているような、一般的なウェブの悪用からアプリケーションと API を保護するために役立ちます。

Nitro インスタンスタイプ C5n、G4、I3en、M5dn、M5n、P3dn、R5dn、R5n の間で交換されるトラフィックは、AWS Transit Gateway やロードバランサーなどの中間ホップがある場合を除き、デフォルトで自動的に暗号化されます。これらの場合、トラフィックは暗号化されません。

Pod 間通信の転送時の暗号化は、mTLS をサポートする AWS App Mesh などのサービスメッシュを使用して実装することもできます。

インバウンドトラフィックコントローラーは、クラスターの外部から発生する HTTP/S トラフィックを、クラスター内で実行しているサービスにインテリジェントにルーティングする方法です。多くの場合、着信トラフィックは Classic Load Balancer や Network Load Balancer などのレイヤー 4 ロードバランサーの前面に置かれます。着信トラフィックコントローラーは、SSL/TLS 接続を終端するように設定できます。

結論として、データ保護に関しては次を検討します。

- サービス管理の暗号化キーに AWS KMS を使用し、AWS 管理の AWS KMS key を利用するか、定期的に AWS KMS key を更新します。
- 転送時の強力な暗号化のサポートを有効にします。
- EKS で Kubernetes シークレットのエンベロープ暗号化を使用して、Kubernetes クラスター内に保存されたアプリケーションシークレットまたはユーザーデータの暗号化のカスタマー管理レイヤーを追加します。
- ユーザーアクセス。

PCI DSS の要件 7 および 8 の統制は、権限を与えられた担当者のみにアクセスを制限し、適切なアクセス制御を確実に設定することに重点を置いています。リソースへのアクセス権は、職務の実行に必要な最小特権モデルに基づく必要があります。コンテナおよび基盤となるホストへのユーザーアクセスは、PCI DSS に準拠した厳格な認証要件に従って認証する必要があります。

コンテナイメージは、特権のあるユーザー以外のアカウントで実行する必要があります。例えば、コンテナのビルドファイルに実行ユーザーの認証情報が定義されていない場合、そのコンテナはデフォルトで、root で実行されます。このセットアップでは、侵害されたコンテナサービスは、root 権限を攻撃者に拡張する可能性があります。攻

撃者は、昇格されたアクセスを使用して基盤となるホストをさらに悪用する場合があります。

ホストアクセスを使用できない AWS Fargate とは異なり、Amazon EC2 の起動タイプを持つ Amazon EKS には、基盤となるシステムを管理するための Secure Shell (SSH) アクセスを有効にするオプションがあります。Secure Shell (SSH) の使用を無効にし、代わりに AWS Systems Manager の Run Command を使用することを検討してください。Run Command は、管理対象の SSH キーを持たず、呼び出したすべての操作は AWS CloudTrail 内で監査できます。

安全なコンテナイメージを作成して確立するために、コンテナイメージへのすべてのアクセスを制限します。コンテナのデプロイでは、アクセスと書き込みの許可を制限するプライベートコンテナレジストリを使用する必要があります。例えば、Amazon ECR を AWS Identity and Access Management (IAM) と統合してアクセスを制御します。Amazon ECR は、コンテナイメージの安全な保存と伝送を提供する、スケーラブルなコンテナレジストリです。Amazon ECR の簡素化されたワークフローと、AWS のサービスとの統合により、コンテナホストへの認証アクセスを過度に提供する必要性も減ります。

Amazon EKS とその必須の AWS IAM Authenticator により、ユーザーは AWS IAM の ID (IAM ユーザーまたは IAM ロール) を使用してクラスターにサインインします。その後、Kubernetes は、ロールベースのアクセスコントロール (RBAC) を通じてユーザーが実行できるアクションを決定します。クラスターおよびインフラストラクチャレベルで認証をセットアップするには、IAM ロールを設定する必要があります。RBAC では、リソースレベル (例: 特定の Pod) またはサービスレベル (例: Pod 全体) に対して認証を設定できます。Namespace レベルのリソースに Roles または RoleBindings を作成し、クラスターレベルのリソースに ClusterRole または ClusterRoleBindings を作成する場合は、最小特権アクセスを採用します。

