Autonomous Driving Data Framework (ADDF) セキュリティと運用ガイド

AWS 規範ガイダンス

AWS 規範ガイダンス: Autonomous Driving Data Framework (ADDF) セキュリティと運用ガイド

Table of Contents

序章 1
対象者 1
ターゲットを絞ったビジネス成果 2
アーキテクチャと用語 3
ADDF の用語 3
ADDF アーキテクチャ 5
責任共有モデル 9
AWS の責任 10
ADDF コアチームの責任 11
ADDF ユーザーの責任 11
AWS アカウント アカウント全般の責任 12
ADDF 固有の責任 12
セキュリティ検査プロセス 14
AWS による定期的なセキュリティ検査 14
オープンソースのセキュリティ検査とコントリビューション 14
あらかじめ組み込まれたセキュリティ機能 15
ADDF モジュールコードの最小特権 15
Infrastructure as Code 16
IaC の自動セキュリティチェック 16
AWS CDK デプロイロールのカスタムの最小特権ポリシー 16
モジュールの deployspec ファイルの最小特権ポリシー 17
データ暗号化 18
Secrets Manager を使用した認証情報ストレージ 18
SeedFarmer と CodeSeeder のセキュリティ検査 18
CodeSeeder の AWS CodeBuild ロールのための、アクセス許可の境界サポート 18
AWS マルチアカウントアーキテクチャ 19
マルチアカウントデプロイの最小特権のアクセス許可 20
安全なセットアップと運用 23
ADDF アーキテクチャを定義する 23
PoC (概念実証) 環境で ADDF を実行する 23
本番環境で ADDF を実行する 24
初期セットアップ 28
ADDF デプロイフレームワークコードをカスタマイズする 29
ADDF でカスタムモジュールを作成する 30

ADDF の反復的なデプロイ 30
反復的なセキュリティ監査 30
ADDF の更新 30
廃止作業 30
次のステップ 32
リソース 33
AWSドキュメント 33
オープンソースのリソース 33
注意 34
ドキュメント履歴 35
用語集 36
..... 36
A 37
B 40
C 42
D 45
E 49
F 51
G 52
H 53
I 54
L 56
M 57
O 61
P 64
Q 66
R 67
S 69
T 73
U 74
V 75
W 75
Z 76
..... lxxviii

Autonomous Driving Data Framework (ADDF) セキュリティと運用ガイド

Andreas Falkenberg、Junjie Tang、Torsten Reitemeyer、Srinivas Reddy Cheruku (Amazon Web Services (AWS))

2022 年 11 月 (ドキュメント履歴)

Autonomous Driving Data Framework (ADDF) は、一元化されたデータストレージ、データ処理パイプライン、視覚化メカニズム、検索インターフェイス、シミュレーションワークロード、分析インターフェイス、事前構築済みダッシュボードの設定など、先進運転支援システム (ADAS) の一般的なタスクを実装したいと考えている自動車チームに、再利用可能なモジュール式コードアーティファクトを提供することを目的としたオープンソースプロジェクトです。ADDF を使用すると、完全にカスタマイズ可能なモジュールを共有、変更、または作成できるため、これらのソリューションの作成とデプロイに必要な労力を削減できます。

このガイドは、ADDF を AWS クラウド 上で安全にデプロイして運用するためのベストプラクティスを理解していただくためのものです。以下のトピックについて説明します。

- アーキテクチャと用語 — 一般的なアーキテクチャ、ワークフロー、重要な用語を確認します。
- 責任共有モデル — ADDF のデプロイとクラウドリソースのセキュリティを保護する際の自分の役割と AWS の役割を理解する。
- セキュリティ検査プロセス — ADDF はオープンソースのプロジェクトであるため、AWS と寄稿者がどのようにセキュリティレビューを完了しているかを確認する。
- あらかじめ組み込まれたセキュリティ機能 — セキュリティのベストプラクティスと機能が ADDF オープンソースプロジェクトとそのデプロイフレームワークにどのように組み込まれているかを確認する。
- 安全なセットアップと運用 — ADDF を AWS クラウド 上にデプロイして運用する方法を学習する。

対象者

このガイドは、ADDF の評価、デプロイ、カスタマイズ、運用を担当する開発運用 (DevOps) チーム、インフラストラクチャエンジニア、管理者、IT セキュリティスタッフ、インシデント対応チームを対象としています。このガイドの推奨事項は、PoC (概念実証) 環境または本番環境に適用できます。

このガイドは、ADDF に関する予備知識を読者が有していないことを前提としています。ただし、先に進む前に、「ADDF readme」(GitHub) を読むことをお勧めします。

ターゲットを絞ったビジネス成果

このガイドは、開発環境と本番環境で ADDF をより確実にかつ安全にセットアップして運用できるようにするためのものです。

ADDF のアーキテクチャと用語

本ガイドのセキュリティとオペレーションのトピックについて理解するときは、Autonomous Driving Data Framework (ADDF) の用語、コンポーネント、アーキテクチャについて大まかに理解しておくことが重要になります。このセクションでは以下のトピックを取り上げます。

- ADDF の用語
- ADDF アーキテクチャ

ADDF の用語

ADDF の主要な用語は以下のとおりです。

- ADDF モジュール — このモジュールは、先進運転支援システム (ADAS) に共通するタスクを実装する Infrastructure as Code (IaC) です。共通するタスクには、一元化されたデータストレージ、データ処理パイプライン、可視化のメカニズム、検索インターフェイス、シミュレーションワークロード、分析インターフェイス、事前に作成されたダッシュボードなどの設定が含まれます。モジュールは独自の要件を元に新たに作ることもできれば、既存のものを再利用またはカスタマイズすることもできます。

  ADDF モジュールを定義するときは AWS Cloud Development Kit (AWS CDK) を使用できます。また、ADDF モジュールを実装するときは、Hashicorp Terraform や AWS CloudFormation などの一般的な IaC フレームワークを使用できます。モジュールには、入力パラメータのセットがあります。入力パラメータは、他のモジュールの出力値に依存します。ADDF のモジュールは、ADDF ターゲット AWS アカウント アカウントの、デプロイの最小単位です。
- ADDF デプロイマニフェストファイル — このファイルは、スタンドアロンの ADDF モジュールのオーケストレーションを定義します。オーケストレーションとは、モジュールをデプロイする順序のことです。ADDF のデプロイマニフェストファイルでは、ADDF グループを使って関連するモジュールを 1 つにまとめることができます。このファイルでは、ADDF ツールチェーン AWS アカウント、ADDF ターゲット AWS アカウント、ターゲット AWS リージョン も定義することができます。
- ADDF デプロイフレームワーク - このフレームワークは、ADDF モジュールを、ADDF デプロイマニフェストファイルで定義されたオーケストレーションに基づいて ADDF ターゲット AWS アカウント にデプロイします。ADDF デプロイフレームワークは、以下の AWS オープンソースプロジェクトを使用して実装します。

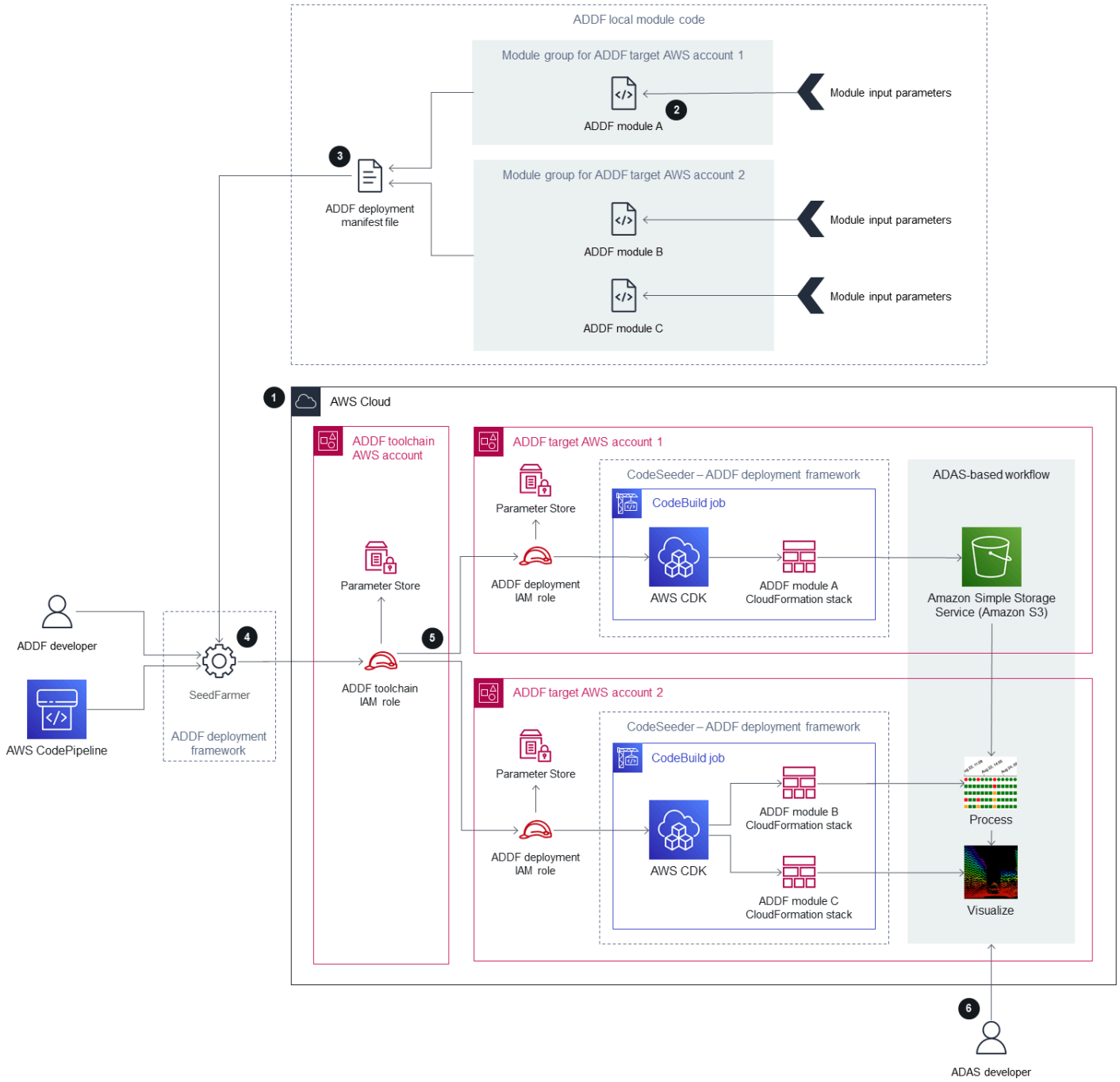
- SeedFarmer (GitHub) — SeedFarmer は、ADDF デプロイに使用される CLI ツールです。各モジュールの状態を管理し、モジュールコードの準備とパッケージ化を行い、ADDF デプロイロールの最小特権ポリシーを作成し、CodeSeeder がデプロイに使用するセマンティックを指示します。SeedFarmer は、直接連携して ADDF デプロイを実行することもできれば、継続的インテグレーションと継続的デプロイ (CI/CD) のパイプラインに統合して使用することもできます。
- CodeSeeder (GitHub) — CodeSeederは、任意のインフラストラクチャをコードパッケージとして、AWS CodeBuild のジョブを介してデプロイします。SeedFarmer は CodeSeeder を自動的にオーケストレーションし、実行します。CodeSeeder と直接連携できるのは SeedFarmer だけです。

ADDF のデプロイフレームワークは、シングルアカウントおよびマルチアカウントのアーキテクチャでのデプロイをサポートするように設計されています。シングルアカウントとマルチアカウントのどちらのアーキテクチャが必要かは、ご自身の組織の要件に基づいて判断します。

