AWS クラウド導入フレームワーク: プラットフォームの視点

AWS 規範ガイダンス

AWS 規範ガイダンス: AWS クラウド導入フレームワーク: プラットフォームの視点

Table of Contents

ようこそ 1
序章 2
プラットフォーム アーキテクチャ 5
Start 5
マルチアカウント戦略を定義する 5
予防的コントロールを管理する 5
組織単位の構造を定義する 6
ネットワーク接続を定義する 6
DNS 戦略を定義する 7
タグ付け標準を定義する 7
オブザーバビリティ戦略を定義する 8
高度化 8
プロアクティブおよび検出コントロールを定義する 8
サービスオンボーディングの標準を定義する 8
パターンと原則を定義する 8
Excel 9
修復パターンを定義する 9
ポリシーを共有し改善する 9
財務管理の能力を理解する 9
プラットフォームエンジニアリング 10
Start 11
ランディングゾーンを構築し、ガードレールをデプロイする 11
認証の仕組みを確立する 11
ネットワークをデプロイする 11
イベントとログデータを収集、集約、保護する 11
コントロールを確立する 12
クラウド財務管理はどのように実装しますか? 12
高度化 12
インフラストラクチャ自動化を構築する 12
一元化したオブザーバビリティサービスを提供する 13
システム管理と AMI ガバナンスを実装する 13
認証情報の使用を管理する 13
適切なセキュリティツールを導入する 14
Excel 14

ID コンストラクトのソースからの取り込みと配布を自動化する 14
異常なパターンの検出およびアラート機能を環境全体に追加する 14
脅威対策に必要な分析およびモデル化を行う 14
アクセス許可情報の収集およびレビューとアクセス権限管理の改善を継続的に行う 15
プラットフォームメトリクスを選択して測定し、継続的に改善する 15
データアーキテクチャ 16
Start 16
包括的な能力を定義する 16
データゾーンを整理する 17
データの俊敏性確保および民主化を計画する 17
安全なデータ提供を定義する 17
費用対効果を得るための計画を立てる 17
高度化 18
特徴量エンジニアリングを理解する 18
データセット非正規化の計画を立てる 18
移植性とスケーラビリティを設計する 18
Excel 19
設定可能なフレームワークを設計する 19
統合分析エンジンの構築計画を立てる 19
DataOps を定義する 19
データエンジニアリング 21
Start 21
データレイクをデプロイする 21
データインジェストのパターンを開発する 21
データ処理を高速化する 23
データ可視化のサービスを提供する 23
高度化 24
ほぼリアルタイムのデータ処理を実装する 24
データ品質を検証する 24
データ変換サービスの有効性を証明する 24
データ民主化を実現する 25
Excel 25
UI ベースのオーケストレーション機能を利用可能にする 25
DataOps を統合する 26
プロビジョニングとオーケストレーション 27
Start 27

ハブアンドスポークカタログモデルをデプロイする 27
再利用を目的にテンプレートをキュレートする 27
再利用を目的にデフォルトパラメータを適用する 28
承認プロセスを確立する 28
高度化 28
セルフサービスポータルを作成する 28
プライベートマーケットプレイスを利用可能にする 28
エンタイトルメントを管理する 29
Excel 29
調達システムと統合する 29
ITSM ツールと統合する 29
ライフサイクル管理とバージョン配布システムを実装する 29
最新のアプリケーションの開発 31
Start 31
最新のアプローチを調査し考察する 31
クラウドネイティブコンピューティング機能を導入する 32
コンテナ化を使用する 32
最新のデータベースを使用する 32
高度化 33
最新のアーキテクチャを最適化する 33
サービスメッシュ技術を使用する 33
可視性とトレーサビリティを確保する 34
Excel 34
マイクロサービスを採用する 34
継続的インテグレーションと継続的デリバリー 36
Start 36
ソフトウェアコンポーネント管理を導入する 36
CI/CD パイプラインを作成する 36
自動テストをデプロイする 37
ドキュメントを作成する 37
Infrastructure as Code を使用する 38
標準メトリクスを維持および追跡します。 38
高度化 39
設定管理ツールを使用する 39
モニタリングと ログ記録を統合する 39
マージのケイデンスを決定する 39

デプロイ後の動作をキャプチャする 39
Excel 40
AI/ML 技術を統合する 41
カオスエンジニアリングプラクティスを導入する 41
パフォーマンスの最適化 41
高度なオブザーバビリティを実装する 42
GitOps プラクティスを実装する 43
結論 44
詳細情報 45
寄稿者 46
ドキュメント履歴 47
用語集 48
..... 48
A 49
B 51
C 53
D 56
E 60
F 63
G 64
H 65
I 67
L 69
M 70
O 74
P 77
Q 80
R 80
S 83
T 87
U 88
V 89
W 89
Z 90
..... xci

AWS クラウド導入フレームワーク: プラットフォームの視点

Amazon Web Services (寄稿者)

2023 年 10 月 (ドキュメント履歴)

デジタルトランスフォーメーション (DX) は、カスタマーエクスペリエンス、イノベーション、柔軟性の向上を目指すエグゼクティブにとって、最も効果的な取り組みと言えます。DX では、機械学習 (ML)、人工知能 (AI)、ビッグデータ、クラウドのスピードとスケーリングを利用して、変化し続けるビジネス状況および顧客ニーズを満たします。

Amazon Web Services (AWS) は、最も包括的で広く採用されているクラウドプラットフォームであり、これを利用することで、組織変革だけでなく、ビジネスリスクの軽減、環境、社会、ガバナンス (ESG) のパフォーマンス向上、収益拡大、運用のさらなる効率化などを実現しやすくなります。

AWS クラウド導入フレームワーク (AWS CAF) の AWS ベストプラクティスを実装すると、ビジネス成果の実現が加速します。AWS CAF の利用により、トランスフォーメーションの機会を特定して優先順位付けする、クラウドの準備状況を評価し改善する、トランスフォーメーションロードマップをイテレーションによって発展させる、なども可能になります。

AWS CAF のガイダンスは、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、オペレーションという 6 つの視点でグループ化されています。各視点については、個別のガイドで説明します。本ガイドでは、プラットフォームの視点を取り上げます。この視点は、エンタープライズグレードのスケーラブルなハイブリッドクラウド環境で迅速なクラウドワークロードデリバリーを実現することに重点が置かれています。

序章

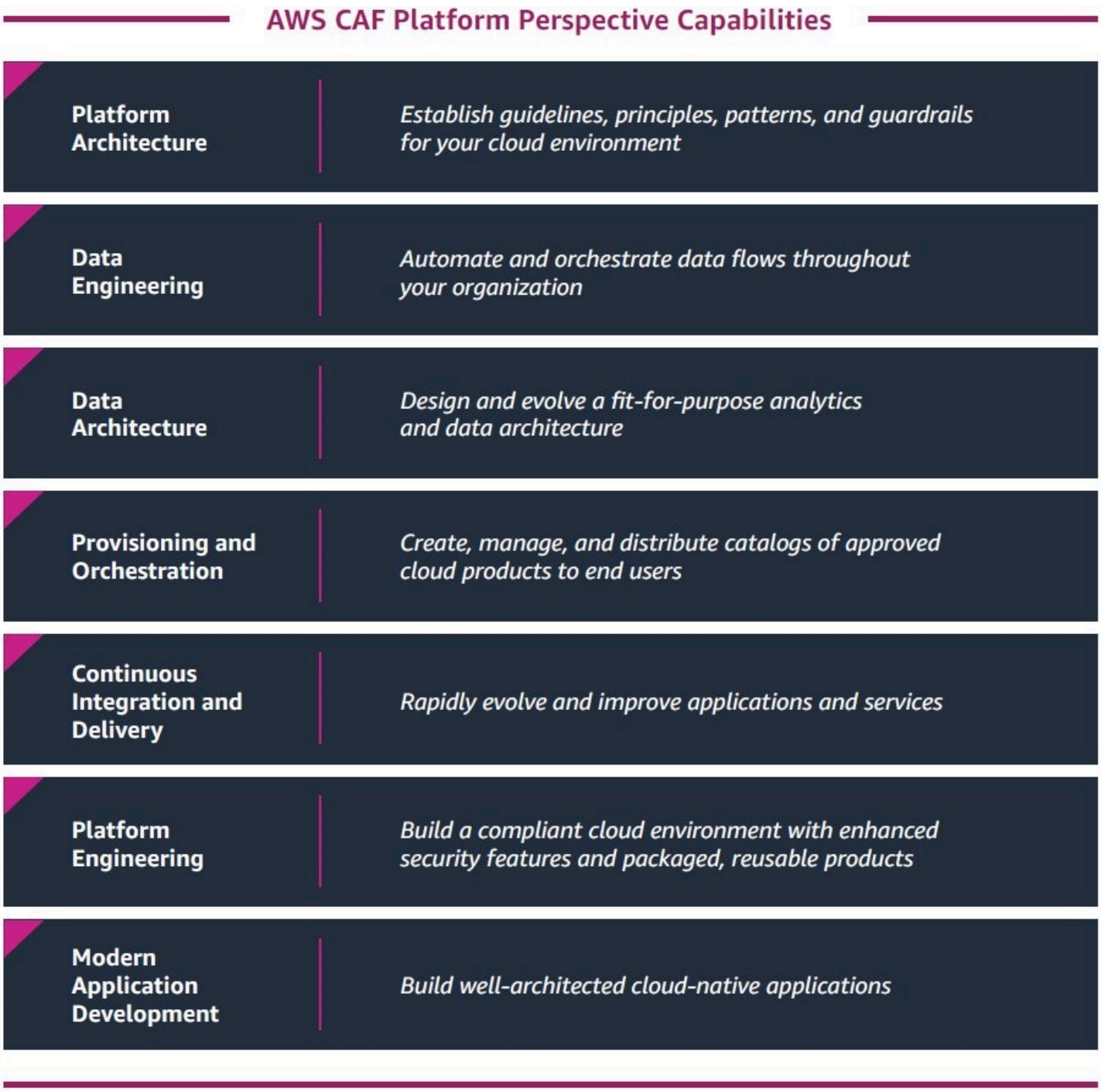
急成長しているスタートアップ、大企業、主要な政府機関といった、何百万ものお客様が AWS を利用しています。(AWS ウェブサイトの「お客様のクラウド導入事例」をご覧ください) こうしたお客様は、レガシーワークロードの移行とモダナイズ、データ駆動型運用への移行、ビジネスプロセスの自動化と最適化、運用モデルの再構築などを実現しているほか、ビジネスリスクの軽減、環境、社会、ガバナンス (ESG) の実績向上、収益拡大、運用の効率化を進め、ビジネス成果を改善できています。

組織がクラウドを効果的に使用してデジタルトランスフォーメーションを行える能力 (クラウドへの対応準備を整えられる組織上の能力) は、一連の基本的能力によって支えられています。ここでの能力とは、プロセスを使用してリソース (人材やテクノロジーといった、あらゆる有形または無形の資産) を投入して成果を上げる組織的能力を意味します。AWS CAF では、こうした能力を特定し、規範ガイダンスを提示していますが、世界中の何千もの組織が、このガイダンスを基に、クラウド準備状況の改善と、クラウドトランスフォーメーションジャーニーの迅速化を効果的に進めています。

AWS CAF では、その能力を 6 つの視点にグループ化しています。

- ビジネス
- 人員
- ガバナンス
- プラットフォーム
- セキュリティ
- オペレーション

プラットフォームの視点は、エンタープライズグレードのスケーラブルなハイブリッドクラウド環境を構築し、迅速なクラウドワークロードデリバリーを実現することに重点が置かれています。この環境は、次の図に示す 7 つの能力で構成されます。こうした能力を管理するのは、クラウドトランスフォーメーションジャーニーに機能的にかかわるステークホルダーであり、一般的な利害関係者として、最高技術責任者 (CTO)、テクノロジーリーダー、アーキテクト、エンジニアなどが挙げられます。



こうした能力については、本ガイドの以下のセクションで説明します。各セクションでは、特定の能力開発に着手し、能力を向上させ、最終的に卓越したレベルにまで高めるためのガイドラインを提示します。

- プラットフォーム アーキテクチャ
- プラットフォームエンジニアリング

- データアーキテクチャ
- データエンジニアリング
- プロビジョニングとオーケストレーション
- 最新のアプリケーションの開発
- 継続的インテグレーションと継続的デリバリー (CI/CD)

プラットフォームの視点は、CAF AWS で重要な役割を果たします。中心的な役割も担っており、これを基に、他のすべての視点で行われた決定を収束させ、ビジネスの俊敏性と価値を実現することになります。ここで行った決定は、基本的なレベルで、ビジネス目標達成の助けにも、妨げにもなるでしょう。AWS CAF における、プラットフォームの視点を持つと、組織の変革を支える、エンタープライズグレードのスケーラブルなクラウド環境を容易に構築できます。AWS CAF のこの視点で、堅牢なプラットフォームを確立して、クラウドジャーニーを推進することが、大きなビジネス変革と成長につながります。

プラットフォームの観点から取り組みを進める際には、開発に必要なビジネスリーダーとの部門横断的な関係構築や、そうしたリーダーがチームや組織にもたらす価値を考慮してください。また、運用モデルの変更とチームトポロジーに重点を置き、要件を満たすようにします。さらに、プラットフォーム構築に必要なチームスキルを開発して、各アプリケーションチームでそのプラットフォームを利用できるようにします。これらについて、決定を下す際には、組織の人材、ビジネス、ガバナンス、セキュリティ、運用目標を念頭に置いてください。これらは、プラットフォームを導入し、その取り組みを成功させる上で不可欠だからです。

AWS と AWS パートナーネットワークでは、ワークショップやトレーニングなどのツールとサービスを提供しており、これらは、このジャーニーでセキュリティ体制を実装し改善するのに有用です。AWSプロフェッショナルサービスでは、エキスパートのグローバルチームを利用できます。こうしたチームが、AWS CAF に基づくサービスコレクションを通じて、クラウドトランスフォーメーションに関する特定の成果の実現をサポートします。