Amazon EKS クラスターを作成すると、クラスターの RBAC 設定で、AWS IAM のエンティティユーザーまたはロール (クラスターを作成するフェデレーティッドユーザーなど) に system:masters 許可が自動的に付与されます。このアクセスは削除できず、aws-auth ConfigMap を通じて管理されません。したがって、専用の IAM ロールを使用してクラスターを作成し、このロールを引き受けることができるユーザーを定期的に監査することがベストプラクティスです。このロールは、クラスターでルーチンアクションを実行するために使用しないでください。このためには、代わりに aws-auth ConfigMap を通じてクラスターへのアクセスを追加のユーザーに付与する必要があります。詳細については、「クラスターのユーザーまたは IAM ロールの管理」を参照してください。

結論として、ユーザーアクセスに関しては次を検討します。

- RoleBindings および ClusterRoleBindings の作成時に、AWS リソースへの最小特権アクセスを採用します。
- クラスターと、可能な場合はサービスアカウントの IAM ロール (IRSA) に対して複数のユーザーが同一のアクセスを必要とする場合は、IAM ロールを使用します。
- Amazon EKS クラスターエンドポイントをプライベートにします。
- 定期的に監査する必要がある専用 IAM ロールでクラスターを作成します。
- クラスターへのアクセスを定期的に監査します。
- 非ルートユーザーとしてアプリケーションを実行します。

アクセスの追跡とモニタリング

イベントのログ記録

PCI DSS の要件 10 内の中核となる統制は、イベントログメカニズムを使用して、潜在的に異常なアクティビティを追跡、監視、および警告する必要性です。コンテナ化された動的な環境では、堅牢で一元化されたログ記録インフラストラクチャを維持し、保持と分析のために、ログがコンテナからすぐにセキュアなストアに送信されるようにすることが不可欠です。

AWS イベントログサービスを使用して、ネットワーク、ホスト、コンテナレベルでのイベントログのモニタリングを設定します。VPC フローログを有効にして、プロトコル、ポート、送信元アドレス、送信先アドレス情報など、パケット情報の詳細を示すネットワークトラフィックをキャプチャします。コンテナホストをモニタリングして、Amazon CloudWatch エージェントや Amazon Kinesis エージェントを確実に有効化および設定することで、健全性、効率性、および可用性を確保します。

コンテナ化されたアプリケーション内でイベントログ機能を有効にして、アプリケーションとコンテナのイベントログデータをキャプチャします。Amazon CloudWatch ダッシュボードを使用して、キャプチャされたすべてのイベントログアクティビティをモニタリングし、アラートを送信します。キャプチャされたイベントデータを暗号化された Amazon S3 バケット内にセキュアに保存して、保持要件を満たします。

Amazon EKS と AWS Fargate では、組み込みのログルーターがサポートされています。つまり、インストールや保守に必要なサイドカーコンテナはありません。ログルーターを使用すると、AWS の幅広いサービスをログの分析とストレージに使用できます。AWS Fargate から Amazon CloudWatch、Amazon OpenSearch Service、Amazon Kinesis Data Firehose の送信先 (Amazon S3、Amazon Kinesis Data

Streams、パートナーツールなど) に直接ログをストリーミングできるようになりました。AWS Fargate は、AWS によって管理される Fluent Bit のアップストリーム準拠ディストリビューションである、Fluent Bit のバージョンを使用しています。詳細については、GitHub の「AWS for Fluent Bit」を参照してください

AWS ツールを使用して、環境の全体像を維持する必要があります。Amazon GuardDuty は、AWS CloudTrail、VPC フローログなど、AWS データソース全体の異常検出、機械学習、イベントの脅威インテリジェンスを通じて脅威検出を提供します。Amazon Athena と Amazon CloudWatch Logs Insights を使用すると、VPC フローログ、AWS CloudTrail、および Amazon CloudWatch から、Amazon S3 に保存した監査証跡ログにクエリを実行して分析することもできます。