- ADDF ツールチェーン AWS アカウント — このアカウントは、ADDF デプロイマニフェストファイルの定義に基づいて ADDF ターゲット AWS アカウント へのモジュールのデプロイをオーケストレーションし、管理します。1 回の ADDF デプロイで使用できる ADDF ツールチェーン AWS アカウント は 1 つのみです。シングルアカウントのアーキテクチャの場合、ADDF ツールチェーン AWS アカウント は ADDF ターゲット AWS アカウント でもあります。このアカウントには ADDF ツールチェーン IAM ロールと呼ばれる AWS Identity and Access Management (IAM) ロールが含まれていますが、ADDF デプロイプロセス中は SeedFarmer がその役割を引き受けます。本ガイドでは、ADDF ツールチェーン AWS アカウント をツールチェーンアカウントと呼んでいます。
- ADDF ターゲット AWS アカウント — こちらは、ADDF モジュールをデプロイするターゲットアカウントです。ターゲットアカウントは 1 つまたは複数使用することができます。これらのアカウントには、ADDF デプロイマニフェストファイルとマップされたそのモジュールに記述された、リソースとアプリケーションロジックが含まれています。シングルアカウントのアーキテクチャの場合、ADDF ターゲット AWS アカウント は ADDF ツールチェーン AWS アカウント でもあります。各 ADDF ターゲットアカウントには、ADDF デプロイ IAM ロールと呼ばれる IAM ロールが含まれていますが、デプロイプロセス中は CodeSeeder がその役割を引き受けます。本ガイドでは、ADDF ターゲット AWS アカウント をターゲットアカウントと呼んでいます。
- ADDF インスタンス — ADDF とモジュールをクラウドにデプロイすると、ADDF デプロイマニフェストファイルで定義されているとおり、こちらが ADDF インスタンスになります。ADDF インスタンスでは、シングルアカウントとマルチアカウントのいずれのアーキテクチャでも使用でき、複数の ADDF インスタンスをデプロイすることができます。インスタンス数の選び方とユースケースに応じたアカウントアーキテクチャの設計方法の詳細については、「ADDF アーキテクチャを定義する」を参照してください。

ADDF アーキテクチャ

以下は、AWS クラウド における ADDF インスタンスのアーキテクチャの概要図です。こちらに示すのはマルチアカウントのアーキテクチャで、専用のツールチェーンアカウントと 2 つのターゲットアカウントが含まれています。本ガイドでは、ADDF を使用してターゲットアカウントにリソースをデプロイするプロセスを最初から最後まで説明します。



1. ADDF AWS アカウント を作成し、ブートストラップします。

   正常に機能させるには、各アカウントを ADDF と AWS CDK にブートストラップする必要があります。ADDF の新規デプロイである場合、または新しいターゲットアカウントを追加する場合は、次の操作を行います。

   a. ツールチェーンアカウントと各ターゲットアカウントの AWS CDK をブートストラップします。手順については「Bootstrapping」(AWS CDK ドキュメント) を参照してください。ADDF では、インフラストラクチャのデプロイに AWS CDK を使用します。

   b. ツールチェーンアカウントと各ターゲットアカウントの ADDF をブートストラップします。手順については「ADDF デプロイガイド」の「Bootstrap AWS アカウント(s)」を参照してください。これにより、SeedFarmer と CodeSeeder で必要になる ADDF 固有の IAM ロールがすべてセットアップされます。

   > **Note**
   >
   > 上記のステップは、ADDF を初めてデプロイする場合か、新しいターゲットアカウントを追加する場合のみ実行します。作成済みの ADDF インスタンスに再度 ADDF をデプロイする場合は、このステップを実行する必要はありません。

2. ADDF モジュールを作成またはカスタマイズします。

   ADDF モジュールを、解決しようとしている特定の問題に基づいて作成またはカスタマイズします。モジュールは、分離されたタスク、またはタスクのグループを代表するものである必要があります。必要であれば、モジュールの入力パラメータを定義し、そのモジュールの出力値を、他のモジュールの入力パラメータとして使用します。

3. モジュールオーケストレーションを ADDF デプロイマニフェストファイルで定義します。

   ADDF マニフェストファイルでモジュールをグループにまとめ、デプロイの順番と、モジュール間の依存関係を定義します。このファイルでは、1 つのツールチェーンアカウントと、各 ADDF グループおよびそのモジュールのターゲットアカウント (AWS リージョン を含む) も指定します。

4. ADDF デプロイマニフェストファイルを評価し、デプロイの範囲を指定します。

   ADDF のデベロッパーまたは CI/CD パイプライン (AWS CodePipeline など) が、CLI ツールの SeedFarmer を呼び出して ADDF デプロイマニフェストファイルの評価を開始します。評価を開始するため、

- SeedFarmer は、ADDF デプロイマニフェストファイルを評価の入力パラメータとして使用します。
- ADDF ツールチェーン IAM ロールを引き受けるため、SeedFarmer は、ステップ 1 の ADDF のブートストラップ中に定義したものと同じ、有効な IAM ロールまたはユーザー認証情報を要求します。

SeedFarmer が ADDF ツールチェーン IAM ロールを引き受けるための正しい認証情報を持っていないか、ADDF デプロイマニフェストファイルにアクセスできない場合、評価は開始されません。

評価を開始できる場合、SeedFarmer はツールチェーンアカウントの ADDF ツールチェーン IAM ロールを引き受けます。このアカウントの ADDF デプロイメント IAM ロールを引き受けることで、これ以降 SeedFarmer はどのターゲットアカウントにもアクセスできるようになります。次に SeedFarmer は、ツールチェーンアカウントとターゲットアカウントの ADDF メタデータを読み取ろうとします。次のいずれかの結果になります。

- 読み取るべき ADDF メタデータがない場合は、これが新しい ADDF インスタンスとなります。SeedFarmer は、デプロイ範囲は ADDF デプロイマニフェストファイルとその内容全体であると判断します。
- ADDF メタデータが存在する場合、SeedFarmer は、ADDF デプロイマニフェストファイルとその内容を、ターゲットアカウントにある既存のデプロイ済みのアーティファクトの、MD5 ハッシュと比較します。デプロイ可能な変更が検出された場合、プロセスは継続します。検出されなければ、プロセスはそこで完了します。

5. 対象に含まれる ADDF モジュールをターゲットアカウントにデプロイします。

CodeSeeder には、ADDF デプロイマニフェストファイルと、前のステップの評価結果に従って、実行すべきデプロイの順番付きリストが作成されています。CodeSeeder は、このリストに基づいて、関連する各ターゲットアカウントで ADDF デプロイ IAM ロールを引き受けます。次に CodeSeeder を、ADDF モジュール用の AWS CDK アプリケーションなど、個々の IaC デプロイを作成または更新する AWS CodeBuild ジョブで実行します。デフォルトでは、ADDF は IaC フレームワークとして AWS CDK を使用しますが、他の一般的な IaC フレームワークもサポートされています。このプロセスが各ターゲットアカウントで完了すると、完全にデプロイされた、クロスアカウントでエンドツーエンドの、ADAS ベースのワークフローが ADDF デプロイマニフェストファイルで定義したとおりに完成します。

シングルアカウントのアーキテクチャを使用する場合、ツールチェーンアカウントとターゲットアカウントは同じアカウントになり、説明した機能は一方のアカウントにすべて含まれます。

6. ADDF でデプロイされたインフラストラクチャを使用します。

ADAS のデベロッパーは、デプロイした ADAS ベースのワークフローを、ユースケースで定義したとおりに使用できます。

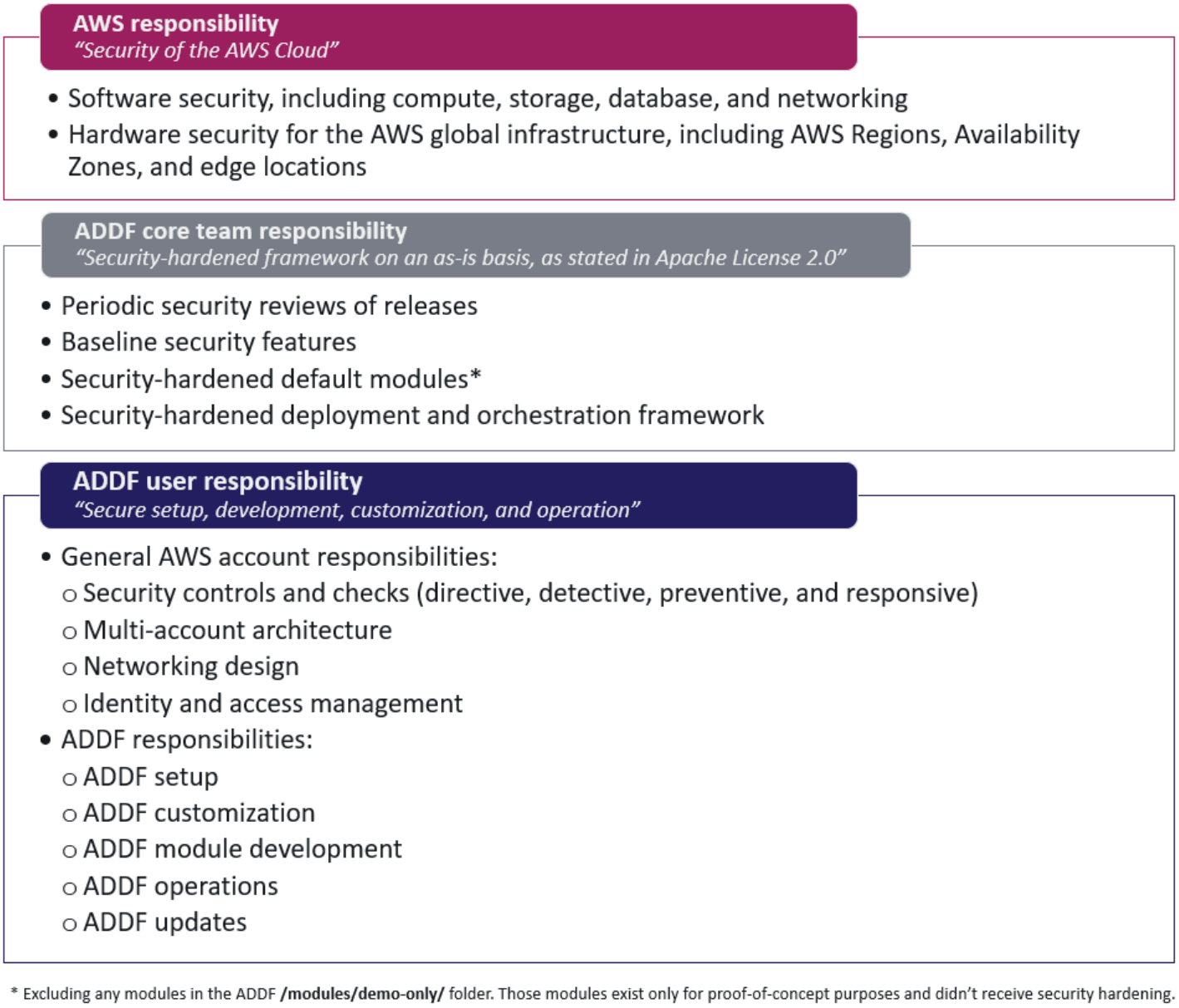
このワークフローには、ADDF マルチアカウント環境の、単一インスタンスのアーキテクチャが記述されています。開発、デプロイ、オペレーションのモデルによっては、多段階の環境で複数の ADDF インスタンスを実行することが推奨されます。一般的な設定には、開発、テスト、本番用のブランチなど、各デプロイステージ専用の ADDF インスタンスが専用の AWS アカウント と共に含まれている場合があります。ADDF のインスタンスごとに一意のリソース名前空間を作成した場合は、同じ AWS リージョン にある、同じシングルアカウントまたはマルチアカウントの環境で、複数の ADDF インスタンスを実行することも可能です。詳細については、「ADDF アーキテクチャを定義する」を参照してください。

ADDF 責任共有モデル

AWS のサービス に適用される責任共有モデルは、Autonomous Driving Data Framework (ADDF) にも適用されます。次の図に示すように、以下のエンティティが ADDF を保護する責任を共有しています。

- AWS — AWS のサービス を提供するクラウドインフラストラクチャプロバイダーです。
- ADDF コアチーム — ADDF コアチームは ADDF リリースを ADDF リポジトリ (GitHub) に公開するエンティティです。
- ADDF ユーザー — ADDF ユーザーには以下が含まれますが、これらに限定されません。
  - ADDF デベロッパー — ADDF モジュールコードを変更、カスタマイズ、または新規作成する人。
  - ADDF オペレーター — ADDF インスタンスをセットアップして運用する人。
  - ADAS デベロッパー — ADDF がデプロイするリソースのエンドユーザーまたはコンシューマー。例えば、ADAS 開発者は、ADDF デプロイの一環として作成されたビジュアライゼーションのフロントエンドにクエリを実行できます。

次の図は、AWS、ADDF コアチーム、ADDF ユーザー間で共有される責任をまとめたものです。



AWS の責任

AWS は、「AWS の責任共有モデル」で定義されているように、AWS クラウド 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 サービスを実行するハードウェア、ソフトウェア、ネットワーク、施設で構成されます。

ADDF コアチームの責任

ADDFコアチームは、「Apache License 2.0」(GitHub) に従い、ベストエフォート方式でそれ自体が安全なフレームワークを提供します。ADDF コアチームには以下の責任があります。