プラットフォーム アーキテクチャ

クラウド環境のガイドライン、原則、パターン、ガードレールを確立し、それらを維持します。

クラウド環境を適切に設計すると、実装の迅速化、リスクの軽減、クラウド導入の推進を実現しやすくなり、プラットフォームアーキテクチャの能力を持つことで、クラウド導入推進のためのエンタープライズ標準について、組織内でコンセンサスが形成されます。認証、セキュリティ、ネットワーク、ログ記録、モニタリングを容易にするには、ベストプラクティスの設計図とガードレールを定義します。さらに、レイテンシー、データ処理、データレジデンシーの要件を満たせるよう、オンプレミスでの保持が必要なワークロードを考慮して計画を立てるとともに、クラウドでのバースト、クラウドを利用したバックアップおよびディザスタリカバリ、分散データ処理、エッジコンピューティングといった、ハイブリッドクラウドのユースケースを評価します。

Start

マルチアカウント戦略を定義する

効果的なマルチアカウント戦略を立てるには、スケールと運用効率上の懸念点を考察します。つまり、ワークロードを分離して、運用ニーズを最も満たす論理パターンに当てはめます。エンタープライズ内の集中型および分散型サービスに対応するには、最初に基本的なアカウントセットを管理し、セキュリティ、財務、運用の各機能を一元化して、分散型および自律型のチームとアカウントに、効果的な管理やガバナンスを行うと良いでしょう。組織全体で同じ認識を持てるよう調整し、プラットフォームとワークロードがどのようにセグメント化および管理されるかを理解する必要があります。こうした構造を理解すると、許容可能なプラットフォーム利用ポリシーの変化に適合しつつ、認証および認可のセキュリティ原則を定着させることができます。

予防的コントロールを管理する

一連のデフォルトコントロール (ガードレール) を取り入れた安全なマルチアカウント環境を計画します。サービスコントロールポリシー (SCP) などの仕組みを理解して利用し、組織全体でのサービス利用を管理します。クラウドプラットフォーム内で利用可能な AWS リージョン リージョンのサービスも管理対象となります。ポリシーを使用すると、その一元化の仕組みによって、すべてのアカウントについて使用可能な最大アクセス許可を管理し、組織のアクセスコントロールガイドラインに準拠できます。

組織単位の構造を定義する

組織単位 (OU) を使用すると、規制要件や、ソフトウェア開発ライフサイクル (SDLC) 環境に基づいて、アカウントを実用的に管理および分類でき、クラウドインフラストラクチャ全体に適切なポリシーとアクセス許可を適用するプロセスも合理化されます。ワークロード OU は、アプリケーションインフラストラクチャのリソースに対応するアカウント専用に設計するもので、これには、適切なポリシーを適用します。OU と SCP を使用すると、アプリケーションおよびサービス運用を常に円滑にすると同時に、組織のクラウドインフラストラクチャでセキュリティとコンプライアンスを強化できます。以上の実践が、最終的には、より効率的で堅牢なクラウド導入プロセスの実現につながります。

ネットワーク接続を定義する

ネットワーク接続は、アプリケーションとワークロードの基盤となる、セキュリティ、スケーラビリティ、可用性に優れたネットワークの構築を支えているという点で、クラウドインフラストラクチャの重要な側面を担っています。ネットワークを適切に設計すると、一貫性のある高パフォーマンス、異なる環境間でのシームレスなオペレーションが実現します。

ネットワークアーキテクチャの設計時には、レイテンシー、データ処理、データレジデンシーなどの要件上、オンプレミスでの保持が必要なワークロードが存在するかどうかを考慮してください。クラウドでのバースト、クラウドを利用したバックアップおよびディザスタリカバリ、分散データ処理、エッジコンピューティングといったハイブリッドクラウドのユースケースを評価すると、以下の側面について主要な要件を特定できます。

- ネットワークとインターネット間の接続。この側面を実現するには、アプリケーションまたはワークロードとインターネット間で、セキュリティと信頼性に優れた接続を可能にしなければなりません。こうした接続は、例えば、ウェブベースリソースへのアクセスを容易にする、ユーザーとアプリケーション間の通信を可能にする、必要に応じてサービスを外部からアクセス可能にするといった運用に不可欠です。
- クラウド環境間の接続。この領域で重点が置かれているのは、クラウドインフラストラクチャ内のさまざまなコンポーネントとサービス間に堅牢な接続を確立することです。これにより、さまざまなクラウドサービス間で、データとリソースの共有およびアクセスが容易になるため、効率的なコラボレーションとスムーズな運用が促進されます。ここでは、仮想プライベートクラウド (VPC) の使用を検討することが重要です。運用を常に簡素化するには、VPC の構築および追跡方法について、標準の計画を検討してください。こうした標準をプログラムによって作成することを検討し、IP Address Manager (IPAM) ソリューションの使用も計画します。その際には、成長に見合う十分な IP スペースを割り当て、複数のアベイラビリティーゾーンを使用する場合にトラブル

シューティングが容易になるようにサブネット構造を設計します。ネットワーク接続を設計し実装するときには、「VPC のセキュリティのベストプラクティス」に従ってください。

- オンプレミスネットワークとクラウド環境間の接続。この側面は、オンプレミスインフラストラクチャとクラウドベース環境の統合を示すものです。この 2 つの間でセキュリティと信頼性に優れた接続が可能になれば、ハイブリッドアーキテクチャの利点を活用できます。例えば、オンプレミスにあるリソースとクラウドサービスを同時に使用することで、パフォーマンス、スケーラビリティ、コスト最適化が向上します。

ネットワーク接続について、この 3 つの主要領域で問題に対処することで、アプリケーションとワークロードをサポートする堅牢なクラウドインフラストラクチャを構築できるため、クラウド導入の利点を活用することが可能です。ネットワーク要件に留意し、マルチアカウント戦略に従ってスケーリング可能なシンプルな設計を行ってください。

DNS 戦略を定義する

DNS 戦略を適切に計画すると、クラウド環境の拡大に伴う複雑さを回避しやすくなります。オンプレミスで DNS の機能を維持する場合は、ハイブリッド DNS アーキテクチャを設計し、オンプレミス DNS インフラストラクチャを使用するとともに、クラウドベースの DNS 要件に合わせてクラウド DNS を使用すると良いでしょう。具体的には、リゾルバーエンドポイントと転送ルールを使用して、クラウドの DNS 解決機能をオンプレミス DNS 環境のそれと統合します。また、1 つ以上のネットワーク内にあるドメインとそのサブドメインのクエリにクラウド DNS をどう応答させるかについての情報を、プライベートホストゾーンを使用して保持します。

タグ付け標準を定義する

リソースのタグ付けは、コスト管理の効率化とリソース所有権の特定に不可欠なプラクティスです。組織でのクラウド消費をどこまで許可するかを、プラットフォーム内の特定のサービス利用も含め、検討してください。また、特定のリソースとそれをデプロイしたチームを追跡するタグ付け戦略を定義します。AWS CAF オペレーションの観点から情報収集し、デプロイ済みインフラストラクチャのタスクを、タグを使用して自動化します。

さらに、リソース関連のメタデータをタグ付けすることで、支出をグループ化し、追跡すると良いでしょう。その際には、AWS CAF ガバナンスの観点のクラウド財務管理 (CFM) 機能で指定されている組織要件に従います。財務ポリシー違反があった場合どのようなアクションを実行するかなど、会計および財務プラクティスをサポートする報告の仕組みも特定してください。

オブザーバビリティ戦略を定義する

オブザーバビリティ戦略の確立は、クラウドアーキテクチャを最適化し保護する上で、重要なステップとなります。この戦略を展開するには、クラウドサービスで生成されたメトリクスとログから実用的なインサイトを引き出し、それを戦略的意思決定に活かすことに焦点を当てます。主要業績評価のモニタリングとアラートの設定を優先し、起こり得る問題に先制して対処します。不要なツールの増加を防ぎ、コストを最適化し、組織の最重要事項に集中するには、このオブザーバビリティ戦略をプラットフォームにもアプリケーションにも取り入れてください。詳細については、「Developing an observability strategy」のプレゼンテーション (AWS re:Invent 2022) を参照してください。

高度化

プロアクティブおよび検出コントロールを定義する

能力をさらに高めるには、環境内にプロアクティブコントロールと検出コントロール (ガードレール) がどの程度必要かを特定しなければなりません。ポリシーを作成し、これによって、組織単位 (OU) 内アカウントでロールとユーザーに持たせるガードレールまたは制限を定義します。プラットフォームのデフォルトの検出ガードレールを確認し、適用するガードレールを選択します。必要に応じて追加の予防コントロールと検出コントロールを作成し、マルチアカウント戦略に沿うよう、OU 別にグループ化します。また、検出コントロールによって識別した非準拠リソースの検査に必要な組織のツールや仕組みを検討してください。

サービスオンボーディングの標準を定義する

許容できるプラットフォーム使用、サービス消費に関連するパターン、それらにガバナンスを確立する方法に、標準を定義するとともに、使用を許可する初期サービスを検討します。これらの標準を概説し、プラットフォームのユーザーおよびオペレーターに公開するためのドキュメントを作成します。さらに、こうした標準が、変化し続ける組織目標や、進歩するクラウドコンピューティング機能に、長期的に適応できるようにします。

パターンと原則を定義する

アプリケーション所有者から収集した情報に基づいて、組織内で許可するアーキテクチャパターンを検討したら、標準化のブループリント定義に着手します。標準化を行うと、クラウドでのスケーリングの際に、ガバナンスが強化され、管理上の負担も軽減されます。また、Infrastructure as Code (IaC) を使用するパターンを定義するとともに、変更管理プロセスおよび IT サービス管理 (ITSM) システムに統合したサービスカタログを使用して、簡素化したデプロイモデルを計画します。その後、

こうしたブループリントの使用方法と、例外を許可する状況を定義し、認証、セキュリティモニタリング、ガードレールを考慮しながら、これらの例外とそのガバナンスに関する計画を立てます。

Excel

修復パターンを定義する

検出ガードレールの検出結果に注釈を付け、優先順位を判断する方法を検討します。そうすることで、セキュリティとコンプライアンスのフレームワークに従って、それらを修正できます。また、ポリシー違反のリソースプロビジョニングを自動検出するための計画を立ててください。予算ポリシーやタグポリシーに違反するリソースも検出対象になります。さらに、サービスレベル目標の設定および測定に必要な能力を特定するとともに、ランブックとプレイブックを更新します。こうしたプラクティスと、フィードバックの仕組みを定期的に見直して、プラットフォームの変化に関連するデータを収集します。それに応じて、ランブックとプレイブックを作成および更新する仕組みを定義してください。

ポリシーを共有し改善する

すべてのドキュメントを対象とする、コンテンツの一元管理システムを構築して、プラットフォームのユーザーとオペレーターが利用できるようにします。また、今後のポリシー変更の考察に役立つよう、フィードバックを収集する仕組みを計画します。

財務管理の能力を理解する

大きなビジネス成果を得るには、常に、透過的かつ包括的に予算を把握することが重要です。これによって、十分な情報に基づく意思決定、リソース配分の効率化、戦略的目標の達成などが可能になります。予算を明確に把握していれば、他社を凌駕できるでしょう。なぜなら、十分な情報に基づく意思決定、効果的なリソース配分、コスト管理、パフォーマンス測定、説明責任およびコンプライアンスの維持が容易になるからです。最終的には、より効率的で、財務的に安定し、成功を収めた組織へと発展できます。タグ付け戦略の成果を得られたら、AWS Budgets のコストフィルターを使用して、リソースタグで経費をフィルタリングできます。これにより、特定のプロジェクト、部門、環境などの条件に合わせて予算を立てやすくなるため、財務管理の能力がさらに高まります。また、コスト配分タグと AWS Cost Categories をタグに関連付けると、コストを報告する際に財務上のインサイトと透明性が向上します。

プラットフォームエンジニアリング

パッケージ化された再利用可能なクラウド製品を使用でき、セキュリティとコンプライアンスが確保されたマルチアカウントクラウド環境を構築します。

開発チームを活性化してイノベーションをサポートするには、ビジネス要求に迅速に応えられるプラットフォームが必要です (AWS CAF ビジネスのパースペクティブを参照してください)。これには、製品管理の要件にきわめて柔軟に対応できる、セキュリティ上の制約に厳格に従える、運用上のニーズを迅速に満たすといった特徴も求められます。このプロセスでは、セキュリティとコンプライアンスが確保されたマルチアカウントクラウド環境を構築して、パッケージ化された再利用可能なクラウド製品をその環境で使用できるようにする必要があります。

効果的なクラウド環境があれば、新しいアカウントを簡単にプロビジョニングすると同時に、そうしたアカウントを組織のポリシーに準拠させることができます。厳選された一連のクラウド製品は、ベストプラクティスのコード化、ガバナンスの支援、クラウドデプロイの速度および一貫性の向上などを可能にします。ここで重要なのは、ベストプラクティスのブループリントと、検出および予防のガードレールをデプロイすることに加え、クラウド環境を既存環境と統合し、目的とする、ハイブリッドクラウドのユースケースを実現することです。

さらに、アカウントのプロビジョニングワークフローを自動化するとともに、複数のアカウントを使用してセキュリティとガバナンスの目標達成をサポートします。オンプレミス環境とクラウド環境間に加え、異なるクラウドアカウント間の接続をセットアップします。既存の ID プロバイダー (IdP) とクラウド環境間にフェデレーションを実装して、既存のログイン認証情報を使用してユーザーを認証できるようにし、ログ記録の一元化、クロスアカウントでのセキュリティ監査の確立、インバウンドおよびアウトバウンド DNS リゾルバーの構築、ダッシュボードでのアカウントおよびガードレールの可視化なども行います。

続いて、企業の標準と設定管理に照らして、利用するクラウドサービスを評価および認定します。エンタープライズ標準の製品 (セルフサービスのデプロイ可能な製品およびサービス) をパッケージ化し継続的に改善するとともに、Infrastructure as Code (IaC) を活用して、宣言的な形式で設定を定義します。また、イネーブルメントチームを作って、開発者やビジネスユーザーにプラットフォームの価値を伝えてもらい、こうしたユーザーがシステム連携を行えるようにします。これにより、組織全体にプラットフォームの利用を広めます。

以下のセクションで説明するタスクを完了するには、組織のプラットフォームエンジニアリングをモダナイズし発展させる能力とチームを備える必要があります。技術詳細については、「AWS上でクラウド基盤を構築する」のホワイトペーパーを参照してください。