アクセスが厳しく制限され、AWS CloudTrail やアプリケーションログを含むすべてのセキュリティログおよび運用ログが保存される、専用の監査アカウントを使用することを強くお勧めします。このアカウント内では、データを保持し、改ざんされないようにするため、Amazon S3 の MFA Delete、Amazon S3 のファイルバージョニング、Amazon S3 のライフサイクルポリシーなどの設定オプションの使用を検討してください。同様に、AWS Organizations とサービスコントロールポリシーを使用して、セキュリティログ記録設定などの重要な設定を変更または無効にできないようにすることをお勧めします。

最後に、AWS アカウント内のリスクとコンプライアンスの評価をサポートするツールの使用を検討してください。AWS Audit Manager は、事前構築されたフレームワークを通じて最新の PCI DSS v3.2.1 をサポートし、自動化された証跡収集や監査対応レポートなどの機能を提供します。

結論として、モニタリングとログ記録に関しては次を検討します。

- Amazon EKS クラスター監査ログを有効にします。

- 認証履歴の追跡に Kubernetes 監査メタデータの注釈を使用します。
- 不審なイベントのアラームを作成します。
- Amazon CloudWatch Log Insights でログを分析します。
- AWS CloudTrail ログを監査します。
- AWS Organizations とサービスコントロールポリシーを使用して、セキュリティコントロールを回避または無効にできないようにします。

ネットワーク侵入検出

PCI DSS の要件 11 内の統制では、ネットワークへの侵入を検出して防止するために、侵入検出および侵入防止手法の使用を指定しています。基準では、CDE の境界および重要なポイントを通過するすべてのトラフィックをモニタリングすることを義務付けています。ほとんどのオンプレミス環境の場合、これらの要件を満たすには、通常、侵入検出システム (IDS) および侵入防止システム (IPS) アプライアンスを使用します。AWS 内でも同様のアプローチを使用できます。

コンテナ化された環境を考慮した場合、ネットワークトラフィックの検査は、コンテナホスト外のネットワークレイヤー、およびコンテナ管理ソフトウェアの仮想コンテナネットワーク内で実施できます。

AWS のコンテナホスト外のネットワークデータを検査するために検討できるオプションがいくつかあります。Amazon GuardDuty は、複数の AWS データソースにわたって脅威を検出および特定するマネージドサービスです。機械学習、異常検出、脅威インテリジェンスを使用して、違法なネットワークアクティビティを識別します。

従来の IDS/IPS ソリューションを検討する場合は、Amazon VPC トラフィックミラーリングの設定を通じて、1 つ以上の Amazon EC2 インスタンスで実行している仮想アプイアンスに、すべてのネットワーク通信のコピーをルーティングできます。

もう 1 つの一般的なソリューションとしては、IP ルーティングを使用するトランジットネットワークアーキテクチャを使用して、すべてのネットワークトラフィックが 1 つのネットワークを通過するようにします。このアーキテクチャでは、AWS Marketplace から入手した仮想 IDS/IPS デバイスを使用して、ネットワーク間を通過するすべてのトラフィックを検査できます。Amazon VPC のゲートウェイを使用して、すべてのトラフィックをオンプレミスの IDS/IPS インフラストラクチャにルーティングすることもできます。最後に、ホストベースの IDS または IPS ソリューションを使用して、Amazon EC2 インスタンスへの配信時にトラフィックを検査することもできます。

別の実行可能なオプションとして、仮想コンテナネットワーク上のコンテナ間の通信を検査できます。AWS Marketplace には、IDS コンテナソリューションを提供するベンダーが登録されています。これらのベンダーは、主にサイドコンテナを使用して異常なトラフィックパターンをモニタリングし、アラートで通知します。また、機械学習を使用してコンテナ間の異常な通信パターンを検出するエージェントベースのソリューションも利用できます。

どのセキュリティ対策を使用するかは、環境のアーキテクチャによって大きく異なります。ネットワークレイヤーでのトラフィック検出には、コンテナのデプロイとトラフィックパターンの高度な計画が必要です。