- リリースの定期的なセキュリティレビュー
- ベースラインセキュリティ機能
- セキュリティが強化されたデフォルトモジュール (`/modules/demo-only/` フォルダ内のモジュールは除外されます。これらのモジュールは PoC (概念実証) を目的としており、セキュリティ強化は行われていません)
- セキュリティが強化されたデプロイとオーケストレーションのフレームワーク

これらのセキュリティ責任は、GitHub リポジトリで提供されるフレームワークにのみ適用され、変更やカスタマイズは行われません。これには、`modules/demo-only/` フォルダの ADDF モジュールを除くすべての ADDF モジュールが含まれます。このフォルダーの ADDF モジュールはセキュリティが強化されていないため、本番環境や、機密データや保護データが保存されている環境にはデプロイしないでください。これらのモジュールはシステム機能を紹介するためのものであり、独自にカスタマイズしたセキュリティ強化モジュールを作成するためのベースとして使用することができます。

> **Note**
>
> フレームワークとしての ADDF はそのまま提供されます。「Apache License 2.0」(GitHub) に記載されているとおり、責任や保証は付属していません。ADDF のセキュリティ評価を独自に行い、それが組織の特定のセキュリティ要件に準拠していることを確認する必要があります。

ADDF ユーザーの責任

ADDF とそのモジュールのセキュリティが保護されるのは、ADDF が安全な方法で設定、カスタマイズ、運用されている場合のみです。ADDF ユーザーは、以下のセキュリティについて全責任を負います。

- AWS アカウント アカウント全般の責任:
  - セキュリティコントロールとチェック (指令、検知、予防、対応)

  - マルチアカウントアーキテクチャ
  - ネットワーク設計
  - Identity and Access Management
- ADDF 固有の責任:
  - ADDF のセットアップ
  - ADDF のカスタマイズ
  - ADDF モジュールの開発
  - ADDF の運用
  - ADDF の更新

AWS アカウント アカウント全般の責任

ADDF 関連のリソースを AWS アカウント にデプロイする前に、「AWS Well-Architected Framework」のベストプラクティスに従って AWS アカウント を設定する必要があります。これには、指令、探知、予防、対応型のセキュリティコントロールが含まれます。セキュリティ違反やインシデントが発生した場合に備えて、詳細な緩和プロセスを整備しておく必要があります。組織のポリシーには、アイデンティティ、アクセス、ネットワークを一元管理するための要件を含める必要があります。通常、これらの要件とサービスはランディングゾーンの専門チームが処理します。

ADDF 固有の責任

ADDF の安全なセットアップ

ADDF ユーザーの責任は、ADDF のドキュメントに従って ADDF を安全にセットアップすることから始まります。「ADDF Deployment Guide」(GitHub) の指示に従うことを強くお勧めします。ADDF のセットアップの詳細については、「ADDF アーキテクチャを定義する」と「初期セットアップ」を参照してください。

ADDF の安全なカスタマイズ

CodeSeeder、SeedFarmer、ADDF コアモジュールなどの ADDF コア機能をカスタマイズする場合、ADDF ユーザーはその変更に対する全責任を負います。詳細については、「ADDF デプロイフレームワークコードをカスタマイズする」を参照してください。

ADDF の安全なモジュール開発

ADDF ユーザーは、ADDF を使用してデプロイされるすべてのカスタムモジュールに対する全責任を負います。さらに、ADDF ユーザーは ADDF が提供するモジュールへのコード変更についての責任も負います。詳細については、「ADDF でカスタムモジュールを作成する」を参照してください。

ADDF の安全な更新と運用

フレームワークの進化に伴い、ADDF は機能とセキュリティの更新を受け取ります。ADDF ユーザーは、GitHub リポジトリに公開されている更新を定期的にチェックし、ADDF を長期にわたって安全に運用する責任を負います。詳細については、「ADDF の反復的なデプロイ」、「反復的なセキュリティ監査」、「ADDF の更新」、および「廃止作業」を参照してください。

ADDF のセキュリティ検査プロセス

Autonomous Driving Data Framework (ADDF) は、セキュリティを念頭において構築されました。AWS は、ADDF を一般公開する前に社内で初回のセキュリティ検査を実施し、特定されたセキュリティ上の問題を解決しました。このフレームワークの継続的なセキュリティ検査には、AWS とオープンソースコミュニティの両方が貢献しています。

AWS による定期的なセキュリティ検査

ADDF は、AWS が所有する awslabs という GitHub 組織の下で公開されています。AWS は、この組織のコードに定期的に自動および手動でセキュリティ検査を実施し、最善を尽くしてセキュリティを検査しています。AWS ポリシーに従って、AWS ではセキュリティ検査の頻度、方法、使用するツールに関する情報を開示していません。また、AWS は ADDF に関する内部の監査レポートを公開していません。ただし、セキュリティ上の問題が発見されれば至急修正し、プルリクエストを通じて公開しています。

> **Note**
>
> フレームワークとしての ADDFは、明示か黙示かを問わず、Apache License 2.0 (GitHub) に記載されている保証または所有権、非侵害、商品性、もしくは特定目的への適合性を含む、いかなる種類の保証または条件もなしに「現状のまま」で提供されます。お客様は、ADDF のセキュリティ評価を自ら実施し、自社組織に特有のセキュリティ要件に準拠しているかを確認する必要があります。また、Apache License 2.0 に規定されているとおり、お客様は ADDF の使用または再配布の適切性を判断する責任を単独で負い、かかるライセンスの下でのお客様による履行または許可に関連するリスクを負うものとします。

オープンソースのセキュリティ検査とコントリビューション

ADDF は、コントリビューションを広く受け入れているオープンソースのプロジェクトです。ユーザーは、フレームワークのセキュリティ検査を独自に実施し、セキュリティ関連の検査結果を報告してこれに貢献することが推奨されています。コードに何か問題があった場合は、「Security issue notifications」(ADDF ドキュメント) に記載されているガイドラインに従ってください。

ADDF にあらかじめ組み込まれたセキュリティ機能

Autonomous Driving Data Framework (ADDF) には、さまざまなセキュリティ機能が組み込まれています。通常、これらの機能は、安全なフレームワークを構築して、組織が企業向けの一般的なセキュリティ要件を満たせるように設計されています。

以下が、あらかじめ組み込まれているセキュリティ機能です。

- ADDF モジュールコードの最小特権
- Infrastructure as Code
- IaC の自動セキュリティチェック
- AWS CDK デプロイロールのカスタムの最小特権ポリシー
- モジュールの deployspec ファイルの最小特権ポリシー
- データ暗号化
- Secrets Manager を使用した認証情報ストレージ
- SeedFarmer と CodeSeeder のセキュリティ検査
- CodeSeeder の AWS CodeBuild ロールのための、アクセス許可の境界サポート
- AWS マルチアカウントアーキテクチャ
- マルチアカウントデプロイの最小特権のアクセス許可

ADDF モジュールコードの最小特権

最小特権は、タスクを実行するために最低限必要な権限を付与する際の、セキュリティのベストプラクティスです。詳細については、「最小特権アクセス許可を適用する」を参照してください。ADDF が提供するモジュールは、コードとデプロイ済みのリソースにおける最小特権の原則に、以下のとおり厳密に従います。

- ADDF モジュール用に生成されたすべての AWS Identity and Access Management (IAM) ポリシーには、ユースケースに対する必要最小限のアクセス許可のみ付与されます。
- AWS のサービス は、最小特権の原則に従って設定されデプロイされます。ADDF が提供するモジュールは、特定のユースケースに必要なサービスとサービスの機能のみを使用します。

Infrastructure as Code

ADDF は、フレームワークとして、ADDF モジュールを Infrastructure as Code (IaC) としてデプロイするように設計されています。IaC は、手動のデプロイプロセスをなくし、手動のプロセスで起きやすいエラーや設定のミスを防止します。

ADDF は、一般的な IaC フレームワークを使ってモジュールをオーケストレーションし、デプロイするように設計されています。これには以下が含まれますが、これらに限定されません。

- AWS Cloud Development Kit (AWS CDK)
- AWS CloudFormation
- Hashicorp Terraform

異なる IaC フレームワークを使用すると異なるモジュールを作成できます。次に、ADDF を使ってそれらをデプロイします。

ADDF モジュールで使用されるデフォルトの IaC フレームワークは AWS CDK です。AWS CDK は、AWS リソースを強制的に定義するときに使用できる、ハイレベルなオブジェクト指向の抽象化です。AWS CDK ではすでに、さまざまなサービスやシナリオに対してデフォルトでセキュリティのベストプラクティスが適用されています。AWS CDK を使用することで、セキュリティの設定ミスによるリスクを防ぎます。

IaC の自動セキュリティチェック

ADDF には、オープンソースの cdk-nag ユーティリティ (GitHub) が組み込まれています。このユーティリティは、AWS CDK に基づく ADDF モジュールが一般的なベストプラクティスとセキュリティ上のベストプラクティスを順守しているかどうかを自動でチェックします。cdk-nag ユーティリティは、ルールとルールパックを使用して、ベストプラクティスに違反しているコードを検出し、報告します。これらのルールと包括的なリストの詳細については、cdk-nag rules (GitHub) を参照してください。

AWS CDK デプロイロールのカスタムの最小特権ポリシー

ADDF は、AWS CDK v2 を幅広く活用します。ユーザーは、すべての ADDF AWS アカウント を AWS CDK にブートストラップする必要があります。詳細については「Bootstrapping」(AWS CDK ドキュメント) を参照してください。

デフォルトでは、AWS CDK は、寛容な AdministratorAccess AWS マネージドポリシーを、ブートストラップしたアカウントで作成された AWS CDK デプロイロールに割り当てます。このロールの完全な名前は `cdk-[CDK_QUALIFIER]-cfn-exec-role-[AWS_ACCOUNT_ID]-[REGION]` です。AWS CDK は、AWS CDK デプロイプロセスの一環として、このロールを使ってブートストラップした AWS アカウント にリソースをデプロイします。

組織のセキュリティ要件によっては、`AdministratorAccess` ポリシーの許容性が過剰になる場合があります。ポリシーとアクセス許可は、AWS CDK ブートストラッププロセスの最中に必要に応じてカスタマイズできます。このポリシーは、`--cloudformation-execution-policies` パラメータを使って、ポリシーが新たに定義されたアカウントを再度ブートストラップすることで変更できます。詳細については「Customizing bootstrapping」(AWS CDK ドキュメント) を参照してください。

> **Note**
>
> このセキュリティ機能は ADDF に固有の機能ではありませんが、ADDF のデプロイの全体的なセキュリティを高めることにつながるため、こちらのセクションに記載されています。

モジュールの deployspec ファイルの最小特権ポリシー

各モジュールには、deployspec.yaml というデプロイ仕様のファイルが含まれています。このファイルは、モジュールのデプロイ手順を定義します。CodeSeeder は、AWS CodeBuild を使用して定義済みのモジュールをターゲットアカウントにデプロイする際に、こちらを使用します。CodeSeeder は、デプロイ仕様ファイルの指示に従ってデフォルトのサービスロールを CodeBuild に割り当てて、リソースをデプロイします。このサービスロールは最小特権の原則に従って設計されています。このロールには AWS CDK アプリケーションをデプロイする際に必要になるすべてのアクセス許可が含まれています。ADDF が提供するモジュールは、すべて AWS CDK アプリケーションと同様に作成されるためです。

ただし、AWS CDK の外部でステージコマンドを実行する必要がある場合は、CodeBuild のデフォルトのサービスロールを使用するのではなく、カスタム IAM ポリシーを作成する必要があります。例えば、Terraform など、AWS CDK 以外の IaC デプロイフレームワークを使用している場合、そのフレームワークが機能するには、そのための十分なアクセス許可を付与する IAM ポリシーを作成することが必要になります。専用の IAM ポリシーが必要になるもう 1 つのシナリオが、`install`、`pre_build`、`build`、`post_build` のいずれかのステージコマンドに、他の AWS のサービス サービスへのダイレクトな AWS Command Line Interface (AWS CLI) 呼び出しを含める場合です。例えば、モジュールに、ファイルを S3 バケットにアップロードするための Amazon

Simple Storage Service (Amazon S3) コマンドが含まれる場合、カスタムポリシーが必要です。カスタム IAM ポリシーを使用すると、AWS CDK デプロイ以外の AWS コマンドを詳細にコントロールできます。カスタム IAM ポリシーの例については、「ModuleStack」(SeedFarmer ドキュメント) を参照してください。ADDF モジュール用にカスタム IAM ポリシーを作成するときは、必ず最小特権のアクセス許可を適用します。