Start

ランディングゾーンを構築し、ガードレールをデプロイする

プラットフォームエンジニアリングを成熟させるには、最初にプラットフォームアーキテクチャの能力で定義されている検出ガードレールと予防ガードレールを使用して、ランディングゾーンをデプロイする必要があります。ガードレールがあれば、アプリケーション所有者がクラウドリソースを消費する際に、そうしたユーザーによる、組織の標準への違反を防止できます。この仕組みによって、アカウントプロビジョニングのワークフローを自動化し、複数のアカウントでセキュリティとガバナンスの目標達成をサポートします。

認証の仕組みを確立する

AWS CAF セキュリティの視点で指定されている標準に従って、ID 管理とアクセスコントロールを、すべての環境、システム、ワークロード、プロセスに実装します。ワークフォース ID の場合、AWS Identity and Access Management (IAM) ユーザーによる使用を制限し、代わりに ID を一元管理できる ID プロバイダーを利用するようにします。これにより、アクセスの作成、管理、および取り消しを 1 箇所で行うことができるため、複数のアプリケーションおよびサービス間のアクセス管理を簡単に行うことができます。また、アクセス権の作成、更新、削除を管理する既存のプロセスを AWS 環境でも使用できるようにします。

ネットワークをデプロイする

自社のプラットフォームアーキテクチャ設計に従って、ネットワークアカウントの作成と一元化管理を行うとともに、自社環境との間で生じるインバウンドおよびアウトバウンドトラフィックを制御します。オンプレミスネットワークと AWS 環境間、インターネットとの間で、および AWS 環境間で、迅速にプロビジョニングされた接続を実現するようにネットワークを設計することをお勧めします。ネットワーク管理を一元化すると、ネットワークコントロールをデプロイでき、これにより、予防および事後的対応コントロールを使用して、環境全体でネットワークと接続を分離することが可能です。

イベントとログデータを収集、集約、保護する

Amazon CloudWatch のクロスアカウントオブザーバビリティを使用します。この機能により、統合インターフェイスを利用して、リンク済みアカウント全体でメトリクス、ログ、トレースを検索、可視化、分析できるため、アカウントの境界を意識しなくても済みます。

ログおよびセキュリティの一元管理に特定のコンプライアンス要件がある場合は、専用のログアーカイブアカウントの設定を検討してください。これにより、ログデータ専用に一元化および暗号化され

たリポジトリを利用できます。また、暗号化キーを定期的にローテーションして、このアーカイブのセキュリティを強化します。

必要であれば、マスキング手法を使用して、機密ログデータを保護する堅牢なポリシーを実装します。さらに、コンプライアンス、セキュリティ、監査ログにログ集約を使用し、ログ設定への不正な変更を防止できるよう、厳格なガードレールと ID 構造も使用するようにします。

コントロールを確立する

AWS CAF セキュリティの視点の定義に従って、ビジネス要件を満たす基本的なセキュリティ機能をデプロイします。追加の予防および検出コントロールもデプロイし、必要であれば、それらのコントロールを、全アカウントにプログラムで一貫してプロビジョニングします。また、プラットフォームアーキテクチャの能力で定義されているように、検出コントロールを運用ツールに統合し、運用の仕組みによって、非準拠のリソースをレビューできるようにします。

クラウド財務管理はどのように実装しますか?

AWS CAF ガバナンスの観点から、組織のタグ付け戦略をクラウド消費の財務上の説明責任と整合させるコスト配分タグと AWS Cost Categoriesを実装します。 AWS Cost Categoriesを使用すると、AWS Cost Explorerや で公開された請求データなどのツールを使用して、クラウド料金を内部コストセンターに請求または表示できますAWS Cost and Usage Report。

高度化

インフラストラクチャ自動化を構築する

次の取り組みに進む前に、プラットフォームアーキテクチャの方針に合うよう、利用するクラウドサービスを評価し、認定します。次に、エンタープライズ標準の製品 (セルフサービスのデプロイ可能な製品およびサービス) をパッケージ化し継続的に改善するとともに、Infrastructure as Code (IaC) を使用して、宣言的な形式で設定を定義します。インフラストラクチャ自動化では、ソフトウェア開発サイクルを模倣しますが、これを行うには、ロールベースのアクセスコントロール (RBAC) または属性ベースのアクセス制御 (ABAC) を使用して、各アカウントの特定のサービスにアクセスできるようにします。また、新しいアカウントを迅速にプロビジョニングし、それらをサービスおよびインシデント管理機能と連携させる方法をデプロイします。これらに対応するには、API を使用するか、セルフサービス機能を開発します。アカウントの作成時には、ネットワーク統合と IP 割り当てを自動化し、コンプライアンスとネットワークセキュリティを確保します。さらに、 AWSと連携するように設定したネイティブコネクタを使用して、新規アカウントを IT サービス管理 (ITSM) ソリューションと統合します。必要に応じて、プレイブックとランブックを更新します。

一元化したオブザーバビリティサービスを提供する

効果的なクラウドオブザーバビリティを実現するには、プラットフォームで、ローカルログデータと一元化されたログデータ両方へのリアルタイム検索および分析がサポートされている必要があります。オペレーションがスケールするにつれ、プラットフォームでログ、メトリクス、トレースをインデックス化、視覚化、解釈可能かどうかが、生データから実用的なインサイトを引き出すためのカギとなります。

ログ、メトリクス、トレースを相関させることで、実用的な結論を抽出し、必要な情報のみに基づいた対策を開発できます。ログ、メトリクス、トレースによって特定されたセキュリティイベントやパターンにプロアクティブな対応を許可するルールも確立します。 AWS ソリューションが拡張されるにつれて、モニタリング戦略が並行してスケーリングされ、オブザーバビリティ機能を維持および強化することを確認します。

システム管理と AMI ガバナンスを実装する

Amazon Elastic Compute Cloud (Amazon EC2) インスタンスを広範囲で使用する組織では、インスタンスを大規模に管理する運用ツールが必要です。多くの組織にとって、ソフトウェアアセット管理、エンドポイントでの検出および対応、インベントリ管理、脆弱性管理、アクセス管理は、基本的な機能と言えます。 こうした機能は、一般的に、インスタンスにインストール済みのソフトウェアエージェントによって提供されます。そのため、エージェントなどのカスタム設定を Amazon マシンイメージ (AMI) にパッケージ化する機能を開発し、これらの AMI をクラウドプラットフォームのコンシューマーに公開します。こうした AMI は、予防および検出コントロールを使用して管理します。AMI には、長期間稼働する EC2 インスタンスの大規模管理を可能にするツールが含まれている必要があり、とりわけ、新規 AMI が定期的に使用されず、変更が多い Amazon EC2 ワークロードの管理に、こうしたツールが必要です。AWS Systems Manager は、大規模な利用が可能で、これによって、エージェントのアップグレード自動化、システムインベントリの収集、EC2 インスタンスへのリモートアクセス、オペレーティングシステム脆弱性へのパッチ適用などを行えます。

認証情報の使用を管理する

AWS CAF セキュリティの視点に従って、ロールと一時的な認証情報を利用できるようにします。具体的には、ツールを使用して、インスタンスまたはオンプレミスシステムへのリモートアクセスを管理しますが、この管理では、プリインストールされたエージェントを使用し、シークレットは保存しません。また、長時間の認証情報への依存を減らして、IaC テンプレートにハードコードされた認証情報をスキャンします。一時的な認証情報を使用できない場合は、アプリケーショントークンやデータベースパスワードなどのプログラムツールを使用して、認証情報のローテーションと管理を自動

化します。ユーザー、グループ、ロールを、最小特権の原則を使用して IaC でコード化するとともに、ガードレールを使用して ID アカウントの手動作成を防ぐようにします。

適切なセキュリティツールを導入する

セキュリティモニタリングツールでは、インフラストラクチャ、アプリケーション、ワークロード全体に、粒度の高いセキュリティモニタリングを行え、パターン分析の集約ビューを利用できる必要があります。他のすべてのセキュリティ管理ツールと同様に、拡張検出および対応 (XDR) ツールを拡張して、AWS CAF セキュリティの観点から定義された要件 AWS に従って、 のアプリケーション、リソース、環境のセキュリティを評価、検出、対応、修復する機能を提供する必要があります。

Excel

ID コンストラクトのソースからの取り込みと配布を自動化する

ロール、ポリシー、テンプレートなどの ID コンストラクトを IaC ツールでコード化し、バージョン管理するとともに、ポリシー検証ツールを使用して、セキュリティ警告、エラー、一般的な警告、推奨の IAM ポリシー変更、その他の検出結果をチェックします。必要に応じて、環境への一時的なアクセスを自動的に許可する ID コンストラクトをデプロイまたは削除し、コンソールを使用している個人によるデプロイも禁止します。

異常なパターンの検出およびアラート機能を環境全体に追加する

環境に見られる既知の脆弱性をプロアクティブに評価し、異常なイベントやアクティビティパターンの検出機能を追加します。検出結果を確認して、プラットフォームアーキテクチャチームへの推奨事項を作成し、こうしたチームが効率化とイノベーションを加速させる変更を行えるようにします。

脅威対策に必要な分析およびモデル化を行う

AWS CAF セキュリティの視点の要件に従い、業界およびセキュリティベンチマークに照らして、継続的なモニタリングおよび測定を実装します。計測アプローチの実装時には、どのタイプのイベントデータと情報を使用すれば、セキュリティ管理機能で最も有用な知見を得られるかを判断します。このモニタリングでは、サービスの使用状況を含む、複数の攻撃ベクトルを対象にします。セキュリティ基盤には、包括的な機能、つまりマルチアカウント環境全体で安全なログ記録と分析を可能にする機能が必要であり、複数ソースのイベントを相関付ける機能も求められます。さらに、こうした設定の変更を防ぐために、具体的なコントロールおよびガードレールを設けます。

アクセス許可情報の収集およびレビューとアクセス権限管理の改善を継続的に行う

ID ロールおよびアクセス許可への変更を記録するとともに、想定した設定状態からの逸脱を検出ガードレールによって検出した際にアラートを生成する機能を実装します。また、集約およびパターン識別ツールを使用して、イベントのコレクションを一元的に確認し、必要であればアクセス権限管理を改善します。

プラットフォームメトリクスを選択して測定し、継続的に改善する

プラットフォームオペレーションで成果を上げるには、包括的なメトリクスを確立し、それらを定期的にレビューします。メトリクスは、組織の目標とステークホルダーのニーズに沿うよう、調整します。また、プラットフォームのパフォーマンスメトリクスと改善メトリクスをともに追跡するだけでなく、パッチ、バックアップ、コンプライアンスなどに関する運用上のパラメータと、チーム有効化やツール導入の指標を総合的に評価します。

CloudWatch クロスアカウントオブザーバビリティを使用すると、メトリクス管理を効率化でき、データ集約および可視化の合理化、情報に基づく意思決定、ターゲットを絞った機能強化も行えます。こうしたメトリクスを成果の指標や変化の推進要因として利用すると、継続的改善の実践環境を築くことができます。

データアーキテクチャ

目的に合ったデータおよび分析アーキテクチャを設計し、発展させます。

実用的なインサイトを得るには、優れた設計のデータおよび分析アーキテクチャが不可欠です。目的に合ったデータおよび分析アーキテクチャを設計し、発展させることで、複雑さ、コスト、技術的負債を軽減でき、増大し続けるデータから貴重なインサイトも得られます。AWS CAF の原則に従うと、既存のプラットフォームとシームレスに統合するデータアーキテクチャを構築できます。そのように CAF に沿った調整を行うことで、最新のデータ処理および分析技術がもたらす利点を活用できるのです。

データおよび分析アーキテクチャは、組織にとって、データから価値を引き出す能力のブループリントと言えます。また、新たなビジネスインサイトを得るのに役立ち、ビジネス成長の推進要因としても機能します。ビジネスニーズを満たすよう、データアーキテクチャをモダナイズするには、アーキテクチャを短期的および長期的なビジネス目標に沿うよう調整し、組織の文化的および状況的要件を独自に満たすものにしなければなりません。今日の世界において、成功したデータおよび分析アーキテクチャの実装と導入に共通しているのは、適切なデータを適切なコンシューマーが適切なタイミングで利用できるようにするという原則に基づいていることです。

以上を実現するには、どうすれば、データアセットの物理的または論理的なモデル化や、データ保護を行えるかに加え、どのようにして、こうしたデータモデルを相互連携させ、ビジネス上の問題への対処や、不明なパターンを導き出すことによるインサイトの生成を可能にするかを計画および整理します。

Start

包括的な能力を定義する

現在のビジネス環境で重要なのは、最新のデータ分析プラットフォームを使用して、データから価値を引き出し、組織内のさまざまなドメインに対応することです。最新のデータアーキテクチャでは、単一のデータアーキテクチャアプローチを採用するのではなく、特定のユースケース専用に構築され最適化されたツールセットやパターンを導入する必要があります。また、このアーキテクチャは、進歩させることができ、スケーラブルなデータレイク、専用の分析サービス、統合データアクセス、統合ガバナンスなどの基本的な機能で構成する必要もあります。

データゾーンを整理する

データアーキテクチャには、データの整理と保存によって迅速かつ簡単にアクセスできるようにする仕組み、という重要な側面があります。これを実現するには、データレイク内にカスタムデータゾーンを設定します。データゾーンは次のように分類します。

- 異種ソースから収集した生データ
- 各ドメインの分析ニーズを満たすために厳選および変換したデータ
- レポートニーズに対応するためのユースケースや、製品ベースのデータマート
- セキュリティとコンプライアンスの管理対象となっている外部公開データ

データの俊敏性確保および民主化を計画する

分析プラットフォームがいかに有効かは、データのプロビジョニング速度に加え、プロビジョニング済みデータを利用可能にするための民主化によって決まります。俊敏なデータプロビジョニングを実現するには、ユースケースに基づいてさまざまな方法でデータを取得および処理する機能をデータアーキテクチャに取り入れます。例えば、リアルタイムまたはほぼリアルタイムな処理、バッチやマイクロバッチによる処理、ハイブリッドな処理などです。また、データの民主化を実現するには、データ共有およびアクセスコントロールのワークフローを定義し、データスチュワードがそれをモニタリングするようにします。データを民主化するイネーブラーの 1 つとして、データマーケットプレイスの実装が挙げられます。