脆弱性のスキャンと侵入テスト

PCI DSS の統制要件 11.2 では、組織が定期的にシステムやプロセスをテストして、脆弱性を特定し、特定した結果を適時に修正することを義務付けています。脆弱性スキ

ャンは四半期ごとに実行します。また、環境に重要な変更を加えた後も実行する必要があります。同様に、侵入テストを毎年 1 回および環境の大幅な変更の後で実施します。AWS リソースの侵入テストは、許可された特定のサービスに対していつでも許可されます。侵入テストに関する AWS サポートポリシーの詳細を参照するか、アカウントチームにお問い合わせください。

ネットワークセグメンテーションを使用しているサービスプロバイダーは、セグメンテーション統制の有効性を 6 か月ごと、またはセグメンテーション統制を変更した後にテストする必要があります。

評価アクティビティの適用範囲には、CDE と、CDE のサポートに使用する補助システムが含まれます。侵入テストを実行する際の適用範囲と方法論のガイダンスについては、「PCI DSS Information Supplement: Penetration Testing Guidance」を参照してください。

お客様の環境によっては、テスト要件がオンプレミス、クラウドリソース、およびコンテナ化された環境に適用される場合があります。Amazon EKS を Amazon EC2 インスタンスにデプロイする場合、お客様は基盤となるホストの脆弱性スキャンを実行する必要があります。

PCI DSS 要件によれば、セキュリティの脆弱性を特定するプロセスを確立し、新たに発見されたセキュリティの脆弱性にリスクのランクを割り当てることは、お客様の責任となります。Amazon Inspector はセキュリティ評価ツールであり、脆弱性を特定し、特定した結果に対して重要度のレベルに基づく優先順位を付けます。DevOps プロセス内に Amazon Inspector を統合すると、評価の自動化により、脆弱性を事前に特定し、ノードの意図しないネットワークアクセシビリティをチェックできます。

また、コンテナ固有のスキャンツールを使用してコンテナイメージをスキャンし、脆弱性を検出することもできます。コンテナのスキャンでは、準拠していないコード、脆弱

なライブラリ、漏洩した可能性があるシークレットを特定します。AWS Marketplace 内のセキュリティベンダーは、システム、コンテナ、アプリケーションをスキャンできるソリューションを提供しています。

内部および外部の侵入テストを実行する場合、評価アクティビティはネットワークレイヤーとアプリケーションレイヤーの両方で行い、基盤となるホストとコンテナ化されたアプリケーションを対象にする必要があります。コンテナホストにパッチを適用して脆弱性に対処し、コンテナイメージを更新して特定したコンテナの脆弱性を軽減します。コンテナのゴールデンイメージを作成し、Amazon ECR などのプライベートコンテナレジストリ内に安全に保存します。

Center for Internet Security (CIS) Kubernetes Benchmark は、Amazon EKS ノードのセキュリティ設定のガイダンスを提供します。このベンチマークは、Kubernetes クラスターで CIS ベンチマークを使用して設定をチェックするための標準のオープンソースツールである kube-bench を使用して実行できます。詳細については、「CIS Amazon EKS ベンチマークの紹介」を参照してください。

まとめ

AWS は、お客様のコンテナ化されたワークロードをサポートするために複数のサービスを提供しており、お客様はデータ処理ニーズに最も適合するように、これらのサービスを設定できます。この柔軟性により、組織は AWS 責任共有モデルに従い、コンテナデプロイのライフサイクル全体を通じてすべてのコンプライアンス要件に絶えず注目する必要があります。本書で説明しているセキュリティ対策の方法は、お客様のコンテナ化されたワークロードに関する PCI DSS コンプライアンス要件に対処するのに役立ちます。

寄稿者

本書の執筆に当たり、次の人物および組織が寄稿しました。

- Arindam Chatterji (シニアソリューションアーキテクト、AWS)
- Tim Sills (ソリューションアーキテクト、AWS)
- Dave Barton (プリンシパルソリューションアーキテクト、AWS)

ドキュメントの改訂

日付	説明
2021 年 12 月 4 日	第 2 版
2021 年 6 月 25 日	初版発行