データ暗号化

ADDF は、機密である可能性の高いデータを保存して処理します。こうしたデータを保護するため、SeedFarmer、CodeSeeder、ADDF が提供するモジュールは (`demo-only` フォルダのモジュールに関しては別段の明記がない限り)、使用したすべての AWS のサービス の、保管中のデータと転送中のデータを暗号化します。

Secrets Manager を使用した認証情報ストレージ

ADDFは、Docker Hub、JupyterHub、Amazon Redshift など、さまざまなサービスのさまざまなシークレットを処理します。ADDF 関連のシークレットを保管するときは AWS Secrets Manager を使用します。これによりユーザーは、機密データをソースコードから削除することができます。

Secrets Manager のシークレットは、ターゲットアカウントが正常に機能するために必要になるものであるため、ターゲットアカウントにのみ保管されます。ツールチェーンアカウントには、通常はいずれのシークレットも含まれません。

SeedFarmer と CodeSeeder のセキュリティ検査

SeedFarmer と CodeSeeder (GitHub リポジトリ) は、ADDF とその ADDF モジュールをデプロイするときに使用します。これらのオープンソースのプロジェクトは、ADDF のセキュリティ検査プロセス にあるとおり、ADDF と同じ、AWS の内部セキュリティ検査を定期的に受けています。

CodeSeeder の AWS CodeBuild ロールのための、アクセス許可の境界サポート

IAM のアクセス許可の境界は、ID 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定義する、一般的なセキュリティメカニズムです。SeedFarmer と CodeSeeder は、各ターゲットアカウントの、IAM アクセス許可の境界のアタッチメントをサポートしています。アクセス許可の境界は、CodeSeeder がモジュールをデプロイする際に CodeBuild が使用する、サービ

スロールのアクセス許可の上限数を設定します。IAM のアクセス許可の境界は、セキュリティチームが ADDF の外部で作成する必要があります。IAM のアクセス許可の境界におけるポリシーのアタッチメントは、ADDF デプロイのマニフェストファイルである deployment.yaml で属性として受け取られます。詳細については、「Permissions boundary support」(SeedFarmer ドキュメント) を参照してください。

ワークフローは次のとおりです。

1. セキュリティチームが自社のセキュリティ要件に従って IAM のアクセス許可の境界を定義し、作成します。IAM のアクセス許可の境界は、各 ADDF AWS アカウント で個別に作成する必要があります。。出力は、アクセス許可の境界ポリシーの Amazon リソースネーム (ARN) リストです。
2. セキュリティチームが、ポリシー ARN リストを 自社の ADDF デベロッパーチームと共有します。
3. ADDF デベロッパーチームが、ポリシー ARN リストをマニフェストファイルに統合します。統合の例については、sample-permissionboundary.yaml (GitHub) と 「Deployment manifest」(SeedFarmer ドキュメント) を参照してください。
4. デプロイが完了すると、アクセス許可の境界が、CodeBuild がモジュールをデプロイするときに使用するすべてのサービスロールにアタッチされます。
5. セキュリティチームは、必要に応じて、アクセス許可の境界が適用されていることをモニタリングします。

AWS マルチアカウントアーキテクチャ

AWS Well-Architected フレームワークのセキュリティの柱で定義されているとおり、リソースとワークロードは、組織の要件に基づいて複数の AWS アカウント に分割することがベストプラクティスとされています。これは、AWS アカウント が分離境界として機能するためです。詳細については、「AWS アカウント の管理と分離」を参照してください。この概念を実現したのが、マルチアカウントアーキテクチャと呼ばれるものです。単一アカウントアーキテクチャと比較すると、適切に設計された AWS マルチアカウントアーキテクチャではワークロードを分類できるため、セキュリティ侵害が発生した場合の影響範囲が小さくなります。

ADDF は、AWS マルチアカウントアーキテクチャをネイティブにサポートしています。お使いの ADDF モジュールは、組織のセキュリティと職務分掌の要件に応じて、必要な数の AWS アカウントに分散することができます。ADDF は、1 つの AWS アカウント にデプロイし、ツールチェーンとターゲットアカウントの機能を組み合わせることが可能です。あるいは、ADDF モジュールまたはモジュールグループ用に、個別のターゲットアカウントを作成することもできます。

留意すべき唯一の制限は、ADDF モジュールは各 AWS アカウント のデプロイにおいて最小の単位になるという点です。

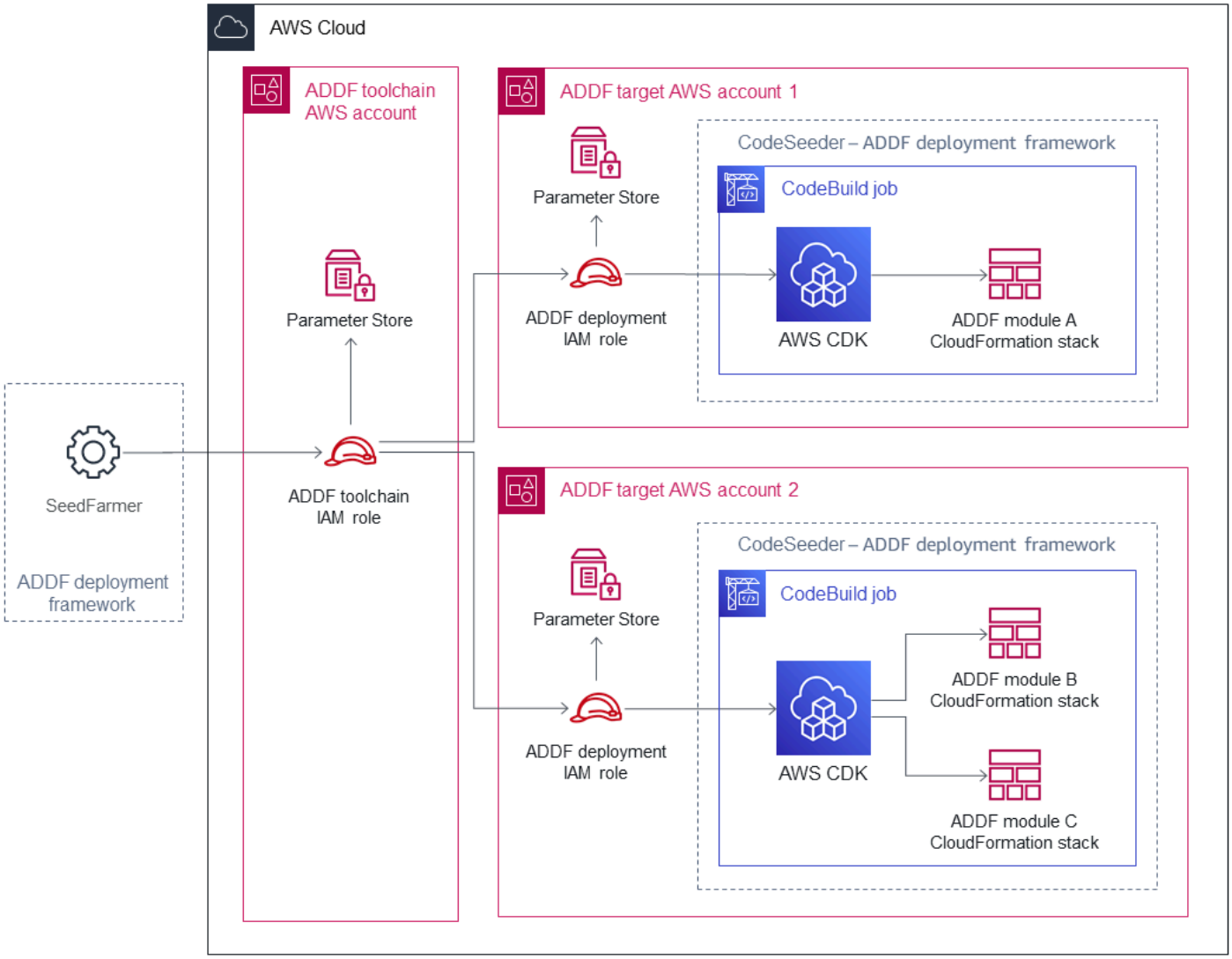
本番環境では、ツールチェーンアカウントと、1 つ以上のターゲットアカウントから構成されるマルチアカウントアーキテクチャを使用することが推奨されます。詳細については、「ADDF アーキテクチャ」を参照してください。

マルチアカウントデプロイの最小特権のアクセス許可

マルチアカウントアーキテクチャを使用する場合、SeedFarmer では、ターゲットアカウントにアクセスし、次の 3 つのアクションを実行する必要があります。

1. ADDF モジュールのメタデータをツールチェーンアカウントとターゲットアカウントに書き込む。
2. 使用中の ADDF モジュールのメタデータをツールチェーンアカウントとターゲットアカウントから読み取る。
3. モジュールをデプロイまたは更新するため、ターゲットアカウントで AWS CodeBuild ジョブを実行する。

以下は、ADDF に固有の AWS Identity and Access Management (IAM) ロールを想定したオペレーションを含む、クロスアカウントの関係を示した図です。



このようなクロスアカウントアクションは、明確に定義された assume-role オペレーションを使用することで実現できます。

- ADDF ツールチェーン IAM ロールは、ツールチェーンアカウントにデプロイされます。このロールは SeedFarmer が引き受けます。このロールは、`iam:AssumeRole` を実行するアクセス許可を持っており、ADDF デプロイ IAM ロールを各ターゲットアカウントで引き受けることができます。さらに、ADDF ツールチェーンの IAM ロールは、ローカルの AWS Systems Manager パラメータストアオペレーションで実行できます。
- ADDF デプロイ IAM ロールは、各ターゲットアカウントにデプロイされます。このロールは、ADDF ツールチェーン IAM ロールを使用して、ツールチェーンアカウントからのみ引き受けることができます。このロールは、ローカルの AWS Systems Manager パラメータストアオペ

レーションと、CodeSeeder を使用して CodeBuild ジョブを開始および記述する AWS CodeBuild アクションを実行するためのアクセス許可を持っています。

これら ADDF に固有の IAM ロールは、ADDF ブートストラッピングプロセスの一環で作成されます。詳細については、「ADDF Deployment Guide」(GitHub) の「Bootstrap AWS アカウント」を参照してください。

クロスアカウントのアクセス許可はすべて、最小特権の原則に従って設定されます。一方のターゲットアカウントが侵害されても、もう一方の ADDF AWS アカウント は影響が最小限か、まったく受けずに済みます。

ADDF のシングルアカウントアーキテクチャの場合でも、ロールの関係は変わりません。分解されて 1 つの AWS アカウント になるだけです。

ADDF の安全なセットアップと運用

Autonomous Driving Data Framework (ADDF) は、組織内の専任の DevOps チームとセキュリティチームによる継続的なメンテナンスとケアが必要となるカスタムソフトウェアとして扱う必要があります。このセクションでは、ADDF のライフサイクル全体のセットアップと運用に役立つ、セキュリティ関連の一般的なタスクについて説明します。

このセクションには、以下のタスクが含まれます。

- ADDF アーキテクチャを定義する
- 初期セットアップ
- ADDF デプロイフレームワークコードをカスタマイズする
- ADDF でカスタムモジュールを作成する
- ADDF の反復的なデプロイ
- 反復的なセキュリティ監査
- ADDF の更新
- 廃止作業

ADDF アーキテクチャを定義する

ADDF インスタンスのセキュリティは、それがデプロイされている AWS アカウント 環境と同程度にしか保護しません。この AWS アカウント 環境は、特定のユースケースのセキュリティと運用上のニーズを満たすように設計する必要があります。例えば、PoC (概念実証) 環境で ADDF インスタンスを設定する場合のセキュリティや、運用関連のタスクと考慮事項は、本番稼働用の ADDF 環境を設定する場合とは異なります。

PoC (概念実証) 環境で ADDF を実行する

ADDF を PoC (概念実証) 環境で使用する場合は、他のワークロードが含まれない ADDF 専用の AWS アカウント を作成することをお勧めします。これにより、ADDF とその機能を利用する際に、アカウントを安全に保つことができます。この方法による利点は以下の通りです。