安全なデータ提供を定義する

最新のデータアーキテクチャは、外部からの影響を防ぐセキュリティ上の要塞である一方、従業員やデータユーザーによるアクセスを、職務規定のとおりに容易にする機能も果たしています。また、医療保険の相互運用性と説明責任に関する法律 (HIPAA)、個人を特定できる情報 (PII)、一般データ保護規則 (GDPR) といったコンプライアンス関連規制にも準拠しています。こうした機能を実現するには、ロールベースのアクセスコントロール (RBAC) およびタグベースのアクセスコントロール (TBAC) メソッドを使用し、AWS では、タグを使用してデータへのアクセスを制御し、アクセスコントロールの管理を簡素化します。以上のような機能を、AWS CAF セキュリティの視点で概説されている原則に従って実装します。

費用対効果を得るための計画を立てる

従来のデータウェアハウスでは、コンピューティングとストレージが緊密に結合されているうえ、リソース利用に高いコストが発生しますが、最新のアーキテクチャは、コンピューティングとス

トレージが切り離され、データライフサイクルに基づく階層型ストレージが実装されています。例えば、AWS では、Amazon Simple Storage Service (Amazon S3) を使用してコストを管理でき、データストレージをコンピューティングから切り離すことも可能です。Amazon S3 ストレージクラスは、さまざまなアクセスパターンに対し、最も低コストのストレージを提供するように特別に設計されています。さらに、AWSコンピューティングツール (Amazon Athena、AWS Glue、Amazon Redshift、Amazon SageMaker Runtime など) は、サーバーレスであるため、インフラストラクチャの管理が不要で、課金は使用した分にのみ適用されます。

高度化

最新のデータアーキテクチャは、データ活用の幅を広げることで強化できます。その用途は、ビジネスや運用上の能力をサポートする標準的な分析から、予測やインサイトに対応した複雑な機能に至るまで多岐にわたるでしょう。また、こうしたアーキテクチャは、迅速な意思決定にも有用です。これを実現するために、このアーキテクチャでは、以下のセクションで説明する能力がサポートされています。

特徴量エンジニアリングを理解する

特徴量エンジニアリングでは、機械学習を使用し、特徴量ストアや特徴量マートをセットアップします。また、データサイエンスチームが、教師あり学習モデルと教師なし学習モデルの両方に特徴量 (派生属性) を新規作成して、特徴量マートに保存します。これにより、変換を簡素化し、データ精度を高めます。複数の分析モデルで機能を再利用できるため、市場投入までの時間が短縮されます。

データセット非正規化の計画を立てる

非正規化データセットやデータマートを構築すると、必要なデータが 1 か所で簡単に利用できるようになるため、ビジネスユーザー向けデータセットが大幅に簡素化され、分析速度も向上します。慎重に設計していれば、1 つのレコードで複数の使用モデルに対応でき、開発ライフサイクル全体が短縮されます。非正規化データセットの効果的なガバナンスも、次の 2 つの理由で重要となります。1 つは、非正規化データを実装すると、冗長なデータセットが大量に作成され、大規模な管理が困難になりかねないことです。もう 1 つは、こうしたデータセットを適切にモデル化していない場合、再利用がますます難しくなる可能性があることです。

移植性とスケーラビリティを設計する

大規模な組織の場合、すべてのアプリケーションとユーザーが 1 つのデータプラットフォームに配置されることはほとんどありません。一般的に、アプリケーションとデータストアは従来のオンプレミスプラットフォームとクラウドプラットフォームに分散されるため、分析チームでデータを混在さ

せマージすることは困難です。そのため、ドメイン、地域、ビジネスユースケースなどの特性に基づいてデータをコンテナ化することをお勧めします。こうしたコンテナ化により、さまざまなプラットフォームとアプリケーション間の移植性が向上するうえ、データの利用も効果的に進みます。また、データをコンテナにセグメント化し、API を介して公開することで、データアーキテクチャのスケーリングがさらに容易になります。これによって、エンドツーエンドのハイブリッドデータフローが実現し、オンプレミスおよびクラウドベースのアプリケーションが、よりシームレスに稼働するようになります。

Excel

最新の分析アーキテクチャを組織内で発展させていく中で重要なのは、再利用可能な概念を導入し、そうした変更を管理することです。そのような概念があれば、持続性と採用率が向上し、コストも抑えられます。以下のセクションでは、どのような概念を考慮すべきかについて説明します。

設定可能なフレームワークを設計する

組織では、多くの場合、独自のビジネスニーズに対応するために、複雑なモデルが複数作成されます。こうしたモデルでは、複数のデータパイプラインや、エンジニアリングした特徴量を作成する必要があるため、時間の経過とともに冗長性が大幅に高まり、運用コストも増加します。設定可能でパラメータ駆動型のベースモデルセットを組み込んだフレームワークを作成すると、開発時間と運用コストを削減でき、分析エンジンに、こうした設定可能なモデルを実装することで、必要な結果を得られます。

統合分析エンジンの構築計画を立てる

ビジネス上の問題はそれぞれが異なり、多くの場合、技術のカスタマイズによって要件を満たす必要があるため、組織内に複数の分析エンジンが存在することになります。複数のプログラミングパラダイムに対応した、AI ベースの統合分析エンジンインターフェイスを設計および開発すると、使用が簡素化され、コストも減少します。

DataOps を定義する

データプロフェッショナルのほとんどが、適切なデータの特定、変換、モデリングといったデータオペレーションの実行に、かなりの時間をかけています。アジャイルなデータオペレーション (DataOps) を導入すると、データエンジニア、データサイエンティスト、データ所有者、アナリストのサイロ化が解消されるため、データアーキテクチャを大幅に強化できます。DataOps は、チーム間のコミュニケーション向上、サイクル時間の短縮、高いデータ品質の確保を可能にします。ビジネスニーズが変化し技術が進歩する中で、データおよび分析アーキテクチャには、時間の経過とともに

さまざまな変化が生じています。組織も、ビジネスをサポートできるよう、データおよび分析アーキテクチャの開発、実装、維持に努め、それらを徐々に発展させる必要があります。

データエンジニアリング

組織全体のデータフローを自動化およびオーケストレーションします。

メタデータを使用することで、生データを処理し最適化された分析結果を生成するパイプラインを自動化します。AWS CAF が示す、プラットフォームアーキテクチャおよびプラットフォームエンジニアリングの能力と、オペレーションの視点に定義されているように、既存のアーキテクチャガードレールとセキュリティコントロールを活用します。また、プラットフォームエンジニアリングのイネーブルメントチームと協力して、再利用可能なブループリントを一般的なパターン用に作成し、これによってパイプラインのデプロイを簡素化します。

Start

データレイクをデプロイする

構造化および非構造化データに適したストレージソリューションを使用して、基本的なデータストレージ機能を確立します。これによって、さまざまなソースからデータ収集して保存し、詳細な処理や分析のためにそれらを利用可能にすることができます。データストレージは、データエンジニアリング戦略で重要な役割を担います。データストレージアーキテクチャを適切に設計すると、データを効率的かつ費用対効果の高い方法で、保存および管理し、それらにアクセスできます。AWS では、特定のビジネスニーズを満たすさまざまなデータストレージサービスが提供されています。

例えば、オブジェクトストレージに Amazon Simple Storage Service (Amazon S3)、リレーショナルデータベースに Amazon Relational Database Service (Amazon RDS)、データウェアハウスに Amazon Redshift をそれぞれ使用して、基本的なデータストレージ機能を確立できます。こうしたサービスでは、データを安全かつ費用対効果の高い方法で保存し、詳細な処理や分析のためにそれらを利用可能にすることができます。データのパーティショニングおよび圧縮といった、データストレージのベストプラクティスを実装して、パフォーマンスを向上させることもお勧めします。

データインジェストのパターンを開発する

データフローを自動化およびオーケストレーションするには、データインジェストプロセスを確立して、データベース、ファイル、API といった各種ソースからデータを収集します。データインジェストプロセスでは、ビジネスの俊敏性に対応し、ガバナンスコントロールを考慮する必要があります。

オーケストレーターは、クラウドベースのサービスを実行するとともに、自動スケジューリングの仕組みを実現する機能を持つ必要があります。タスク間の条件付きリンクと依存関係のオプションに

加え、ポーリング機能とエラー処理機能も備えていなければなりません。さらに、パイプラインをスムーズに実行できるよう、アラートおよびモニタリングシステムとシームレスに統合されている必要があります。

一般的なオーケストレーションの仕組みを次に示します。

- 時間ベースのオーケストレーション: 再帰的な間隔と定義済みの頻度でワークフローを開始します。
- イベントベースのオーケストレーション: ファイルの作成や API リクエストといったイベントの発生に基づいて、ワークフローを開始します。
- ポーリング: タスクまたはワークフローで、(API などを介して) サービスを呼び出し、定義済みのレスポンスを待ってから次のステップに進む仕組みを実装します。

最新のアーキテクチャ設計は、マネージドサービスを活用して、クラウドでのインフラストラクチャ管理の簡素化や、開発者チームやインフラストラクチャチームの負担軽減を行うことに重点が置かれています。このアプローチは、データエンジニアリングにも適用されるため、必要に応じてマネージドサービスを利用してデータインジェストパイプラインを構築し、データエンジニアリングプロセスを迅速化すると良いでしょう。こうしたタイプのサービスの例には、Amazon Managed Workflows for Apache Airflow (Amazon MWAA) と AWS Step Functions の 2 つがあります。

- Apache Airflow は、プログラムによるワークフロー作成、スケジューリング、モニタリングなどに広く利用されているオーケストレーションツールです。AWS では、Amazon Managed Workflows for Apache Airflow (Amazon MWAA) というマネージドサービスが提供されており、これによって、オーケストレーションツールのインフラストラクチャ管理ではなく、構築作業に集中できるようになります。Amazon MWAA では、Python スクリプトを使用してワークフローを簡単に作成することも可能です。有向非巡回グラフ (DAG) は、各タスクの関係と依存関係を示す方法で、ワークフローをタスクのコレクションとして表すものです。必要な数の DAG を持つことができ、Apache Airflow では、各タスクの関係性および依存関係に従ってそれらを実行します。
- AWS Step Functions を使用すると、IT およびビジネスプロセスを自動化する、ローコードのビジュアルワークフローを構築しやすくなります。Step Functions で構築するワークフローをステートマシンと呼び、ワークフローの各ステップをステートと呼びます。Step Functions を使用すると、組み込みのエラー処理、パラメータの受け渡し、推奨のセキュリティ設定、ステート管理を行うワークフローを作成でき、これによって、書き込みと保守が必要なコードの量も減少します。タスクの処理は、別の AWS サービス、もしくはオンプレミスまたはクラウド環境でホストしているアプリケーションと連携させて実行します。

データ処理を高速化する

データ処理は、先進的な組織で収集されている膨大な量のデータを理解する上で重要なステップであり、データ処理に着手できるよう、AWS では、強力な抽出、変換、ロード (ETL) 機能を得られる AWS Glue などのマネージドサービスが提供されています。こうしたサービスを利用すると、データのクリーニング、正規化、集計といった、生データの処理および変換に着手して、それらを分析に使用する準備を整えられます。

データを処理するには、初期データの変換を実行する集約やフィルタリングなどのシンプルな手法を最初に実践します。データ処理のニーズが変化していく中で、より高度な ETL プロセスを実装でき、これにより、さまざまなソースからデータを抽出して、特定のニーズに合うよう変換し、それらを一元化したデータウェアハウスやデータベースにロードして統合分析を行えます。こうしたアプローチを取ることで、データの精度と完全性を高め、適切なタイミングでそれらを分析に利用できるようになります。

AWS のマネージドサービスをデータ処理に利用すると、自動化、スケーラビリティ、コスト効率のレベルが高まり、そのメリットを得ることができます。こうしたサービスにより、スキーマ検出、データプロファイリング、データ変換といった日常的なデータ処理タスクの多くが自動化されるため、解放された貴重なリソースを、より戦略的なアクティビティに投入できます。さらに、これらのサービスを自動スケーリングして、今後のデータボリューム増加に対応することも可能です。

データ可視化のサービスを提供する

どうすれば、意思決定者がデータを利用しやすくなるか、また、それらを可視化し、有意義かつ迅速に解釈できるようになるかを特定します。可視化によってデータのパターンを解釈できるようになるため、さまざまな利害関係者のエンゲージメントがその技術的スキルに関係なく高まります。優れたプラットフォームがあるデータエンジニアリングチームなら、迅速かつわずかなオーバーヘッドでデータを可視化するリソースをプロビジョニングできます。セルフサービス機能を提供することも可能で、このサービスでは、エンジニアリングの専門知識がなくてもデータストアを簡単にクエリできるツールを使用します。これには、可視化したデータやインタラクティブなダッシュボードからサーバーレスビジネスインテリジェンスを利用したり、バックエンドデータを自然言語でクエリしたりできる組み込みツールの使用を検討してください。

高度化

ほぼリアルタイムのデータ処理を実装する

データ処理は、あらゆるデータエンジニアリングパイプラインに不可欠なコンポーネントであり、これによって、生データを有意義なインサイトに変換できます。スピードが求められる今日のビジネス環境では、従来のバッチ処理に加えて、リアルタイムのデータ処理がますます重要になっています。リアルタイムのデータ処理を実装していれば、イベントの発生時にすぐに対応でき、意思決定と運用効率を向上させることも可能です。

データ品質を検証する

データ品質は、データから導き出すインサイトや意思決定の精度と信頼性に直接影響します。高品質で信頼できるデータを分析に使用できるようにするには、データ検証およびクレンジングプロセスの実装が不可欠です。

データ検証では、事前に定義されているルールと基準に照らして、データの精度、完全性、一貫性を検証する必要があります。これにより、データ内の不一致や工ラーを特定しやすくなり、目的に沿うデータを用意できます。データクレンジングでは、データの不正確性、不整合、重複を特定し、修正しなければなりません。

データ品質関連のプロセスおよびツールを実装すると、データから派生したインサイトの精度と信頼性が高まり、ひいては、意思決定と運用効率が向上します。これにより、組織のパフォーマンスが強化されるだけでなく、生成したデータと分析結果に対するステークホルダーの確信と信頼も高まります。

データ変換サービスの有効性を証明する

データ変換では、高度な分析モデルと機械学習モデルにデータを準備します。また、データの正規化、エンリッチメント、重複排除などの手法を使用して、データがクリーンで一貫性があり、分析の準備が整っていることを確認する必要があります。

- データの正規化: データを標準形式に変換して整理し、冗長性をなくし、さまざまなソース間でデータの一貫性を確保します。これにより、複数のソースから得たデータの分析と比較が容易になるため、オペレーションをより包括的に把握できます。
- データエンリッチメント: 人口統計データや市場動向といった外部ソースからの情報を追加して、既存のデータを強化します。これにより、内部データソースだけでは明らかにならないと思われる顧客行動や業界動向について、貴重なインサイトを得られます。

- 重複排除: 重複するデータエントリを特定して削除するとともに、データの精度を高め、エラーをなくす必要があります。これは、わずかな重複によっても分析結果に歪みが生じかねない大規模データセットを処理する場合に、特に重要です。

高度なデータ変換手法を使用すると、データの品質と精度を高め、より複雑な分析にデータをすぐに提供できます。そうすることで、より適切な意思決定、運用効率の向上、市場での競争優位の獲得などが可能になります。

データ民主化を実現する

データ民主化の文化を促進するには、すべての従業員がデータにアクセスでき、それらを理解し使用できるようにします。データ民主化が実現すると、従業員がデータ駆動型の意思決定を行ったり、組織のデータ駆動型文化に貢献したりするようになります。つまり、サイロ化が解消し、意思決定を推進するためにすべての従業員がデータを共有し使用する文化が作られます。

まとめると、データ民主化で重要なのは、組織内の誰もがデータの評価、アクセス、理解を行える文化を作ることです。データ民主化の実現によって、データ駆動型の文化を形成すると、イノベーションの推進や、意思決定の改善が可能になり、最終的には、これらをビジネス成果につなげられます。

Excel

UI ベースのオーケストレーション機能を利用可能にする

効果的なアプローチを取る俊敏な組織を作る場合に重要なのは、最新のオーケストレーションプラットフォームを計画し、事業部門全体の開発および運用の人的リソースが使用できるようにすることです。ここでの目標は、単一のチーム、テクノロジー、サポートモデルに依存せずに、データパイプラインとワークフローを開発、デプロイ、共有することです。これを実現するには、UI ベースのオーケストレーションなどの機能を使用します。ドラッグアンドドロップ操作などの機能により、技術的な専門知識がほとんどないユーザーでも、DAG やステートマシンのデータフローを構築できます。その後、こうした UI ベースのコンポーネントで、データパイプラインをオーケストレーションする実行可能コードを生成できます。

DataOps を導入すると、データ管理の複雑さを克服し、シームレスなデータフローを組織全体で確立しやすくなります。また、メタデータ駆動型のアプローチによって、品質とコンプライアンスが確保され、組織の要件や規則に従うことができ、マイクロサービス、コンテナ化、サーバーレスの機能といったツールセットに投資すると、スケーラビリティと俊敏性が向上します。

データからの価値創出をデータエンジニアリングチームに任せ、日々のインフラストラクチャタスクを自動化することで、自動化とオーケストレーションの優秀性を実現できます。また、データフロー管理タスクをほぼリアルタイムでモニタリングしログ記録すると、修復アクションをすぐに実行できるようになり、データフローパイプラインのパフォーマンスとセキュリティも向上します。こうした原則は、スケーラビリティとパフォーマンスの目標達成に加え、安全なデータ共有モデルの確立や、将来の成功を見据えた組織の構築に有用です。

DataOps を統合する

DataOps は、データエンジニアリングへの最新のアプローチであり、この手法では、データパイプラインの作成、テスト、デプロイを合理化する開発プロセスと運用プロセスの統合が重視されています。DataOps のベストプラクティスを実装するには、Infrastructure as Code (IaC) と継続的インテグレーションおよび継続的デリバリー (CI/CD) のツールを使用します。こうしたツールでは、パイプラインの自動作成、テスト、デプロイがサポートされているため、効率が大幅に向上し、エラーも減少します。DataOps チームは、プラットフォームエンジニアリングのイネーブルメントチームと協力して、こうした自動化を構築します。これにより、それぞれのチームが、最善を尽くすべきタスクに集中できるようになります。

DataOps メソッドを実装すると、データエンジニア、データサイエンティスト、ビジネスユーザーのためのコラボレーション環境作りが進み、データパイプラインと分析ソリューションの迅速な開発、デプロイ、モニタリングが可能になります。このアプローチによって、チーム間でよりシームレスなコミュニケーションとコラボレーションを行えるようになり、イノベーションの迅速化と成果の向上につながります。

DataOps の利点を最大活用するには、データエンジニアリングプロセスの合理化が重要です。これを達成するには、コードレビュー、継続的インテグレーション、自動テストといった、プラットフォームエンジニアリングチームのベストプラクティスを実装します。こうしたプラクティスの実装により、データパイプラインの信頼性、スケーラビリティ、安全性を確保するとともに、ビジネスおよび技術の両ステークホルダーのニーズを満たすことができます。

プロビジョニングとオーケストレーション

承認済みクラウド製品のカタログを作成、管理し、ユーザーに配布します。

組織の成長に伴い、一貫性があり、スケーラブルで、繰り返し可能な方法による、インフラストラクチャのプロビジョニングが、以前よりも困難になります。プロビジョニングとオーケストレーションを合理化すると、一貫したガバナンスや、コンプライアンス要件への準拠が可能になるとともに、ユーザーが承認済みクラウド製品のみをデプロイできるようになります。

組織内で事前承認した製品を再利用すると、一貫性のある短期間でのアプリケーション構築が可能になると同時に、組織のセキュリティおよびガバナンス要件にも準拠できます。

Start

ハブアンドスポークカタログモデルをデプロイする

ポートフォリオとしてサービスカタログで管理するソフトウェアアセットは、ハブアンドスポークパターンで 1 つ以上のアカウントのユーザーと共有します。プライベートマーケットプレイスとプライベートオファーを使用すると、一連のサードパーティーソリューションをキュレートし、Infrastructure as Code (IaC) テンプレートで配布できます。

事前承認された製品をビルダーが消費できるようにするには、こうした製品にレビュー、承認、ユーザーへの公開を行うプロセスを定義します。最初に、こうした事前承認済み製品を置く一元管理型リポジトリを設計して、実装し、組織のユーザーが各製品を消費する必要がある場合は、このリポジトリ内のライセンスと製品へのアクセスを許可するシステムを設計します。

組織内のビルダーが、製品を公開システムに送信し、承認を依頼できるようにすると、製品の承認後、組織内のすべてのユーザーにそれらが利用可能となります。

再利用を目的にテンプレートをキュレートする

ソリューションの IaC テンプレートをコード化し、ハブアンドスポークモデルを定義したら、各スポークアカウントに 2 種類のテンプレートカテゴリを定義しなければなりません。1 つはプロビジョニング/強制、もう 1 つは使用可能です。プロビジョニング/強制テンプレートは、基本機能として管理アカウントから各メンバーアカウントに直接プロビジョニングします。使用可能テンプレートは、ビルダーがセルフサービス方式で参照およびプロビジョニングできるように用意するものです。

再利用を目的にデフォルトパラメータを適用する

デフォルトのパラメータが含まれる IaC テンプレートを実装し、それらをビルダーが事前に選択できるようにします。これにより、各パラメータの詳細を評価しなくても、ガバナンスに従うことができ、誤った選択を防ぐことも可能です。このアプローチでは、セットアップに必要な設定項目のみを公開します。例えば、AWS Service Catalog によって、このアプローチを実装しますが、その際には、特定のポートフォリオ製品に適用するルールを制御する制約機能も取り入れます。ビルダーチームがテンプレートのセルフサービスプロビジョニングを使用する場合は、こうしたカスタマイズを事前に設定します。

承認プロセスを確立する

その製品を使用するビジネス上の理由がある場合、ユーザーが未承認製品へのアクセスリクエストを送信できるようにする必要があります。また、使用している製品の更新版が利用可能になったらユーザーに知らせる通知システムを構築して、最新のセキュリティ更新プログラムにユーザーが準拠できるようにします。

ビルダーがセルフサービスポータルからレビュー用の新製品を送信するためのワークフローを確立します。また、ビルダーが、このポータルを使用して製品の対象者を定義し、製品へのアクセスが必要なユーザーグループを特定できるようにするとともに、ビルダーからの送信ごとに、定義したプロセスを使用して製品をレビュー、承認し、セルフサービスポータルに公開します。

高度化

セルフサービスポータルを作成する

承認済みクラウド製品の配布、参照、消費を行えるセルフサービスポータルを作成し、組織内のユーザーが、このポータルを使用して、インフラストラクチャ構築に必要な製品の検索や、自身の環境へのアプリケーションデプロイを行えるようにします。ポータル上の製品に対するユーザーへのアクセス許可と、ユーザーがライセンス製品を使用できる回数に制限を設けるとともに、直接プロビジョニングできる、あるいはセルフサービスモデルとして利用可能な基本のリソースセットを、各スポークアカウント内で定義します。これを定義するのは、AWS Control Towerのカスタマイズといったソリューションによって、こうしたアカウントを作成するからです。

プライベートマーケットプレイスを利用可能にする

プライベートマーケットプレイスとは、購入済み製品 (ソフトウェア、データ、プロフェッショナルサービス) のカタログがキュレートされている場所で、ハブアンドスポークパターン (1 つの管理

アカウントと複数のメンバーアカウント) で実装するものです。これによって、スポークアカウントが、承認済みソフトウェアのみをサブスクライブできるようにします。こうした製品ガバナンスは、ソフトウェアのコスト管理や、法的および契約上のレビューの合理化に有用です。プライマリハブとして機能するよう、管理アカウントレベルでプライベートマーケットプレイスを作成します。

エンタイトルメントを管理する

ベンダー定義の制限内でライセンスを使用できるようにするコントロールを有効にして、承認されたユーザーとワークロードのみに使用を許可します。これにより、コストのかかる監査や予期しないライセンス調整のリスクが軽減されます。

Excel

調達システムと統合する

既存の調達プロセスを AWS Marketplace に統合し、補完します。これを行うには、調達システム (Coupa または SAP Ariba) をプライベートマーケットプレイスに拡張して、ユーザーが既存の調達および承認プロセスに従ってソフトウェアを取得できるようにします。適切な IAM 管理のアクセス許可を作成し、 AWS Marketplace を使用して調達ソリューションを設定するために必要な情報を生成し、統合を完了するように調達ソリューションを設定します。例えば、パンチアウトを設定し、AWS 請求書に発注書をアタッチし、調達プロセスを調整して標準プロビジョニングソリューションを使用できます。

ビルダーが内部 API を介して事前承認された製品にアクセスできるようにします。これにより、ユーザー側で、製品をアプリケーションに組み込んだり、パーソナライズした独自のチームポータルを構築して製品を消費したりできます。また、新しい製品を作成するための送信と公開のプロセスを統合し、ユーザーが API 経由で、新しいライセンスと製品へのアクセスをリクエストできるようにします。

ITSM ツールと統合する

必要に応じて、IT サービス管理 (ITSM) ツールと連携させ、構成管理データベース (CMDB) の更新を自動化します。組織で使用する製品を評価するプロセスと仕組みに加え、事前承認された製品を使用中のユーザーに、コンプライアンス上、製品の更新が必要であることを通知する仕組みを確立します。また、ITSM ツールを使用して、環境を分析し、重要な更新が必要な場合に組織全体の製品にセキュリティやコンプライアンス上の更新をプッシュします。

ライフサイクル管理とバージョン配布システムを実装する

IaC テンプレートのバージョンと、そのテンプレートからプロビジョニングしたサービスのバージョンを、開発ライフサイクル全体を通して管理します。カタログに実装したハブアンドスポークを使用すると、スポークレベルで強制更新が必要かどうか (例: セルフサービスプロビジョニングで特定のバージョンが同時に利用可能な場合) や、どのバージョンを廃止対象としてマークする必要があるかを定義でき、必要に応じて、新しいバージョンの監査と配布を管理することも可能です。

最新のアプリケーションの開発

適切に設計したクラウドネイティブアプリケーションを構築します。

クラウドネイティブなアプリケーションを組織で適切に設計、構築し、競争力を維持するには、モダンアプリケーションの開発プラクティスがきわめて重要です。コンテナやサーバーレスコンピューティングといったクラウドネイティブな技術を使用すると、スケーラブルでアジャイルなアプリケーションを構築し、変化し続ける市場需要に適応できます。こうした技術により、リソース使用率の最適化、コスト削減、アプリケーションパフォーマンスの向上などが可能になります。

モダンアプリケーションの設計時には、運用と開発に配慮したアジャイルソリューションを開発します。モダンアプリケーションを実現するには、顧客の需要の変化に自動的に反応し、障害から回復する機能を持たせるとともに、変更の迅速な開発およびデプロイや、アプリケーションパフォーマンスのモニタリングを行えるようにします。また、自己修復に加え、トラフィックが少量でも大量でもスケーリングできる機能を取り入れ、必要であれば、トラフィックがない場合はコストをまったく生じさせない機能も追加します。

クラウドネイティブアプリケーションを適切に設計し、構築するには、基盤技術とそのベストプラクティスを深く理解する必要があります。また、マイクロサービスアーキテクチャを導入して、モジュール式かつ疎結合となるようアプリケーションを設計することで、独立したデプロイとスケーラビリティを実現しなければなりません。こうしたアプローチを取ると、アプリケーションを小規模で管理しやすいコンポーネントに分割し、これらを迅速かつ個別に開発、テスト、デプロイできます。

Start

最新のアプローチを調査し考察する

最初に、マイクロサービスの開発を可能にするコンテナやサーバーレス技術などのアプローチを調査します。マイクロサービスは、リソースの効率向上、セキュリティの強化、インフラストラクチャコストの最小化などを可能にします。また、既存の差別化アプリケーションとエンタープライズアプリケーションをモダナイズすることで、効率を高め、既存投資の価値を最大化できるようにします。さらに、リプラットフォーム (セルフマネージドのコンテナ、データベース、メッセージブローカーをマネージドクラウドサービスに移行する) とリファクタリング (アプリケーションを再構築してクラウドネイティブアーキテクチャを導入する) を、価値主導の意思決定に基づいて考察します。