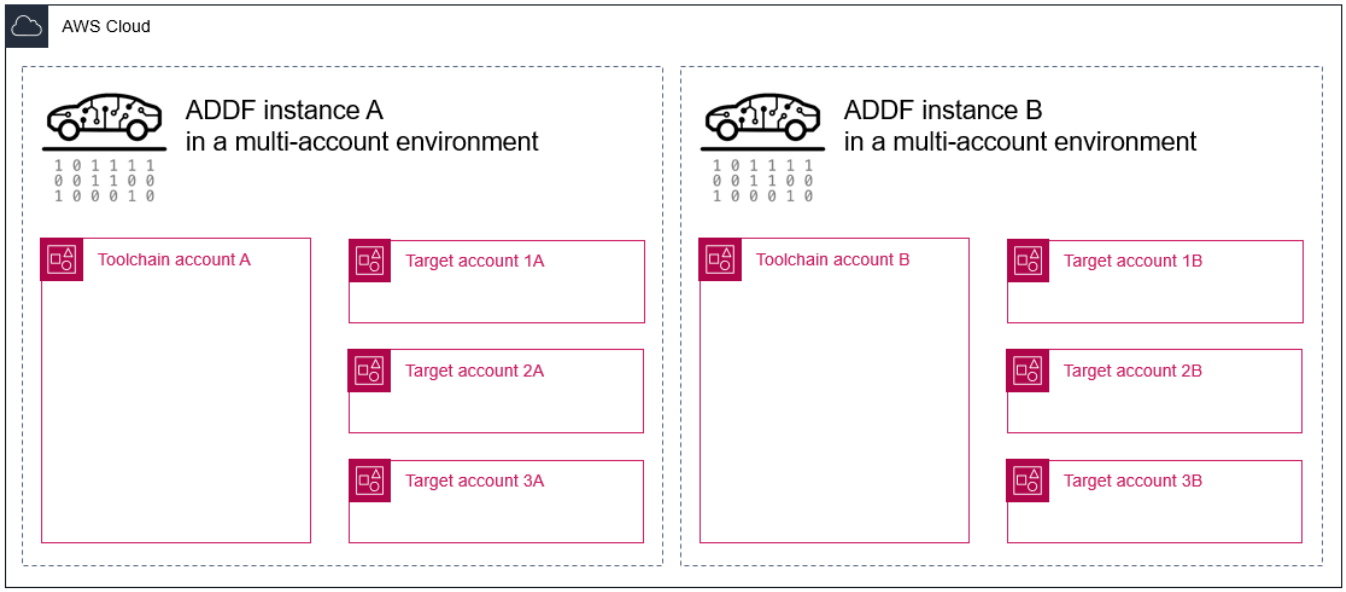
- ADDF の設定に重大な誤りがあっても、他のワークロードに悪影響が及ぶことがない。
- その他のワークロードの設定ミスによって ADDF の設定に悪影響が及ぶリスクがない。

PoC (概念実証) 環境の場合でも、可能な限り、本番環境で ADDF を実行する に記載されているベストプラクティスに従うことをお勧めします。

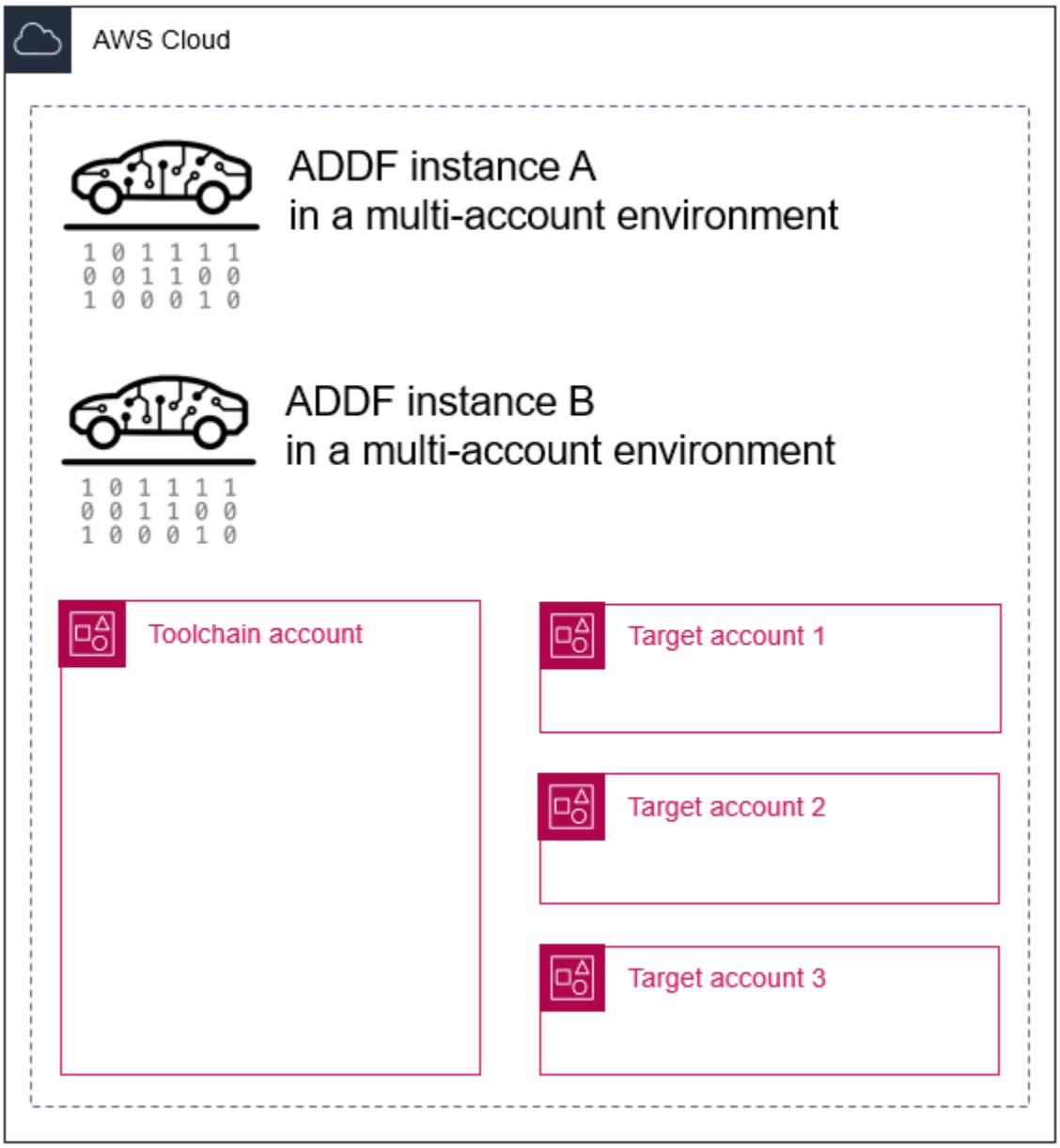
本番環境で ADDF を実行する

ADDF を企業の本番環境で使用する場合は、組織のセキュリティに関するベストプラクティスを検討し、それに応じて ADDF を実装することを強くお勧めします。組織のセキュリティに関するベストプラクティスに加えて、以下を実装することをお勧めします。

- 長期的かつ献身的な ADDF DevOpsチームを結成する — ADDF はカスタムソフトウェアとして扱う必要があります。専任の DevOps チームによる継続的なメンテナンスとケアが必要です。本番環境で ADDF の実行を開始する前に、ADDF デプロイの耐用年数が終了するまでの間、十分な規模と能力を備えた DevOps チームを定義し、十分なリソースを投入する必要があります。
- マルチアカウントアーキテクチャを使用する – 各 ADDF インスタンスは、他の無関係なワークロードが含まれていない、専用の AWS マルチアカウント環境にデプロイする必要があります。「AWS アカウントの管理と分離」(AWS Well-Architected フレームワーク) で定義されているように、組織の要件に基づいてリソースとワークロードを複数の AWS アカウント に分離することがベストプラクティスです。これは、AWS アカウント が分離境界として機能するためです。単一アカウントアーキテクチャと比較すると、適切に設計された AWS マルチアカウントアーキテクチャではワークロードを分類できるため、セキュリティ侵害が発生した場合の影響範囲が小さくなります。マルチアカウントアーキテクチャを使用すると、アカウントを AWS のサービス クォータの範囲内に維持することができます。組織のセキュリティと職務分離の要件を満たすために、ADDF モジュールを必要な数の AWS アカウント に分散します。
- 複数の ADDF インスタンスをデプロイする — 組織のソフトウェア開発プロセスに従って ADDF モジュールを適切に開発、テスト、デプロイするために、必要な数だけ個別の ADDF インスタンスをセットアップします。複数の ADDF インスタンスを設定する場合は、次のいずれかの方法を使用できます。
  - 異なる AWS マルチアカウント環境における複数の ADDF インスタンス — 個別の AWS アカウント を使用して異なる ADDF インスタンスを分離することができます。例えば、組織に専用の開発、テスト、本番稼働のステージがある場合、ステージごとに個別の ADDF インスタンスと専用アカウントを作成できます。これにより、エラーがステージ全体に広がるリスクを軽減したり、承認プロセスを実装しやすくしたり、ユーザーアクセスを特定の環境のみに制限したりするなど、多くの利点が得られます。次のイメージは、別々のマルチアカウント環境にデプロイされた 2 つの ADDF インスタンスを示しています。



- 同じ AWS マルチアカウント環境における複数の ADDF インスタンス — 同じ AWS マルチアカウント環境を共有する複数の ADDF インスタンスを作成できます。これにより、同じ AWS アカウント 内に独立したブランチが効果的に作成されます。例えば、異なるデベロッパーが並行して作業している場合、デベロッパーは同じ AWS アカウント に専用の ADDF インスタンスを作成することができます。これにより、デベロッパーは独立したブランチで、開発やテストの作業を行うことができます。この方法を使用する場合、ADDF インスタンスごとに ADDF リソースに一意のリソース名を付ける必要があります。これは ADDF が事前に提供しているモジュールでデフォルトでサポートされています。この方法は、AWS のサービス クォータの範囲内で使用できます。以下のイメージは、共有のマルチアカウント環境にデプロイされた 2 つの ADDF インスタンスを示しています。



- 同じ AWS 単一アカウント環境における複数の ADDF インスタンス — このアーキテクチャは前の例と非常に似ています。違いは、複数の ADDF インスタンスがマルチアカウント環境ではなく単一アカウント環境にデプロイされる点です。このアーキテクチャは、非常に限られた範囲内で、複数のデベロッパーが異なるブランチで同時に作業するごく単純な ADDF のユースケースに適しています。

SeedFarmer は ADDF インスタンスのデプロイを制御する単一のツールであるため、組織のデプロイ戦略と CI/CD プロセスに適合する環境とアカウントアーキテクチャを構築できます。

- 組織のセキュリティ要件に応じて、AWS Cloud Development Kit (AWS CDK) ブートストラッププロセスをカスタマイズする — デフォルトでは、ブートストラッププロセス中に AWS CDK が AdministratorAccess AWS 管理ポリシーを割り当てます。このポリシーは完全な管理権限を付与します。このポリシーが組織のセキュリティ要件に対して寛容すぎる場合は、適用するポリシーをカスタマイズすることができます。詳細については、「AWS CDK デプロイロールのカスタムの最小特権ポリシー」を参照してください。
- IAM でアクセスを設定する際のベストプラクティスを遵守する — 構造化された AWS Identity and Access Management (IAM) アクセスソリューションを確立して、ユーザーが ADDF AWS アカウント にアクセスできるようにします。ADDF のフレームワークは、範囲を最小特権の原則に従うように設計されています。IAM のアクセスパターンも最小特権の原則に従い、組織の要件に準拠し、「IAM でのセキュリティのベストプラクティス」(IAM ドキュメント) を遵守する必要があります。

- 組織のベストプラクティスに従ってネットワークを設定する — ADDF には、基本的なパブリックまたはプライベート仮想プライベートクラウド (VPC) を作成するネットワーク AWS CloudFormation スタックがオプションで含まれています。組織の設定によっては、この VPC によってリソースがインターネットに直接公開がされる場合があります。組織のネットワークのベストプラクティスに従い、セキュリティが強化されたカスタムネットワークモジュールを作成することをお勧めします。
- セキュリティ防止、検知、緩和に関する対策を AWS アカウント レベルでデプロイする — AWS は、Amazon GuardDuty、AWS Security Hub、Amazon Detective、AWS Config などのさまざまなセキュリティサービスを提供しています。これらのサービスを ADDF AWS アカウント で有効にし、組織のセキュリティ防止、検出、緩和、インシデント処理の各プロセスを統合します。「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 アーキテクチャセンター)、とそのサービスのドキュメントに含まれているサービス固有の推奨事項に従うことをお勧めします。詳細については、「AWS セキュリティドキュメント」を参照してください。

実装と設定の詳細は、組織固有の要件とプロセスに大きく依存するため、ADDF はこれらのトピックを取り上げていません。むしろ、これらのトピックに取り組むことは、組織の中核的責任です。一般的に、AWS ランディングゾーンを管理するチームが ADDF 環境の計画と実装を支援します。

初期セットアップ

「ADDF Deployment Guide」(GitHub) に従って ADDF を設定します。デプロイの開始点は、autonomous-driving-data-framework Git Hub リポジトリ内の `/manifest` フォルダです。`/manifest/example-dev` フォルダにはデモ用のサンプルデプロイが含まれています。このサンプルを独自のデプロイを設計するための開始点として使用してください。そのディレクトリには、deployment.yaml という名前の ADDF デプロイマニフェストファイルがあります。このファイルには、SeedFarmer が AWS クラウド 内で ADDF とそのリソースを管理、デプロイ、または削除するためのすべての情報が含まれています。ADDF モジュールのグループを専用ファイルに作成することができます。core-modules.yaml はコアモジュールグループの一例で、ADDF が提供するすべてのコアモジュールが含まれています。つまり、deployment.yaml ファイルには、ターゲットアカウントにデプロイされるグループとモジュールへのすべての参照が含まれ、デプロイの順序も指定されています。

安全かつコンプライアンスに準拠した設定を行うために、特に PoC (概念実証) を目的としていない環境では、デプロイする各モジュールのソースコードを確認することをお勧めします。セキュリティ強化のベストプラクティスに従って、目的のユースケースに必要なモジュールだけをデプロイするようにしてください。

> **Note**
>
> `modules/demo-only/` フォルダ内の ADDF モジュールはセキュリティが強化されていないため、本番環境や機密データや保護されたデータが保存されている環境にはデプロイしないでください。これらのモジュールはシステム機能を紹介するためのものであり、独自にカスタマイズしたセキュリティ強化モジュールを作成するためのベースとして使用することができます。

ADDF デプロイフレームワークコードをカスタマイズする

ADDF デプロイフレームワークとそのオーケストレーション、およびデプロイロジックは、あらゆる要件に合わせて完全にカスタマイズできます。ただし、以下の理由により、カスタマイズを控えるか、変更を最小限に抑えることをお勧めします。

- アップストリーム互換性を維持 — アップストリーム互換性により、最新機能やセキュリティ更新に合わせた ADDF の更新を簡単に行うことができます。フレームワークを変更すると、SeedFarmer、CodeSeeder、すべての ADDF コアモジュールとのネイティブ下位互換性が失われます。
- セキュリティ上の影響 — ADDF デプロイフレームワークの変更は複雑な作業であり、意図しないセキュリティ上の影響をもたらす可能性があります。最悪のシナリオでは、フレームワークの変更によってセキュリティ上の脆弱性が生じる可能性があります。

可能であれば、ADDF デプロイフレームワークや ADDF コアモジュールコードを変更するのではなく、独自のモジュールコードを構築してカスタマイズしてください。

> **Note**
>
> ADDF デプロイフレームワークの特定の部分を改善したり、セキュリティをさらに強化したりする必要があると思われる場合は、プルリクエストを介して変更を ADDF リポジトリに提供してください。詳細については、「オープンソースのセキュリティ検査とコントリビューション」を参照してください。

ADDF でカスタムモジュールを作成する

ADDF のコアコンセプトは、新しい ADDF モジュールの作成または既存のモジュールの拡張です。モジュールを作成またはカスタマイズする場合は、AWS のセキュリティに関する一般的なベストプラクティスと、セキュリティ保護されたコーディングに関する組織のベストプラクティスに従うことをお勧めします。また、セキュリティ問題のリスクをさらに軽減するために、組織のセキュリティ要件に基づいて、初期および定期的に内部または外部の技術的なセキュリティレビューを実施することをお勧めします。

ADDF の反復的なデプロイ

「ADDF Deployment Guide」(GitHub) の説明に従って、ADDF とそのモジュールをデプロイします。ターゲットアカウントのリソースを追加、更新、削除する ADDF の反復的なデプロイをサポートするために、SeedFarmer はツールチェーンとターゲットアカウントのパラメータストアに保存されている MD5 ハッシュを使用して、現在デプロイされているインフラストラクチャをローカルコードベースのマニフェストファイルで定義されているインフラストラクチャと比較します。

この方法は、ソースリポジトリ (SeedFarmer を操作するローカルコードベース) が信頼できる情報源であり、その中で明示的に宣言されたインフラストラクチャがデプロイの望ましい結果であるという GitOps パラダイムに従っています。GitOps の詳細については、「What is GitOps」(GitLab ウェブサイト) を参照してください。

反復的なセキュリティ監査

組織内の他のソフトウェアと同様に、ADDF とカスタム ADDF モジュールコードをセキュリティリスク管理、セキュリティレビュー、セキュリティ監査サイクルに統合します。

ADDF の更新

ADDF は、継続的な開発作業の一環として定期的な更新を受け取ります。更新には、機能の更新、セキュリティ関連の改善と修正が含まれます。新しいフレームワークのリリースを定期的に確認し、更新を適切なタイミングで適用することをお勧めします。詳細については、「Steps to update ADDF」(ADDF ドキュメント) を参照してください。

廃止作業

ADDF が不要になったら、ADDF とそれに関連するすべてのリソースを AWS アカウント から削除します。インフラストラクチャが無人で使用されていない場合、不要なコストが発生し、セキュリ

ティリスクが発生する可能性があります。詳細については、「Steps to destroy ADDF」(ADDF ドキュメント) を参照してください。

次のステップ

本ガイドでは、AWS クラウド 環境で Autonomous Driving Data Framework (ADDF) をデプロイする際の、セキュリティとオペレーションに関するベストプラクティスおよび留意事項について説明してきました。ここからは、ADDF を安全にセットアップし運用する際のお客様の役割と責任を理解していただくため、ADDF ユーザー、ADDF コアチーム、AWS 間の責任共有モデルについて確認します。また、環境に固有の推奨事項など、ADDF をそのライフサイクル全般を通じて安全に運用していただくための推奨事項についても説明します。

リソース のセクションでご紹介しているリソースを事前にお読みいただくことをお勧めします。準備が整ったら、「ADDF デプロイガイド」(GitHub) にある手順に従って ADDF をセットアップしていきましょう。

ADDF のセットアップ中や運用中に、デプロイのフレームワークに改善やセキュリティ強化が必要だと思われたときは、プルリクエストを通じて変更事項を ADDF リポジトリに投稿してください。詳細については、「オープンソースのセキュリティ検査とコントリビューション」を参照してください。

リソース

AWSドキュメント

- AWS 上で Autonomous Driving Data Framework (ADDF) を使用した、カスタマイズされたワークフローの開発とデプロイ(AWS ブログ記事)
- AWS セキュリティサービスのドキュメント
- IAM でのセキュリティのベストプラクティス
- AWS アカウントの管理と分離
- AWS CDK のブートストラッピング
- AWS 責任共有モデル
- AWS Well-Architected フレームワーク

オープンソースのリソース

- ADDF repository (GitHub)
- ADDF Deployment Guide (GitHub)
- CodeSeeder repository (GitHub)
- SeedFarmer repository (GitHub)

注意

お客様は、本書に記載されている情報を独自に評価する責任を負うものとします。本書は、(a) 情報提供のみを目的とし、(b) AWS の現行製品と慣行について説明しており、これらは予告なしに変更されることがあり、(c) AWS およびその関連会社、サプライヤー、またはライセンサーからの契約上の義務や保証をもたらすものではありません。AWS の製品やサービスは、明示または黙示を問わず、一切の保証、表明、条件なしに「現状のまま」提供されます。

お客様に対する AWS の責任は AWS 契約によって規定されます。本書は、AWS とお客様との間で締結されるいかなる契約の一部でもなく、その内容を修正するものでもありません。

ドキュメント履歴

このガイドは、このドキュメントの大きな変更点をまとめたものです。今後の更新に関する通知を受け取る場合は、RSS フィードをサブスクライブできます。

変更	説明	日付
初版発行	—	2022 年 11 月 15 日

AWS 規範的ガイダンスの用語集

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

数字

7 Rs

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

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

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

A

ABAC

「属性ベースのアクセスコントロール」を参照してください。

抽象化されたサービス

「 マネージドサービス」を参照してください。

ACID

「原子性、一貫性、分離性、耐久性」を参照してください。

アクティブ - アクティブ移行

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

アクティブ - パッシブ移行

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

集計関数

行のグループを操作し、グループの単一の戻り値を計算する SQL 関数。集計関数の例としては、 SUMや などがありますMAX。

AI

「人工知能」を参照してください。

AIOps

「人工知能オペレーション」を参照してください。

匿名化

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

アンチパターン

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

アプリケーションコントロール

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

アプリケーションポートフォリオ

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

人工知能 (AI)

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

AI オペレーション (AIOps)

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

非対称暗号化

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

原子性、一貫性、分離性、耐久性 (ACID)

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

属性ベースのアクセス制御 (ABAC)

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

信頼できるデータソース

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

アベイラビリティーゾーン

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

AWS クラウド導入フレームワーク (AWS CAF)

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

AWS ワークロード認定フレームワーク (AWS WQF)

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

B

不正なボット

個人または組織に混乱や損害を与えることを目的としたボット。

BCP

事業継続計画を参照してください。

動作グラフ

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

ビッグエンディアンシステム

最上位バイトを最初に格納するシステム。エンディアンネス も参照してください。

二項分類

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

ブルームフィルター

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

ブルー/グリーンデプロイ

2 つの異なる同一の環境を作成するデプロイ戦略。現在のアプリケーションバージョンは 1 つの環境 (青) で実行し、新しいアプリケーションバージョンは他の環境 (緑) で実行します。この戦略は、影響を最小限に抑えながら迅速にロールバックするのに役立ちます。

ボット

インターネット経由で自動タスクを実行し、人間のアクティビティやインタラクションをシミュレートするソフトウェアアプリケーション。インターネット上の情報のインデックスを作成するウェブクローラーなど、一部のボットは有用または有益です。悪質なボット と呼ばれる他のボッ

トの中には、個人や組織に混乱を与えたり、損害を与えたりすることを意図しているものがあります。

ボットネット

マルウェアに感染し、ボットのヘルダーまたはボットオペレーター と呼ばれる、単一関係者の管理下にあるボットのネットワーク。ボットは、ボットとその影響をスケールするための最もよく知られているメカニズムです。

ブランチ

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

ブレークグラスアクセス

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

ブラウンフィールド戦略

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

バッファキャッシュ

アクセス頻度が最も高いデータが保存されるメモリ領域。

ビジネス能力

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

ビジネス継続性計画 (BCP)

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

C

CAF

AWS 「クラウド導入フレームワーク」を参照してください。

Canary デプロイ

エンドユーザーへのバージョンの低速かつ増分的なリリース。確信できたら、新しいバージョンをデプロイし、現在のバージョン全体を置き換えます。

CCoE

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

CDC

「データキャプチャの変更」を参照してください。

変更データキャプチャ (CDC)

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

カオスエンジニアリング

障害や破壊的なイベントを意図的に導入して、システムの耐障害性をテストします。AWS Fault Injection Service （ AWS FIS ） を使用して、 AWS ワークロードに負荷をかけてレスポンスを評価する実験を実行できます。

CI/CD

「継続的インテグレーションと継続的デリバリー」を参照してください。

分類

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

クライアント側の暗号化

ターゲットがデータ AWS のサービス を受信する前に、ローカルでデータを暗号化します。

Cloud Center of Excellence (CCoE)

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

クラウドコンピューティング

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

クラウド運用モデル

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

導入のクラウドステージ

組織が に移行するときに通常実行する 4 つのフェーズ AWS クラウド :

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

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

CMDB

「設定管理データベース」を参照してください。

コードリポジトリ

ソースコードやその他の資産 (ドキュメント、サンプル、スクリプトなど) が保存され、バージョン管理プロセスを通じて更新される場所。一般的なクラウドリポジトリには、 GitHub または が含まれます AWS CodeCommit。コードの各バージョンはブランチと呼ばれます。マイクロサー

ビスの構造では、各リポジトリは 1 つの機能専用です。1 つの CI/CD パイプラインで複数のリポジトリを使用できます。

コールドキャッシュ

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

コールドデータ

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

コンピュータビジョン (CV)

機械学習を使用してデジタルイメージやビデオなどのビジュアル形式から情報を分析および抽出する AI の分野。例えば、 はオンプレミスのカメラネットワークに CV を追加するデバイス AWS Panorama を提供し、Amazon SageMaker は CV の画像処理アルゴリズムを提供します。

設定ドリフト

ワークロードの場合、設定は想定した状態から変化します。これにより、ワークロードが非準拠になる可能性があり、通常は段階的かつ意図的ではありません。

構成管理データベース (CMDB)