既存のクラウドベースのアプリケーションを効果的なアプローチで更新するには、Strangler Fig パターンを使用して、アーキテクチャを徐々にマイクロサービスに分解する必要があります。この手順は、現代的なアプリケーション方法論の導入に役立つため、そうした技術固有の利点を享受し、

組織全体にその価値を示すことができます。必要であれば、イベント駆動型アーキテクチャを活用する個別のマイクロサービスとしてアプリケーションを構築することを検討してください。また、ワークロードのパフォーマンスや信頼性に影響を与えないよう、変更不可能なサービスクォータや物理リソースがアーキテクチャで考慮されていることを確認します。https://docs.aws.amazon.com/wellarchitected/latest/reliability-pillar/manage-service-quotas-and-constraints.html

クラウドネイティブコンピューティング機能を導入する

クラウドネイティブコンピューティング機能は、モダンアプリケーション開発のカギとなるものです。このアプローチを取るには、コンピューティングユニットのホスト方法を検討し、各ユースケースやサービスに最適な選択肢を特定する必要があります。例えば、AWS Lambda は、アプリケーションコードを実行するためのサーバーレスメカニズムを備え、イベント駆動型アーキテクチャで重要な役割を果たします。Lambda 関数は、オンデマンドで起動して、定義した最大同時実行数まで並行して実行できるため、スケールしてさまざまなタスクを実行することが可能です。

コンテナ化を使用する

最新のソフトウェア開発では、アプリケーションとその依存関係の管理がますます複雑になっており、こうした状況は、さまざまな環境においてどの程度の一貫性が必要かを検討しているときに顕著に表れます。このような課題への対処として、Docker などのコンテナ化技術が、アプリケーションとその依存関係をパッケージ化する効果的なソリューションとして新たに注目されています。コンテナを使用すると、アプリケーションのランタイム環境に関係なく、デプロイの一貫性と再現性が確保されるため、ローカル環境での開発をクラウド環境の本番開発と同じ方法で行えます。このアプローチなら、環境内やその設定の不一致によって生じかねないエラーも軽減できます。

最新のデータベースを使用する

最新のデータベースを使用すると、アプリケーション内の各マイクロサービスで要件に沿う適切な目的別データベースを使用できるため、俊敏性とパフォーマンスが向上し、コストも減少します。例えば、あるマイクロサービスで、セッションデータの保存時に NoSQL データベースを使用して高スループットを実現し、別のマイクロサービスで、リレーショナルデータベースを使用して複雑なテーブル結合を実行し、さらに別のマイクロサービスで、量子台帳データベースを使用してブロックチェーンの変更を追跡するなどが考えられるでしょう。

最新のデータベースは、スケーラビリティと柔軟性を備えており、従来のデータベースよりも、セキュリティ、コンプライアンス、信頼性に優れています。これにより、データの保存および管理が以前よりも効率化され、アプリケーションが適切なデータに適切なタイミングでアクセスでき、パフォーマンスとユーザーエクスペリエンスも向上します。

モダンアプリケーション開発では、最新のデータベースへの移行がきわめて重要です。適切なデータストレージソリューションを使用すると、データ管理機能を最適化して、効率と信頼性に優れたアプリケーションを提供できます。また、各マイクロサービスを独立させ、それらに適した技術を選択すると、データ機能がさらに最適化されるため、最大限の効率およびスケーラビリティと、最小コストを実現できます。

高度化

最新のアーキテクチャを最適化する

さらなる最適化を実現するには、サーバーレス技術の実装を高度化し、Amazon API Gateway や AWS Lambda などの AWS サービスを利用して、個別にスケーリングおよびデプロイ可能なアーキテクチャを開発します。また、Amazon Route 53 と AWS Cloud Map を使用してサービス検出の機能を実装し、コンポーネント間でシームレスな通信を確保します。

API バージョニング、キャッシュ、レート制限の導入によって、さまざまなアプリケーションバージョン間で互換性とパフォーマンスを維持するとともに、AWS Identity and Access Management (IAM) とリソースポリシーを使用して、セキュリティを強化します。これにより、インフラストラクチャを保護し、承認済みエンティティにのみアクセスを許可します。

可能であれば、サーバーレスサービスを利用して、基盤インフラストラクチャを管理することなくコンテナを実行します。これにより、コアアプリケーション開発に集中すると同時に、リソース管理とパフォーマンスを向上させることができ、スケーラビリティ、柔軟性、コスト効率の利点も最大活用しやすくなります。

サーバーレスアーキテクチャの複雑さを詳細に把握した上で、こうした高度なプラクティスを取り入れると、改善とファインチューニングの機会を発見でき、ひいては、クラウドネイティブアプリケーションの可能性が最大化します。そのように取り組むことで、より高度なアプリケーションパターンを導入しやすくなるため、全体的なユーザーエクスペリエンスがさらに向上し、ソフトウェア開発プロセスの俊敏性と効率も高まります。

サービスメッシュ技術を使用する

アプリケーションの構築とデプロイにマイクロサービスアーキテクチャの導入が増えると、こうしたサービス間の複雑さ、セキュリティ、通信を管理することが重要になります。そこで、マイクロサービスのセキュリティ、オブザーバビリティ、信頼性を確保するために重要な役割を果たすのが、Istio、Linkerd、Consul といったサービスメッシュ技術です。

可視性とトレーサビリティを確保する

最新のプラクティスを実装すると、開発プロセスの可視性とトレーサビリティが向上するとともに、業界標準とベストプラクティスに容易に準拠できます。また、モダンアプリケーション開発には、可視性とモニタリングが不可欠です。モニタリングおよびログ記録ソリューションを実装し、アプリケーションパフォーマンスに関する貴重なインサイトを提供することで、改善すべき分野を特定し、アプリケーションを最適化できます。さらに、プラットフォームエンジニアリングチームと協力して、アプリケーションエラー、パフォーマンス、コンプライアンスをエンドツーエンドで可視化およびモニタリング可能なツールを利用できるようにすると良いでしょう。これが、問題の迅速な検出、診断、解決につながります。

Excel

マイクロサービスを採用する

多くの組織にとって、モダンアプリケーション開発の実現は、ビジネスの成功を意味します。こうした変革の中心となるのは、マイクロサービスであり、こうした強力なアーキテクチャパターンの採用は組織にメリットをもたらします。

マイクロサービスを導入すると、スケーラビリティ、耐障害性、俊敏性に優れたアプリケーションアーキテクチャが実現します。アプリケーションを独立してデプロイできる小規模サービスに分割することで、アプリケーションの他の部分に影響を与えずに、特定のコンポーネントを迅速にイテレーションすることもできます。サーキットブレーカーやバルクヘッドなどの高度な耐障害性パターンは、こうしたアプリケーションの高可用性を確保する上で重要な役割を果たします

サーキットブレーカーは、安全上の仕組みで、これにより、異常なサービスからの通信を一時的に停止または変更して連鎖的な障害を防ぎ、復旧を行えます。バルクヘッドを使用すると、リソースを分離し、起こり得る障害の影響範囲を狭められます。こうしたパターンの組み合わせにより、予期しない障害や問題に耐え、最適なパフォーマンスが維持される堅牢なアーキテクチャを構築できます。

マイクロサービスの実装で、もう 1 つ重要なのは、ドメイン駆動型設計 (DDD) の原則の導入です。DDD では、ビジネスドメインの理解を共有し、適切に構造化したソフトウェア設計に反映させることに重点が置かれています。このアプローチによって、よりまとまりのある保守可能なマイクロサービスが実現し、組織のニーズに合わせたアプリケーションの進歩が可能になります。

サービス間通信の最適化は、マイクロサービスベースのアプリケーションでもきわめて重要です。gRPC や GraphQL などの高度なプロトコルを実装すると、サービス間の通信効率が大幅に向上します。また、こうしたプロトコルにより、型の安全性、低レイテンシー、柔軟性などを実現する機

能を得られるため、アプリケーションの全体的なパフォーマンスおよび保守性を改善しやすくなります。

マイクロサービスを導入した組織では、イノベーション、俊敏性、コラボレーションを促進する環境が実現します。ビジネス能力を中心に開発チームが編成されることが多く、こうしたチームは、継続的インテグレーションと継続的デリバリー (CI/CD) のプラクティスに重点を置きます。意思決定、実験、迅速なイテレーションを行う権限を持ち、責務と説明責任を共有する文化を受け入れています。

継続的インテグレーションと継続的デリバリー

従来のソフトウェア開発およびインフラストラクチャ管理プロセスを使用している組織よりも、迅速にアプリケーションとサービスを進歩させ改善します。

DevOps プラクティスを継続的インテグレーションおよび継続的デリバリー (CI/CD) とともに導入すると、アプリケーションの構築、テスト、デプロイのプロセスを合理化、自動化、効率化できます。CI/CD により、ソフトウェアデリバリーの迅速化や、デプロイエラーリスクの軽減を行えるほか、最新の機能やバグ修正を加えることで、アプリケーションを常に最新状態に維持できます。主な目的は、従来のソフトウェア開発およびインフラストラクチャ管理プロセスの手順を発展させ、アプリケーションとサービスの高度化と改善を短期間で行えるようにすることです。

Start

ソフトウェアコンポーネント管理を導入する

ソフトウェアコンポーネント管理とは、コンポーネント管理のプラクティスであり、このプラクティスでは、ライブラリ、フレームワーク、ソースコードリポジトリ、モジュール、アーティファクト、サードパーティー製要素との依存関係といった、ソフトウェア構築に使用する個々のコンポーネントをすべて管理します。これを実践する場合は、Git や Apache Subversion などのバージョン管理システムを使用して、ソースコードの管理、コラボレーション環境の実現、コード変更履歴の維持などを行うと良いでしょう。リポジトリ内での変更およびイベントをモニタリングすると、プロセス自動化、パイプライン作成、コード管理を行え、必要であれば、ワークフローを追加のサービスと統合できます。

CI/CD パイプラインを作成する

CI/CD パイプラインとは、自動化された一連の手順であり、バージョン管理システムに変更をコミットすることで、これらを開始します。パイプラインは、通常、アプリケーションの構築、自動テストの実行、特定の環境にコードをデプロイする手順などで構成されます。自動 CI/CD パイプラインは、AWS CodePipeline、Jenkins、GitLab、CircleCI などのツールを使用してセットアップできます。また、パイプライン生成をサポートするバージョン管理システム内に、直接セットアップすることも可能です。

最初に、継続的インテグレーションを目的とした最小限の実行可能なパイプラインを作成し、さらに多くのアクションとステージからなる継続的デリバリーパイプラインに移行します。継続的デリバリーの設定は、コードとして扱います。ブランチやチームごとに個別のパイプラインを複数使用でき

るため、セットアップが必要な設定変数や、パイプラインを使用するチームに最適なサポート方法を検討してください。

次に、デプロイウィンドウ (コードをデプロイする日時) を検討します。システム需要が低い時間帯を対象として考えると、ロールバックの必要がある場合に、顧客への影響を最小化できるでしょう。その他のベストプラクティスには、金曜日のデプロイを避けるや、ピーク時または休日前にコードフリーズを行うなどがあります。作成者がコミットできない場合 (休暇中など) のコードデプロイのルールを定義することも検討してください。デプロイが失敗し、外部のサポートに頼らざるを得ない場合もあることに注意します。また、インプレース、ローリング、イミュータブル、ブルー/グリーンデプロイといったさまざまなデプロイ方法を評価するとともに、継続的デリバリーワークフローにフルマネージドサービスを利用することも検討します。このサービスにより、可用性とセキュリティを強化し、複雑さと管理労力を最小限に抑えられます。

自動テストをデプロイする

最新のプラクティスとして、シフトレフト (開発者が認識しやすく IDE が使用されているタイミングや、開発ライフサイクルの早い段階でテストを行うこと) をお勧めします。これにより、問題をリポジトリにコミットしてパイプラインを開始する前に、問題を検出し修正することができます。この方法を取ると、開発者のコーディング中にエラーが検出されるため、開発者との間で、すぐにフィードバックとそれへの対応を繰り返すことができます。シフトレフトのテストでは、パイプラインの実行が必要でないため、コスト削減につながりますが、それが必要な場合は、フィードバックが途切れがちになり、運用コストも増加する可能性があるでしょう。

自動テストは、開発プロセスの早い段階でエラーを検出するものですが、ここでは、ユニットテスト、統合テスト、機能テストも行います。そのため、開発者には、できるだけ早く、ツールを使用して単体テストを作成するよう勧め、中央リポジトリにコードをプッシュする前に、それらを実行してもらうと良いでしょう。さらに、自動化プロセスに、静的コード分析、パフォーマンスベンチマーク、セキュリティアプリケーションテストが含まれていることも確認します。

ドキュメントを作成する

CI/CD パイプラインを実装して開発ワークフローを合理化するだけでなく、明確で包括的なドキュメントも維持し、パイプラインでの有効性、保守性、スケーラビリティを継続的に確保する必要があります。開発チームがパイプラインの設計、コンポーネント、プロセスを明確に理解するできるようにするという点で、CI/CD パイプラインでは、ドキュメントが重要な役割を果たします。ドキュメントの作成時には、最初にパイプラインを概説します。続いて、アーキテクチャおよび設計のトレードオフ、使用するツールと技術、具体的な初期設定および各種設定、セキュリティ対策とアクセスコントロールなどを説明し、トラブルシューティングとメンテナンスの情報も追加します。

Infrastructure as Code を使用する

Terraform、Ansible、AWS CloudFormation のツールでインフラストラクチャを管理し、一貫性と再現性のある環境を実現します。インフラストラクチャをコードとして扱い、インフラストラクチャの変更を追跡し、変更はコンソールから直接行わないようにします。すべてのインフラストラクチャをコードとして定義し、パイプラインを使用してこれらの変更をデプロイします。データベースもコードとして定義し、プロビジョニングします。サニタイズ済み本番データの小規模サブセットを使用して、パイプラインでデータベース統合をコードとして実行することを検討してください。可能であれば、コードを変更し、そうした変更を追跡します。

ソフトウェアのコードと同様、インフラストラクチャのコードでも、次のベストプラクティスに従います。

- バージョンコントロールを使用します。
- バグ追跡およびチケット発行システムを使用します。
- 適用前に、変更を同僚に確認してもらいます。
- インフラストラクチャコードのパターンおよび設計を確立します。
- インフラストラクチャの変更をテストします。

標準メトリクスを維持および追跡します。

高レベルのパフォーマンスを維持するには、主要なメトリクスを開発して追跡し、以下の点から、パイプラインの正常性とビジネスへの影響を把握します。

- ビルド頻度。ビルド数からは、チームの生産性と変更の複雑さに関するインサイトを得られます。
- デプロイ頻度。定期的なデプロイは、健全でアジャイルな開発プロセスを示しています。
- 変更のリードタイム。変更が本番環境に反映される平均時間を測定すると、デプロイプロセスのボトルネックの特定に役立ちます。
- パイプラインの平均所要時間。最初のパイプラインステージから後続の各ステージまでにかかる平均時間の情報は、ワークフローの最適化に役立ちます。
- 本番環境の変更回数。本番環境に反映させる変更の数を追跡すると、本番環境の安定性についてインサイトを得られます。
- ビルド時間 平均ビルド時間は、コードベースまたはインフラストラクチャで起こり得る問題を示している可能性があります。

高度化

設定管理ツールを使用する

設定管理ツールは、ソフトウェアおよびインフラストラクチャのデプロイ、設定、管理を自動化する上で重要な役割を果たします。こうしたツールの体系的なアプローチによって、変更を処理するとともに、さまざまな環境のインフラストラクチャ、ソフトウェア、設定を望ましい状態に維持できます。また、開発者が宣言言語または命令言語を使用して、システムの望ましい状態を定義できるようになり、それらの設定をターゲットシステムに適用するプロセスを自動化することで、一貫性と再現性も確保されます。

設定管理ツールを使用して、ソフトウェアおよびインフラストラクチャのデプロイ、設定、管理を自動化します。AWS Systems ManagerState Manager は、安全でスケーラブルな設定管理サービスで、これにより、マネージドノードや他の AWS リソースを、定義した状態に維持するプロセスを自動化できます。

モニタリングとログ記録を統合する

モニタリングおよびログ記録のソリューションを CD パイプラインに統合すると、開発チームとソフトウェア開発プロセス全体で多くの利点を得られます。これらのソリューションにより、アプリケーションパフォーマンスに関するリアルタイムのインサイト取得、問題の迅速な特定と解決、継続的改善の促進などが可能になるため、アプリケーションライフサイクル全体を通して、信頼性、パフォーマンス、スケーラビリティを維持しやすくなります。モニタリングおよびログ記録ソリューションへの投資は、堅牢で効率的な CD パイプラインを維持する上で重要な側面であり、高品質のソフトウェアを提供するという成果につながります。

マージのケイデンスを決定する

コード変更は、メインライン (トランクまたはメイン) ブランチに少なくとも 1 日に 1 回、理想的には各タスクの後 1 日に複数回コミットまたはマージします。こうしたケイデンスにより、毎日複数のパイプライン呼び出しが発生します。プルベースの分岐ワークフローモデルは、そのようなアプローチに沿ったものです。機能フラグ、ダークローンチや同様の手法を使用すると、顧客が使用する機能をカスタマイズできます。

デプロイ後の動作をキャプチャする

デプロイ後、自動合成テストを使用して本番環境の動作をキャプチャし、結果を継続的デリバリーパイプラインと同期することで、是正措置を迅速に行えるようにします。開発者にとっての最優先事項

は、パイプラインで検出されたエラーをできるだけ早く修正して、コード変更をソースコードリポジトリにコミットし、パイプラインでエラー解決を検証することです。

デプロイ後のベストプラクティスとして、最も重要な主要業績評価指標 (KPI) を観察するとともに、本番環境にエラーがないことを検証すると良いでしょう。また、エラー処理の自動化と、デプロイ後の KPI 評価によって、リリースの影響を定量化することと、開発者が改善に利用できる速度、セキュリティ、安定性のメトリクスを自動的に生成することも推奨されます。詳細については、AWS の「DevOps Monitoring Dashboard」を参照してください。

Excel

最先端のプラクティスと技術を導入して、最適なパフォーマンスを実現します。CI/CD プロセスを継続的に改善することで、ソフトウェアの品質改善、市場投入の迅速化、俊敏性の向上が可能になります。新しい手法やツールは継続的に出現するため、最新情報を常に把握してそれらに適応し、競争上の優位性を維持することがきわめて重要です。

適応性を維持するには、次の点を考慮します。

- アプリケーション、設定、インフラストラクチャ、データ、AWS アカウントと組織、デプロイパイプライン、ネットワーク、セキュリティおよびコンプライアンス管理といったあらゆる要素をコードとして定義します。
- コンピューティングイメージ、共有サービス、アプリケーションそれぞれに応じたデプロイパイプラインを作成します。
- GitOps モデルの導入を検討します。このモデルに従ってプルベースのリクエストでワークフローを開始することで、コードの記述どおりに、既存インフラストラクチャの状態と望ましい状態を比較しながら変更をデプロイします。
- CD パイプラインを使用して、機械学習 (ML)、データ、モノのインターネット (IoT) といったワークロードをデプロイすることを検討します。
- すべてのビルドアーティファクトにデジタル署名し、安全なリポジトリに保存します。
- ソフトウェアの出どころを追跡します。これを行うには、ソフトウェア部品表を自動生成することで、顧客にデプロイするすべてのアーティファクト (バージョン管理とデジタル署名の対象) を記録します。
- ソフトウェアデリバリープロセス内の手動アクティビティをすべて削除した後に、手動のレビューボードを削除します。

ソフトウェアデリバリープロセス全体が自動化されたアプリケーションおよびサービスの場合は、継続的デプロイを検討してください。この手法では、パイプライン内の全チェックに合格した変更を顧客の本番環境にデプロイします。可視化については、AWS ウェブサイトで、「What is Continuous Delivery?」にある最初の図を参照してください。

AI/ML 技術を統合する

人工知能 (AI) と機械学習 (ML) の技術を CI/CD パイプラインに統合すると、次のような利点を得られます。

- 自動テスト生成
- テストの優先順位付けのインテリジェント化
- 予測分析による問題検出
- 異常検出と根本原因分析
- コードレビューと品質保証
- デプロイの最適化

詳細については、AWS ウェブサイトの「デベロッパーのオペレーションにインテリジェンスを追加する」を参照してください。

カオスエンジニアリングプラクティスを導入する

カオスエンジニアリングでは、意図的にシステム障害を発生させ、システムが持つ予期しないイベントへの耐性と、回復力をテストします。これにより、弱点を特定し、プロアクティブに対処できるため、システム全体の信頼性が向上し、起こり得る問題の影響が最小化されます。

カオスエンジニアリングプラクティスを導入してシステムの耐障害性をテストするには、Gremlin、Chaos Monkey、Litmus などのツールを使用します。また、制御された実験を定期的に実行して脆弱性を特定し、耐障害性を検証するとともに、予期しない障害が適切に処理されるようアプリケーションを強化します。こうしたプロアクティブなアプローチが、システムの信頼性向上や、より堅牢な CI/CD パイプラインの実現に寄与するのです。

パフォーマンスの最適化

アプリケーションのパフォーマンスを継続的に最適化するには、プロファイリングツール、リアルタイムモニタリング、フィードバックループを使用します。例えば、次のような手法を適用して、増加したトラフィックと需要をアプリケーションで処理できるようにします。

- コード最適化
- プロファイリング
- リアルタイムモニタリング
- フィードバックループ
- キャッシュ
- 負荷分散
- スケーラビリティおよびパフォーマンスのテスト

高度なオブザーバビリティを実装する

クラウドインフラストラクチャのオブザーバビリティを高めるには、メトリクス、ログ、トレースの基本的な収集、集約、分析を超える機能が必要です。Amazon CloudWatch や AWS X-Ray などのツールによってオブザーバビリティを強化すると、継続的デリバリーおよびイノベーションを推進する戦略的プラクティスに発展させることができます。

堅牢な CI/CD パイプラインで高度なオブザーバビリティを実現すると、アプリケーションやインフラストラクチャだけでなく、パイプライン自体を含め、システム全体のパフォーマンスと正常性に関するインサイトを導き出せます。こうしたインサイトにより、以下の対応を行えます。

- 起こり得る問題を迅速に特定、理解、対処して、アプリケーションの安定性を向上させるとともに、ダウンタイムを削減する
- CI/CD プロセスを合理化して、より迅速で信頼性の高いデリバリーを行う
- コード変更とデプロイの影響を詳細に把握し、情報に基づく意思決定を促進する
- リソース使用の最適化によって、業務効率と費用対効果を向上させる

オブザーバビリティを高めるには:

- オブザーバビリティをアプリケーションとインフラストラクチャの全レイヤーで確保し、包括的なビューで、システムのパフォーマンス、動作、正常性を確認できるようにします。
- Amazon CloudWatch などのツールによって、データ収集、ストレージ、分析を一元化することで、オブザーバビリティデータを統合し、アクセスや解釈を容易に行えるようにします。
- 分散トレースに AWS X-Ray を使用して、アプリケーションとその基盤サービスのパフォーマンスを把握します。
- 継続的改善のフィードバックループを確立するとともに、オブザーバビリティデータを使用してシステムの反復的な機能強化を推進します。

高度なオブザーバビリティの導入で重要なのは、システムの維持だけではありません。運用上の優秀性実現と継続的イノベーション推進に向けた戦略的な活動も重視する必要があります。

GitOps プラクティスを実装する

GitOps プラクティスを実装し、Git リポジトリを信頼できる単一のソースとして使用して、インフラストラクチャとアプリケーションの設定を管理します。こうしたアプローチを取ることで、変更管理の簡素化、トレーサビリティの向上、環境間での一貫性確保などが可能になります。

結論

このガイドは、クラウド導入を成功させる基盤を効果的に実装し管理するプレイブックとして利用でき、以下の方法を解説するものです。

- プラットフォームアーキテクチャの技術的な課題や複雑さに直接対処することで、クラウド環境とそこに存在するデータの堅牢なガイドラインと原則を確立します。
- きわめて効果的なプロビジョニングおよびオーケストレーションを実現することで、プラットフォームエンジニアリング基盤を構築します。
- コンプライアンスが確保されたマルチアカウントクラウド環境を用意し、承認済みクラウド製品をスケーラブルかつ反復可能な方法で管理および配布します。
- データエンジニアリングに必要なツールを使用してデータアーキテクチャに関する意思決定をサポートすることで、データに基づく意思決定を推進します。
- こうした能力をモダンアプリケーション開発戦略や CI/CD プロセスと組み合わせて、組織内の俊敏性と効率性を高め、イノベーションを推進します。
- 部門間の関係を築くとともに、自身の意思決定に他の AWS CAF の視点から情報を取り入れることで、プラットフォームとそれを支えるチームにとっての成果を得られるようにします。

詳細情報

AWS クラウド導入フレームワーク (AWS CAF) のリソース

- eBook
- オーディオブック
- インフォグラフィック
- 人工知能、機械学習、および生成 AI の AWS CAF
- ビジネスのパースペクティブ
- 人々の視点
- ガバナンスのパースペクティブ
- オペレーションのパースペクティブ
- セキュリティのパースペクティブ

その他のリソース:

- AWS アーキテクチャセンター
- AWS 導入事例
- AWS 参考文献
- AWS「用語集」
- AWS 情報センター
- AWS 規範ガイダンス
- AWS パートナーソリューション (以前のクイックスタート)
- AWS セキュリティドキュメント
- AWS ソリューションライブラリ
- AWS トレーニングと認定
- AWS Well-Architected
- AWS ホワイトペーパーとガイド
- のご利用開始にあたってAWS
- Overview of Amazon Web Services

寄稿者

このガイドの寄稿者は次のとおりです。

- AWS、Senior Partner Solutions Architect、Tony Santiago
- AWS、Enterprise Technologist、Matias Undurraga
- AWS、Senior Solutions Architect、Alex Torres
- AWS、Senior DevSecOps Consultant、Michael Rhyndress
- AWS、Principal Solutions Architect and CloudOps Specialist、Alex Livingstone
- AWS、Principal SDE、Bruce Cooper
- AWS、Senior Advisory Consultant、Ravinder Thota
- AWS、Senior Practice Manager、Sausan Yazji
- AWS、DevSecOps、Director、Paul Duvall
- AWS、Principal Cloud Delivery Manager、Jeremy Tennant
- AWS、Principal Infrastructure Lead、Sneh Shah
- AWS、AWS Cloud Adoption Framework、Worldwide Lead、Sasa Baskarada

ドキュメント履歴

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

変更	説明	日付
初版発行	—	2023 年 10 月 25 日

AWS 規範ガイダンスの用語集

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

数字

7 Rs

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

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

A

ABAC

「属性ベースのアクセス制御」をご覧ください。

抽象化されたサービス

「マネージドユーザー」をご覧ください。

ACID

「原子性、一貫性、分離性、耐久性 (ACID)」をご覧ください。

アクティブ/アクティブ移行

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

アクティブ/パッシブ移行

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

集計関数

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

AI

「人工知能」をご覧ください。

AIOps

「AI オペレーション」をご覧ください。

匿名化

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

アンチパターン

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

アプリケーション制御

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

アプリケーションポートフォリオ

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

人工知能 (AI)

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

AI オペレーション (AIOps)

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

非対称暗号化

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

原子性、一貫性、分離性、耐久性 (ACID)

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

属性ベースのアクセス制御 (ABAC)

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

信頼できるデータソース

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

アベイラビリティーゾーン (AZ)

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

AWS クラウド導入フレームワーク (AWS CAF)

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

AWS ワークロード認定フレームワーク (AWS WQF)

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

B

不正なボット

個人や組織に混乱や損害を与えることを目的としたボット。

BCP

「ビジネス継続性計画 (BCP)」をご覧ください。

動作グラフ

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

ビッグエンディアンシステム

最上位バイトを最初に格納するシステム。「エンディアン性」もご覧ください。

二項分類

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

ブルームフィルター

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

ブルー/グリーンデプロイ

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

ボット

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

ボットネット

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

ブランチ

コードリポジトリに含まれる領域。リポジトリに最初に作成するブランチは、メインブランチといいます。既存のブランチから新しいブランチを作成し、その新しいブランチで機能を開発した