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

コンフォーマンスパック

コンプライアンスチェックとセキュリティチェックをカスタマイズするためにアセンブルできる AWS Config ルールと修復アクションのコレクション。YAML テンプレートを使用して、コンフォーマンスパックを AWS アカウント および リージョンで単一のエンティティとしてデプロイすることも、組織全体にデプロイすることもできます。詳細については、 AWS Config ドキュメントの「コンフォーマンスパック」を参照してください。

継続的インテグレーションと継続的デリバリー (CI/CD)

ソフトウェアリリースプロセスのソース、ビルド、テスト、ステージング、本番の各ステージを自動化するプロセス。CI/CD は一般的にパイプラインと呼ばれます。プロセスの自動化、生産性

の向上、コード品質の向上、配信の加速化を可能にします。詳細については、「継続的デリバリーの利点」を参照してください。CD は継続的デプロイ (Continuous Deployment) の略語でもあります。詳細については「継続的デリバリーと継続的なデプロイ」を参照してください。

CV

「コンピュータビジョン」を参照してください。

D

保管中のデータ

ストレージ内にあるデータなど、常に自社のネットワーク内にあるデータ。

データ分類

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

データドリフト

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

転送中のデータ

ネットワーク内 (ネットワークリソース間など) を活発に移動するデータ。

データメッシュ

一元化された管理とガバナンスにより、分散型の分散型データ所有権を提供するアーキテクチャフレームワーク。

データ最小化

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

データ境界

AWS 環境内の一連の予防ガードレール。信頼できる ID 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していることを確認できます。詳細については、「 でのデータ境界の構築 AWS」を参照してください。

データの前処理

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

データ出所

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

データ件名

データを収集、処理している個人。

データウェアハウス

分析などのビジネスインテリジェンスをサポートするデータ管理システム。データウェアハウスには通常、大量の履歴データが含まれており、クエリや分析によく使用されます。

データベース定義言語 (DDL)

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

データベース操作言語 (DML)

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

DDL

「データベース定義言語」を参照してください。

ディープアンサンブル

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

ディープラーニング

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

defense-in-depth

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

委任管理者

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

デプロイメント

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

開発環境

「環境」を参照してください。

検出管理

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

開発バリューストリームマッピング (DVSM)

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

デジタルツイン

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

ディメンションテーブル

スタースキーマ では、ファクトテーブル内の量的データに関するデータ属性を含む小さなテーブル。ディメンションテーブル属性は通常、テキストフィールドまたはテキストのように動作する離散数値です。これらの属性は、クエリの制約、フィルタリング、結果セットのラベル付けに一般的に使用されます。

ディザスタ

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

ディザスタリカバリ (DR)

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

DML

「データベース操作言語」を参照してください。

ドメイン駆動型設計

各コンポーネントが提供している変化を続けるドメイン、またはコアビジネス目標にコンポーネントを接続して、複雑なソフトウェアシステムを開発するアプローチ。この概念は、エリック・エヴァンスの著書、Domain-Driven Design: Tackling Complexity in the Heart of Software (ドメイン駆動設計:ソフトウェアの中心における複雑さへの取り組み) で紹介されています (ボストン: Addison-Wesley Professional、2003)。strangler fig パターンでドメイン駆動型設計を使用する方法の詳細については、コンテナと Amazon API Gateway を使用して、従来の Microsoft ASP.NET (ASMX) ウェブサービスを段階的にモダナイズ を参照してください。

DR

「ディザスタリカバリ」を参照してください。

ドリフト検出

ベースライン設定からの偏差の追跡。例えば、 AWS CloudFormation を使用してシステムリソース のドリフトを検出したり、 を使用して AWS Control Tower ガバナンス要件への準拠に影響を与える可能性のあるランディングゾーンの変更を検出したりできます。

DVSM

「開発値ストリームマッピング」を参照してください。

E

EDA

「探索的データ分析」を参照してください。

エッジコンピューティング

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

暗号化

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

暗号化キー

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

エンディアン

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

エンドポイント

「サービスエンドポイント」を参照してください。

エンドポイントサービス

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

エンタープライズリソースプランニング (ERP)

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

エンベロープ暗号化

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

環境

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

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

エピック

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

ERP

「エンタープライズリソース計画」を参照してください。

探索的データ分析 (EDA)

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

F

ファクトテーブル

スタースキーマ の中央テーブル。事業運営に関する定量的データを保存します。通常、ファクトテーブルには、メジャーを含む列とディメンションテーブルへの外部キーを含む列の 2 種類の列が含まれます。

フェイルファスト

開発ライフサイクルを短縮するために頻繁で段階的なテストを使用する哲学。これはアジャイルアプローチの重要な部分です。

障害分離境界

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

機能ブランチ

「ブランチ」を参照してください。

特徴量

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

特徴量重要度

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

機能変換

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

FGAC

「きめ細かなアクセスコントロール」を参照してください。

きめ細かなアクセス制御 (FGAC)

複数の条件を使用してアクセス要求を許可または拒否すること。

フラッシュカット移行

段階的なアプローチを使用するのではなく、変更データキャプチャによる継続的なデータレプリケーションを使用して、可能な限り短い時間でデータを移行するデータベース移行方法。目的はダウンタイムを最小限に抑えることです。

G

ジオブロッキング

「地理的制限」を参照してください。

地理的制限 (ジオブロッキング)

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

Gitflow ワークフロー

下位環境と上位環境が、ソースコードリポジトリでそれぞれ異なるブランチを使用する方法。Gitflow ワークフローはレガシーと見なされ、トランクベースのワークフローはモダンで推奨されるアプローチです。

グリーンフィールド戦略

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

ガードレール

組織単位 (OU) 全般のリソース、ポリシー、コンプライアンスを管理するのに役立つ概略的なルール。予防ガードレールは、コンプライアンス基準に一致するようにポリシーを実施します。これらは、サービスコントロールポリシーと IAM アクセス許可の境界を使用して実装

されます。検出ガードレールは、ポリシー違反やコンプライアンス上の問題を検出し、修復のためのアラートを発信します。これらは、 AWS Config、、Amazon AWS Security Hub、GuardDuty、Amazon Inspector AWS Trusted Advisor、およびカスタム AWS Lambda チェックを使用して実装されます。

H

HA

「高可用性」を参照してください。

異種混在データベースの移行

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

ハイアベイラビリティ (HA)

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

ヒストリアンのモダナイゼーション

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

同種データベースの移行

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

ホットデータ

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

ホットフィックス

本番環境の重大な問題を修正するために緊急で配布されるプログラム。緊急性のため、通常、修正は一般的な DevOps リリースワークフローの外で行われます。

ハイパーケア期間

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

I

IaC

「Infrastructure as Code」を参照してください。

ID ベースのポリシー

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

アイドル状態のアプリケーション

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

IIoT

「産業モノのインターネット」を参照してください。

イミュータブルインフラストラクチャ

既存のインフラストラクチャを更新、パッチ適用、または変更するのではなく、本番ワークロード用の新しいインフラストラクチャをデプロイするモデル。イミュータブルなインフラストラクチャは、本質的にミュータブルなインフラストラクチャ よりも一貫性、信頼性、予測性が高くなります。詳細については、 AWS Well-Architected フレームワークの「イミュータブルインフラストラクチャを使用したデプロイ」のベストプラクティスを参照してください。

インバウンド (受信) VPC

AWS マルチアカウントアーキテクチャでは、アプリケーション外からのネットワーク接続を受け入れ、検査し、ルーティングする VPC。AWS Security Reference Architecture では、アプリ

ケーションとより広範なインターネット間の双方向のインターフェイスを保護するために、インバウンド、アウトバウンド、インスペクションの各 VPC を使用してネットワークアカウントを設定することを推奨しています。

増分移行

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

インダストリー 4.0

接続、リアルタイムデータ、自動化、分析、AI/ML の進歩を通じて、 のビジネスプロセスのモダナイゼーションを指すために 2016 年に Klaus Schwab によって導入された用語。

インフラストラクチャ

アプリケーションの環境に含まれるすべてのリソースとアセット。

Infrastructure as Code (IaC)

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

産業分野における IoT (IIoT)

製造、エネルギー、自動車、ヘルスケア、ライフサイエンス、農業などの産業部門におけるインターネットに接続されたセンサーやデバイスの使用。詳細については、「Building an industrial Internet of Things (IIoT) digital transformation strategy」を参照してください。

インスペクション VPC

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

IoT

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

解釈可能性

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

IoT

「モノのインターネット」を参照してください。

IT 情報ライブラリ (ITIL)

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

IT サービス管理 (ITSM)

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

ITIL

「IT 情報ライブラリ」を参照してください。

ITSM

「IT サービス管理」を参照してください。

L

ラベルベースアクセス制御 (LBAC)

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

ランディングゾーン

ランディングゾーンは、スケーラブルで安全な、適切に設計されたマルチアカウント AWS 環境です。これは、組織がセキュリティおよびインフラストラクチャ環境に自信を持ってワークロー

ドとアプリケーションを迅速に起動してデプロイできる出発点です。ランディングゾーンの詳細については、安全でスケーラブルなマルチアカウント AWS 環境のセットアップ を参照してください。

大規模な移行

300 台以上のサーバの移行。

LBAC

「ラベルベースのアクセスコントロール」を参照してください。

最小特権

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

リフトアンドシフト

「7 Rs」を参照してください。

リトルエンディアンシステム

最下位バイトを最初に格納するシステム。エンディアンネス も参照してください。

下位環境

「環境」を参照してください。

M

機械学習 (ML)

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

メインブランチ

「ブランチ」を参照してください。

マルウェア

コンピュータのセキュリティまたはプライバシーを侵害するように設計されているソフトウェア。マルウェアは、コンピュータシステムの中断、機密情報の漏洩、不正アクセスにつながる

可能性があります。マルウェアの例としては、ウイルス、ワーム、ランサムウェア、トロイの木馬、スパイウェア、キーロガーなどがあります。

マネージドサービス

AWS のサービス がインフラストラクチャレイヤー、オペレーティングシステム、プラットフォーム AWS を運用し、ユーザーがエンドポイントにアクセスしてデータを保存および取得します。Amazon Simple Storage Service (Amazon S3) と Amazon DynamoDB は、 マネージドサービスの例です。これらは抽象化されたサービス とも呼ばれます。

製造実行システム (MES)

生産プロセスを追跡、モニタリング、文書化、制御するためのソフトウェアシステム。これにより、加工品を現場の完成製品に変換します。

MAP

「移行促進プログラム」を参照してください。

メカニズム

ツールを作成し、ツールの導入を推進し、調整のために結果を検査する完全なプロセス。メカニズムは、動作中にそれ自体を強化して改善するサイクルです。詳細については、 AWS Well-Architected フレームワークの「メカニズムの構築」を参照してください。

メンバーアカウント

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

MES

「製造実行システム」を参照してください。

メッセージキューイングテレメトリトランスポート (MQTT)

リソースに制約のある IoT デバイス用の、パブリッシュ/サブスクライブパターンに基づく軽量の machine-to-machine (M2M) 通信プロトコル。

マイクロサービス

明確に定義された API を介して通信し、通常は小規模な自己完結型のチームが所有する、小規模で独立したサービスです。例えば、保険システムには、販売やマーケティングなどのビジネス機能、または購買、請求、分析などのサブドメインにマッピングするマイクロサービスが含まれる場合があります。マイクロサービスの利点には、俊敏性、柔軟なスケーリング、容易なデプロ

イ、再利用可能なコード、回復力などがあります。詳細については、AWS 「サーバーレスサービスを使用したマイクロサービスの統合」を参照してください。

マイクロサービスアーキテクチャ

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

Migration Acceleration Program (MAP)

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

大規模な移行

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

移行ファクトリー

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

移行メタデータ

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

移行パターン

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

Migration Portfolio Assessment (MPA)

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

移行準備状況評価 (MRA)

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

移行戦略

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

ML

「機械学習」を参照してください。

モダナイゼーション

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

モダナイゼーション準備状況評価

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

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

モノリシックアプリケーション (モノリス)

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

MPA

「移行ポートフォリオ評価」を参照してください。

MQTT

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

多クラス分類

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

変更可能なインフラストラクチャ

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

O

OAC

「オリジンアクセスコントロール」を参照してください。

OAI

「オリジンアクセスアイデンティティ」を参照してください。

OCM

「組織変更管理」を参照してください。

オフライン移行

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

OI

「オペレーション統合」を参照してください。

OLA

「運用レベルの契約」を参照してください。

オンライン移行

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

OPC-UA

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

オープンプロセス通信 - 統合アーキテクチャ (OPC-UA)

産業用オートメーション用の machine-to-machine (M2M) 通信プロトコル。OPC-UA は、データの暗号化、認証、認可スキームを備えた相互運用性標準を提供します。