り、バグを修正したりできます。機能を構築するために作成するブランチは、通常、機能ブランチと呼ばれます。機能をリリースする準備ができたら、機能ブランチをメインブランチに統合します。詳細については、「ブランチの概要」(GitHub ドキュメント) を参照してください。

ブレークグラスアクセス

例外的な状況では、承認されたプロセスを通じて、ユーザーが AWS アカウント 通常アクセス許可を持たない 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、 AWS Well-Architected ガイダンスの「ブレークグラス手順の実装」インジケータを参照してください。

ブラウンフィールド戦略

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

バッファキャッシュ

アクセス頻度が最も高いデータが保存されるメモリ領域。

ビジネス能力

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

ビジネス継続性計画 (BCP)

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

C

CAF

「AWS クラウド導入フレームワーク」を参照してください

カナリアデプロイ

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

CCoE

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

CDC

「変更データキャプチャ」を参照してください。

変更データキャプチャ (CDC)

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

カオスエンジニアリング

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

CI/CD

「継続的インテグレーションと継続的デリバリー」を参照してください。

分類

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

クライアント側の暗号化

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

Cloud Center of Excellence (CCoE)

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

クラウドコンピューティング

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

クラウド運用モデル

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

導入のクラウドステージ

組織が、 AWS クラウドへの移行時に通常実行する 4 つの段階。

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

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

CMDB

「構成管理データベース (CMDB)」を参照してください。

コードリポジトリ

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

コールドキャッシュ

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

コールドデータ

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

コンピュータビジョン (CV)

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

設定ドリフト

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

構成管理データベース (CMDB)

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

コンフォーマンスパック

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

継続的インテグレーションと継続的デリバリー (CI/CD)

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

CV

「コンピュータビジョン」を参照してください。

D

保管中のデータ

ストレージ内にあるデータなど、常に自社のネットワーク内にあるデータ。

データ分類

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

データドリフト

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

転送中のデータ

ネットワーク内 (ネットワークリソース間など) を活発に移動するデータ。

データメッシュ

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

データ最小化

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

データ境界

AWS 環境内の一連の予防ガードレール。信頼できる 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できるようにします。詳細については、「 でのデータ境界の構築 AWS」を参照してください。

データの前処理

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

データ出所

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

データ件名

データを収集、処理している個人。

データウェアハウス

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

データベース定義言語 (DDL)

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

データベース操作言語 (DML)

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

DDL

「データベース定義言語」を参照してください。

ディープアンサンブル

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

深層学習

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

多層防御

一連のセキュリティメカニズムとコントロールをコンピュータネットワーク全体に層状に重ねて、ネットワークとその内部にあるデータの機密性、整合性、可用性を保護する情報セキュリティの手法。この戦略を採用するときは AWS、リソースの保護に役立つように、 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ます。たとえば、多層防御アプローチでは、多要素認証、ネットワークセグメンテーション、暗号化を組み合わせることができます。

委任管理者

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

Organizations ドキュメントの「AWS Organizationsで使用できるサービス」を参照してください。

トラブルシューティング

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

開発環境

「環境」を参照してください。

検出管理

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

開発バリューストリームマッピング (DVSM)

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

デジタルツイン

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

ディメンションテーブル

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

ディザスタ

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

ディザスタリカバリ (DR)

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

DML

「データベース操作言語」を参照してください。

ドメイン駆動型設計

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

DR

「ディザスタリカバリ」を参照してください。

ドリフト検出

ベースライン設定からの偏差を追跡します。たとえば、 AWS CloudFormation を使用してシステムリソースのドリフトを検出したり、 を使用して AWS Control Tower 、ガバナンス要件のコンプライアンス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

「開発バリューストリームマッピング」を参照してください。

E

EDA

「探索的データ分析」を参照してください。

EDI

「電子データ交換」を参照してください。

エッジコンピューティング

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

電子データ交換 (EDI)

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

暗号化

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

暗号化キー

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

エンディアン

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

エンドポイント

「サービスエンドポイント」を参照してください。

エンドポイントサービス

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

エンタープライズリソースプランニング (ERP)

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

エンベロープ暗号化

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

環境

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

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

エピック

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

ERP

「エンタープライズリソース計画」を参照してください。

探索的データ分析 (EDA)

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

F

ファクトテーブル

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

フェイルファスト

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

障害分離境界

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

機能ブランチ

「ブランチ」を参照してください。

特徴量

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

特徴量重要度

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

機能変換

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

数ショットプロンプト

LLM に、タスクと望ましい出力を示す例を少数提示した後に、類似のタスクを実行させること。この手法は、プロンプトに記述された例 (ショット) からモデルが学習する「インコンテキスト学

習」の一種です。数ショットプロンプトは、特定のフォーマット、推論、専門知識が必要なタスクに効果的です。「ゼロショットプロンプト」も参照してください。

FGAC

「きめ細かなアクセス制御」を参照してください。

きめ細かなアクセス制御 (FGAC)

複数の条件を使用してアクセス要求を許可または拒否すること。

フラッシュカット移行

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

FM

「基盤モデル」を参照してください。

基盤モデル (FM)

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

G

生成 AI

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

ジオブロッキング

「地理的制限」を参照してください。

地理的制限 (ジオブロッキング)

特定の国のユーザーがコンテンツ配信にアクセスできないようにするための、Amazon CloudFront のオプション。アクセスを許可する国と禁止する国は、許可リストまたは禁止リスト

を使って指定します。詳細については、CloudFront ドキュメントの「コンテンツの地理的ディストリビューションの制限」を参照してください。

Gitflow ワークフロー

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

ゴールデンイメージ

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

グリーンフィールド戦略

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

ガードレール

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

H

HA

「高可用性」を参照してください。

異種混在データベースの移行

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

高可用性 (HA)

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

ヒストリアンのモダナイゼーション

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

ホールドアウトデータ

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

同種データベースの移行

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

ホットデータ

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

ホットフィックス

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

ハイパーケア期間

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

I

IaC

「Infrastructure as Code」を参照してください。

ID ベースのポリシー

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

アイドル状態のアプリケーション

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

IIoT

「インダストリアル IoT」を参照してください。

イミュータブルインフラストラクチャ

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

インバウンド (受信) VPC

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

増分移行

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

インダストリー 4.0

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

インフラストラクチャ

アプリケーションの環境に含まれるすべてのリソースとアセット。

Infrastructure as Code (IaC)

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

インダストリアル IoT (IIoT)

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

インスペクション VPC

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

IoT

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

解釈可能性

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

IoT

「IoT」を参照してください。

IT 情報ライブラリ (ITIL)

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

IT サービス管理 (ITSM)

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

ITIL

「IT 情報ライブラリ」を参照してください。

ITSM

「IT サービス管理」を参照してください。

L

ラベルベースアクセス制御 (LBAC)

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

ランディングゾーン

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

大規模言語モデル (LLM)

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

大規模な移行

300 台以上のサーバの移行。

LBAC

「ラベルベースアクセス制御」を参照してください。

最小特権

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

リフトアンドシフト

「7 Rs」を参照してください。

リトルエンディアンシステム

最下位バイトを最初に格納するシステム。「エンディアン性」もご覧ください。

LLM

「大規模言語モデル」を参照してください。

下位環境

「環境」を参照してください。

M

機械学習 (ML)

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

メインブランチ

「ブランチ」を参照してください。

マルウェア

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

マネージドサービス

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

製造実行システム (MES)

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

MAP

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

メカニズム

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

メンバーアカウント

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

MES

「製造実行システム」を参照してください。

Message Queuing Telemetry Transport (MQTT)

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

マイクロサービス

明確に定義された API を介して通信し、通常は小規模な自己完結型のチームが所有する、小規模で独立したサービスです。例えば、保険システムには、販売やマーケティングなどのビジネス

機能、または購買、請求、分析などのサブドメインにマッピングするマイクロサービスが含まれる場合があります。マイクロサービスの利点には、俊敏性、柔軟なスケーリング、容易なデプロイ、再利用可能なコード、回復力などがあります。詳細については、AWS「サーバーレスサービスを使用したマイクロサービスの統合」を参照してください。

マイクロサービスアーキテクチャ

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

Migration Acceleration Program (MAP)

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

大規模な移行

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

移行ファクトリー

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

移行メタデータ

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

移行パターン

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

Migration Portfolio Assessment (MPA)

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

移行準備状況評価 (MRA)

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

移行戦略

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

ML

「機械学習」を参照してください。

モダナイゼーション

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

モダナイゼーション準備状況評価

組織のアプリケーションのモダナイゼーションの準備状況を判断し、利点、リスク、依存関係を特定し、組織がこれらのアプリケーションの将来の状態をどの程度適切にサポートできるかを決定するのに役立つ評価。評価の結果として、ターゲットアーキテクチャのブループリント、モダナイゼーションプロセスの開発段階とマイルストーンを詳述したロードマップ、特定されたギャップに対処するためのアクションプランが得られます。詳細については、「AWS クラウドでのアプリケーションのモダナイゼーションの準備状況を評価する」を参照してください。

モノリシックアプリケーション (モノリス)

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

MPA

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

MQTT

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

多クラス分類

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

ミュータブルなインフラストラクチャ

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

O

OAC

「オリジンアクセス制御」を参照してください。

OAI

「オリジンアクセスアイデンティティ」を参照してください。

OCM

「組織変更管理」を参照してください。

オフライン移行

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

OI

「オペレーション統合」を参照してください。

Ola

「オペレーショナルレベルアグリーメント」を参照してください。

オンライン移行

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

OPC-UA

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

Open Process Communications - Unified Architecture (OPC-UA)

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

オペレーショナルレベルアグリーメント (OLA)

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

運用準備状況レビュー (ORR)

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

運用テクノロジー (OT)

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

オペレーション統合 (OI)

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

組織の証跡

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

組織変更管理 (OCM)

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

オリジンアクセス制御 (OAC)

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

オリジンアクセスアイデンティティ (OAI)

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

ORR

「運用準備状況レビュー」を参照してください。

OT

「運用テクノロジー」を参照してください。

アウトバウンド (送信) VPC

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

P

アクセス許可の境界

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

個人を特定できる情報 (PII)

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

PII

「個人を特定できる情報」を参照してください。

プレイブック

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

PLC

「プログラマブルロジックコントローラー」を参照してください。

PLM

「製品ライフサイクル管理」を参照してください。

ポリシー

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

多言語の永続性

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

ポートフォリオ評価

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

述語

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

述語プッシュダウン

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

予防的コントロール

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

プリンシパル

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

プライバシーバイデザイン

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

プライベートホストゾーン

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

プロアクティブコントロール

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

製品ライフサイクル管理 (PLM)

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

本番環境

「環境」を参照してください。

プログラマブルロジックコントローラー (PLC)

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

プロンプトチェイニング

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

仮名化

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

発行/サブスクライブ (pub/sub)

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

Q

クエリプラン

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

クエリプランのリグレッション

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

R

RACI マトリックス

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

RAG

「検索拡張生成」を参照してください。

ランサムウェア

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

RASCI マトリックス

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

RCAC

「行と列のアクセス制御」を参照してください。

リードレプリカ

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

リアーキテクト

「7 Rs」を参照してください。

目標復旧時点 (RPO)

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

目標復旧時間 (RTO)

サービスの中断から復旧までの最大許容遅延時間。

リファクタリング

「7 Rs」を参照してください。

リージョン

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

リグレッション

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

リホスト

「7 Rs」を参照してください。

リリース

デプロイプロセスで、変更を本番環境に昇格させること。

再配置

「7 Rs」を参照してください。

リプラットフォーム

「7 Rs」を参照してください。

再購入

「7 Rs」を参照してください。

回復性

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

リソースベースのポリシー

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

実行責任者、説明責任者、協業先、報告先 (RACI) に基づくマトリックス

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

レスポンシブコントロール

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

保持

「7 Rs」を参照してください。

廃止

「7 Rs」を参照してください。

検索拡張生成 (RAG)

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

ローテーション

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

行と列のアクセス制御 (RCAC)

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

RPO

「目標復旧時点」を参照してください。

RTO

「目標復旧時間」を参照してください。

ランブック

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

S

SAML 2.0

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

SCADA

「監視制御とデータ取得」を参照してください。

SCP

「サービスコントロールポリシー」を参照してください。

シークレット

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

セキュリティバイデザイン

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

セキュリティコントロール

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

セキュリティ強化

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

Security Information and Event Management (SIEM) システム

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

セキュリティレスポンスの自動化

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

サーバー側の暗号化

送信先で、それ AWS のサービス を受け取る によるデータの暗号化。

サービスコントロールポリシー (SCP)

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

サービスエンドポイント

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

サービスレベルアグリーメント (SLA)

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

サービスレベルインジケータ (SLI)

エラー率、可用性、スループットといった、サービスパフォーマンス面の指標。

サービスレベル目標 (SLO)

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

責任共有モデル

クラウドのセキュリティとコンプライアンス AWS について と共有する責任を説明するモデル。AWS はクラウドのセキュリティを担当しますが、 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

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

単一障害点 (SPOF)

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

SLA

「サービスレベルアグリーメント」を参照してください。

SLI

「サービスレベルインジケータ」を参照してください。

SLO

「サービスレベルの目標」を参照してください。

スプリットアンドシードモデル

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

SPOF

「単一障害点」を参照してください。

スタースキーマ

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

strangler fig パターン

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

サブネット

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

監視制御とデータ取得 (SCADA)

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

対称暗号化

データの暗号化と復号に同じキーを使用する暗号化のアルゴリズム。

合成テスト

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

システムプロンプト

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

T

タグ

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

ターゲット変数

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

タスクリスト

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

テスト環境

「環境」を参照してください。

トレーニング

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

トランジットゲートウェイ

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

トランクベースのワークフロー

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

信頼されたアクセス

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

チューニング

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

ツーピザチーム

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

U

不確実性

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

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

ワークロード

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

ワークストリーム

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

WORM

「Write-Once-Read-Many」を参照してください。

WQF

「AWS ワークロード資格フレームワーク」を参照してください

Write-Once-Read-Many (WORM)

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

Z

ゼロデイエクスプロイト

ゼロデイ脆弱性を悪用した攻撃 (一般的にマルウェアによる)。

ゼロデイ脆弱性

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

ゼロショットプロンプト

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

ゾンビアプリケーション

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

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