オペレーショナルレベルアグリーメント (OLA)

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

運用準備状況レビュー (ORR)

インシデントや潜在的な障害の理解、評価、防止、または範囲の縮小に役立つ質問とそれに関連するベストプラクティスのチェックリスト。詳細については、 AWS Well-Architected フレームワークの「運用準備状況レビュー (ORR)」を参照してください。

運用テクノロジー (OT)

産業運用、機器、インフラストラクチャを制御するために物理環境と連携するハードウェアおよびソフトウェアシステム。製造では、OT と情報技術 (IT) システムの統合が、Industry 4.0 トランスフォーメーションの主要な焦点です。

オペレーション統合 (OI)

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

組織の証跡

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

組織変更管理 (OCM)

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

オリジンアクセスコントロール (OAC)

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

オリジンアクセスアイデンティティ (OAI)

では CloudFront、Amazon S3 コンテンツを保護するためのアクセスを制限するオプションです。OAI を使用すると、 は Amazon S3 が認証できるプリンシパル CloudFront を作成します。認証されたプリンシパルは、特定の CloudFront ディストリビューションを介してのみ S3 バケット内のコンテンツにアクセスできます。OACも併せて参照してください。OAC では、より詳細な、強化されたアクセスコントロールが可能です。

ORR

「運用準備状況レビュー」を参照してください。

OT

「運用技術」を参照してください。

アウトバウンド (送信) VPC

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

P

アクセス許可の境界

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

個人を特定できる情報 (PII)

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

PII

「個人を特定できる情報」を参照してください。

プレイブック

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

PLC

「プログラム可能なロジックコントローラー」を参照してください。

PLM

「製品ライフサイクル管理」を参照してください。

ポリシー

アクセス許可の定義 (アイデンティティベースのポリシー を参照) 、アクセス条件の指定 (リソースベースのポリシー を参照) 、または の組織内のすべてのアカウントに対する最大アクセス許可の定義 AWS Organizations (サービスコントロールポリシー を参照) が可能なオブジェクト。

多言語の永続性

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

ポートフォリオ評価

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

述語

`true` または を返すクエリ条件。`false`通常は `WHERE`句にあります。

述語のプッシュダウン

転送前にクエリ内のデータをフィルタリングするデータベースクエリ最適化手法。これにより、リレーショナルデータベースから取得して処理する必要があるデータの量が減少し、クエリのパフォーマンスが向上します。

予防的コントロール

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

プリンシパル

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

プライバシーバイデザイン

エンジニアリングプロセス全体を通してプライバシーを考慮に入れたシステムエンジニアリングのアプローチ。

プライベートホストゾーン

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

プロアクティブコントロール

非準拠のリソースのデプロイを防止するように設計されたセキュリティコントロール。これらのコントロールは、プロビジョニング前にリソースをスキャンします。リソースがコントロールに準拠していない場合、プロビジョニングされません。詳細については、 AWS Control Tower ドキュメントの「 コントロールリファレンスガイド」および「 でのセキュリティコントロールの実装」の「プロアクティブコントロール」を参照してください。 AWS

製品ライフサイクル管理 (PLM)

設計、開発、発売から成長と成熟まで、製品のデータとプロセスのライフサイクル全体にわたる管理、および辞退と削除。

本番環境

「環境」を参照してください。

プログラミング可能ロジックコントローラー (NAL)

製造では、マシンをモニタリングし、承認プロセスを自動化する、信頼性が高く、適応性の高いコンピュータです。

仮名化

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

パブリッシュ/サブスクライブ (pub/sub)

マイクロサービス間の非同期通信を可能にするパターン。スケーラビリティと応答性を向上させます。例えば、マイクロサービスベースの MES では、マイクロサービスは他のマイクロサービスがサブスクライブできるチャネルにイベントメッセージを発行できます。システムは、公開サービスを変更せずに新しいマイクロサービスを追加できます。

Q

クエリプラン

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

クエリプランのリグレッション

データベースサービスのオプティマイザーが、データベース環境に特定の変更が加えられる前に選択されたプランよりも最適性の低いプランを選択すること。これは、統計、制限事項、環境設

定、クエリパラメータのバインディングの変更、およびデータベースエンジンの更新などが原因である可能性があります。

R

RACI マトリックス

責任、説明責任、相談、通知 (RACI) を参照してください。

ランサムウェア

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

RASCI マトリックス

責任、説明責任、相談、通知 (RACI) を参照してください。

RCAC

「行と列のアクセスコントロール」を参照してください。

リードレプリカ

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

再構築

「7 Rs」を参照してください。

目標復旧時点 (RPO)

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

目標復旧時間 (RTO)

サービスの中断から復旧までの最大許容遅延時間。

リファクタリング

「7 Rs」を参照してください。

リージョン

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

回帰

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

リホスト

「7 R」を参照してください。

リリース

デプロイプロセスで、変更を本番環境に昇格させること。

再配置

「7 Rs」を参照してください。

プラットフォーム変更

「7 Rs」を参照してください。

再購入

「7 Rs」を参照してください。

回復性

中断に耐えたり、中断から回復したりするアプリケーションの機能。で障害耐性を計画する場合、高可用性とディザスタリカバリが一般的な考慮事項です AWS クラウド。詳細については、AWS クラウド 「レジリエンス」を参照してください。

リソースベースのポリシー

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

実行責任者、説明責任者、協業先、報告先 (RACI) に基づくマトリックス

移行活動とクラウド運用に関わるすべての関係者の役割と責任を定義したマトリックス。マトリックスの名前は、マトリックスで定義されている責任の種類、すなわち責任 (R) 、説明責任

(A) 、協議 (C) 、情報提供 (I) に由来します。サポート (S) タイプはオプションです。サポートを含めると、そのマトリックスは RASCI マトリックスと呼ばれ、サポートを除外すると RACI マトリックスと呼ばれます。

レスポンシブコントロール

有害事象やセキュリティベースラインからの逸脱について、修復を促すように設計されたセキュリティコントロール。詳細については、Implementing security controls on AWSのResponsive controlsを参照してください。

保持

「7 Rs」を参照してください。

廃止

「7 R」を参照してください。

ローテーション

攻撃者が認証情報にアクセスすることをより困難にするために、シークレットを定期的に更新するプロセス。

行と列のアクセス制御 (RCAC)

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

RPO

「目標復旧時点」を参照してください。

RTO

「目標復旧時間」を参照してください。

ランブック

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

S

SAML 2.0

多くの ID プロバイダー (IdPs) が使用するオープンスタンダード。この機能により、フェデレーティッドシングルサインオン (SSO) が有効になるため、ユーザーは にログイン AWS

Management Console したり AWS 、 API オペレーションを呼び出したりできます。組織内のすべてのユーザーに対して IAM でユーザーを作成する必要はありません。SAML 2.0 ベースのフェデレーションの詳細については、IAM ドキュメントのSAML 2.0 ベースのフェデレーションについてを参照してください。

SCADA

「監視コントロールとデータ収集」を参照してください。

SCP

「サービスコントロールポリシー」を参照してください。

シークレット

では AWS Secrets Manager、暗号化された形式で保存するパスワードやユーザー認証情報などの機密情報または制限付き情報。シークレット値とそのメタデータで構成されます。シークレット値は、バイナリ、単一の文字列、または複数の文字列にすることができます。詳細については、Secrets Manager ドキュメントの「Secrets Manager シークレットの内容」を参照してください。

セキュリティコントロール

脅威アクターによるセキュリティ脆弱性の悪用を防止、検出、軽減するための、技術上または管理上のガードレール。セキュリティコントロールには、予防的 、検出的 、???応答的 、プロアクティブ の 4 つの主なタイプがあります。

セキュリティ強化

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

Security Information and Event Management (SIEM) システム

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

セキュリティレスポンスの自動化

セキュリティイベントに自動的に応答または修正するように設計された、事前定義されプログラムされたアクション。これらのオートメーションは、セキュリティのベストプラクティスを実装するのに役立つ検出的または応答的な AWS セキュリティコントロールとして機能します。自動

レスポンスアクションの例としては、VPC セキュリティグループの変更、Amazon EC2 インスタンスへのパッチ適用、認証情報のローテーションなどがあります。

サーバー側の暗号化

送信先にあるデータの、それを受け取る AWS のサービス による暗号化。

サービスコントロールポリシー (SCP)

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

サービスエンドポイント

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

サービスレベルアグリーメント (SLA)

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

サービスレベルインジケータ (SLI)

エラー率、可用性、スループットなど、サービスのパフォーマンス側面の測定。

サービスレベルの目標 (SLO)

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

責任共有モデル

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

SIEM

「セキュリティ情報とイベント管理システム」を参照してください。

単一障害点 (SPOF)

システムを中断させる可能性のあるアプリケーションの単一の重要なコンポーネントの障害。

SLA

「サービスレベルアグリーメント」を参照してください。

SLI

「サービスレベルインジケータ」を参照してください。

SLO

「サービスレベルの目標」を参照してください。

split-and-seed モデル

モダナイゼーションプロジェクトのスケーリングと加速のためのパターン。新機能と製品リリースが定義されると、コアチームは解放されて新しい製品チームを作成します。これにより、お客様の組織の能力とサービスの拡張、デベロッパーの生産性の向上、迅速なイノベーションのサポートに役立ちます。詳細については、「」の「アプリケーションをモダナイズするための段階的アプローチ AWS クラウド」を参照してください。

SPOF

単一障害点 を参照してください。

star スキーマ

トランザクションデータまたは測定データを保存するために 1 つの大きなファクトテーブルを使用し、データ属性を保存するために 1 つ以上の小さなディメンションテーブルを使用するデータベース組織構造。この構造は、データウェアハウスまたはビジネスインテリジェンスの目的で使用するように設計されています。

strangler fig パターン

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

サブネット

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

監視統制とデータ収集 (SCADA)

製造では、ハードウェアとソフトウェアを使用して物理アセットと生産オペレーションをモニタリングするシステム。

対称暗号化

データの暗号化と復号に同じキーを使用する暗号化のアルゴリズム。

合成テスト

ユーザーインタラクションをシミュレートして潜在的な問題を検出したり、パフォーマンスをモニタリングしたりする方法でシステムをテストします。Amazon CloudWatch Synthetics を使用してこれらのテストを作成できます。

T

タグ

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

ターゲット変数

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

タスクリスト

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

テスト環境

「環境」を参照してください。

トレーニング

お客様の機械学習モデルに学習するデータを提供すること。トレーニングデータには正しい答えが含まれている必要があります。学習アルゴリズムは入力データ属性をターゲット (お客様が予測したい答え) にマッピングするトレーニングデータのパターンを検出します。これらのパター

ンをキャプチャする機械学習モデルを出力します。そして、お客様が機械学習モデルを使用して、ターゲットがわからない新しいデータでターゲットを予測できます。

トランジットゲートウェイ

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

トランクベースのワークフロー

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

信頼されたアクセス

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

チューニング

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

ツーピザチーム

2 つのピザを食べることができる小さな DevOps チーム。ツーピザチームの規模では、ソフトウェア開発におけるコラボレーションに最適な機会が確保されます。

U

不確実性

予測機械学習モデルの信頼性を損なう可能性がある、不正確、不完全、または未知の情報を指す概念。不確実性には、次の 2 つのタイプがあります。認識論的不確実性は、限られた、不完全なデータによって引き起こされ、弁論的不確実性 は、データに固有のノイズとランダム性によって引き起こされます。詳細については、深層学習システムにおける不確実性の定量化 ガイドを参照してください。

未分化なタスク

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

上位環境

「環境」を参照してください。

V

バキューミング

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

バージョンコントロール

リポジトリ内のソースコードへの変更など、変更を追跡するプロセスとツール。

VPC ピアリング

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

脆弱性

システムのセキュリティを脅かすソフトウェアまたはハードウェアの欠陥。

W

ウォームキャッシュ

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

ウォームデータ

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

ウィンドウ関数

現在のレコードに関連する行のグループに対して計算を実行する SQL 関数。ウィンドウ関数は、移動平均の計算や、現在の行の相対位置に基づく行の値へのアクセスなどのタスクの処理に役立ちます。

ワークロード

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

ワークストリーム

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

WORM

「書き込み 1 回」を参照し、多くの を読み取ります。

WQF

「AWS ワークロード認定フレームワーク」を参照してください。

Write Once, Read Many (WORM)

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

Z

ゼロデイエクスプロイト

ゼロデイ脆弱性 を利用する攻撃、通常はマルウェア。

ゼロデイ脆弱性

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

ゾンビアプリケーション

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

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