AWS ホワイトペーパー

AWS でのデプロイオプションの概要

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS でのデプロイオプションの概要: AWS ホワイトペーパー

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon の商標およびトレードドレスはAmazon 以外の製品およびサービスに使用することはできません。また、お客様に誤解を与える可能性がある形式で、または Amazon の信用を損なう形式で使用することもできません。Amazon が所有していない他のすべての商標は、それぞれの所有者の所有物であり、Amazon と提携、接続、または後援されている場合とされていない場合があります。

Table of Contents

要約 1
要約 1
序章 2
AWS デプロイサービス 3
AWS CloudFormation 3
AWS Elastic Beanstalk 6
AWS CodeDeploy 10
AWS CodeDeploy の AWS Lambda 12
Amazon Elastic Container Service 13
Amazon ECS Anywhere 16
での Amazon Elastic Container Service AWS Outposts 17
アマゾン エラスティックKubernetesサービス 18
Amazon EKS Anywhere 22
AWS App Runner 22
Amazon Lightsail 24
Amazon Lightsail コンテナ 25
Red Hat OpenShift Service on AWS 26
AWS Local Zones 26
AWS Wavelength 27
追加のデプロイサービス 27
Amazon Simple Storage Service 27
AWS Proton 27
AWS App2Container 28
AWS Copilot 28
AWS Serverless Application Model 29
AWS Cloud Development Kit (AWS CDK) 29
Amazon EC2 Image Builder 30
デプロイ戦略 33
プリベーキングとブートストラップの AMIs 33
ブルー/グリーンデプロイ 33
ローリングデプロイ 34
Canary デプロイ 34
インプレースデプロイ 35
デプロイサービスの組み合わせ 35

結論 37
寄稿者 38
詳細情報 39
ドキュメントの改訂 40
注意 41
..... xlii

AWS でのデプロイオプションの概要

公開日: 2024 年 5 月 31 日 (ドキュメントの改訂)

要約

アマゾン ウェブ サービス (AWS) には、インフラストラクチャのプロビジョニングとアプリケーションのデプロイのための複数のオプションが用意されています。アプリケーションアーキテクチャがシンプルな 3 層ウェブアプリケーションであっても、複雑なワークロードのセットであっても、AWS はアプリケーションと組織の要件を満たすためのデプロイサービスを提供します。

このホワイトペーパーは、AWS が提供するさまざまなデプロイサービスの概要を求めるユーザーを対象としています。これらのデプロイサービスで利用可能な一般的な機能を示し、アプリケーションスタックをデプロイおよび更新するための基本的な戦略を明確にします。

序章

アプリケーションのデプロイソリューションを設計することは、AWS で適切に設計されたアプリケーションを構築する上で重要な部分です。アプリケーションの性質と必要な基盤となるサービスに基づいて、AWS のサービスを使用して、アプリケーションと組織の両方のニーズに合わせて調整できる柔軟なデプロイソリューションを作成できます。

増え続ける AWS サービスのカタログは、アプリケーションアーキテクチャを構成するサービスを決定するプロセスだけでなく、アプリケーションの作成、管理、更新方法を決定するプロセスも複雑にします。AWS でデプロイソリューションを設計するときは、ソリューションが以下の機能にどのように対処するかを検討する必要があります。

- プロビジョニング - アプリケーションに必要な raw インフラストラクチャまたはマネージドサービスインフラストラクチャを作成します。
- 設定 - 環境、ランタイム、セキュリティ、可用性、パフォーマンス、ネットワーク、またはその他のアプリケーション要件に基づいてインフラストラクチャをカスタマイズします。
- デプロイ - インフラストラクチャリソースにアプリケーションコンポーネントをインストールまたは更新し、以前のアプリケーションバージョンから新しいアプリケーションバージョンへの移行を管理します。
- スケーリング - 一連のユーザー定義基準に基づいて、アプリケーションで使用可能なリソースの量をプロアクティブまたは事後的に調整します。
- モニタリング - アプリケーションアーキテクチャの一部として起動されるリソースを可視化します。リソースの使用状況、デプロイの成功または失敗、アプリケーションのヘルス、アプリケーションログ、設定ドリフトなどを追跡します。

このホワイトペーパーでは、AWS が提供するデプロイサービスについて説明し、あらゆるタイプのアプリケーションに適したデプロイアーキテクチャを設計するための戦略の概要を説明します。

AWS デプロイサービス

スケーラブルで効率的で費用対効果の高いデプロイソリューションを設計するタスクは、アプリケーションバージョンの更新方法に限定されず、アプリケーションライフサイクル全体を通じてサポートインフラストラクチャを管理する方法も考慮する必要があります。リソースのプロビジョニング、設定管理、アプリケーションのデプロイ、ソフトウェアの更新、モニタリング、アクセスコントロール、その他の懸念事項はすべて、デプロイソリューションを設計する際に考慮すべき重要な要素です。

AWS サービスは、アプリケーションライフサイクルの 1 つ以上の側面の管理機能を提供できます。必要な制御のバランス (リソースの手動管理) と利便性 (リソースの AWS 管理) およびアプリケーションの種類に応じて、これらのサービスを単独で使用することも、組み合わせて機能豊富なデプロイソリューションを作成することもできます。このセクションでは、組織がより迅速かつ確実にアプリケーションを構築して配信できるようにするために使用できる AWS サービスの概要を説明します。

AWS CloudFormation

AWS CloudFormation は、YAML または JSON で表現されたカスタムテンプレート言語を使用して、ほぼすべての AWS リソースをプロビジョニングおよび管理できるようにするサービスです。テンプレートは AWS CloudFormation 、スタックと呼ばれるグループにインフラストラクチャリソースを作成し、これらのリソースを完全に制御しながら、アプリケーションの運用に必要なすべてのコンポーネントを定義およびカスタマイズできます。テンプレートを使用すると、インフラストラクチャにバージョン管理を実装したり、インフラストラクチャを迅速かつ確実にレプリケートしたりできます。

AWS CloudFormation は、ルートテーブルやサブネット設定などの低レベルコンポーネントから CloudFront ディストリビューションなどの高レベルコンポーネントまで、すべてのアプリケーションインフラストラクチャコンポーネントのプロビジョニングと管理をきめ細かく制御します。 AWS CloudFormation は、他の AWS デプロイサービスやサードパーティーツールと一般的に使用され、より特殊なデプロイサービスと組み合わせて AWS CloudFormation 、インフラストラクチャコンポーネントへのアプリケーションコードのデプロイを管理します。

AWS は、CloudFormation サービスの基本機能に加えて、以下の拡張機能を提供しています。

- AWS Cloud Development Kit (AWS CDK) は、TypeScript、JavaScript、Python、Java、または C#/ を使用して AWS インフラストラクチャをプログラムでモデル化するオープンソースの Software Development Kit (SDK) です。NET。

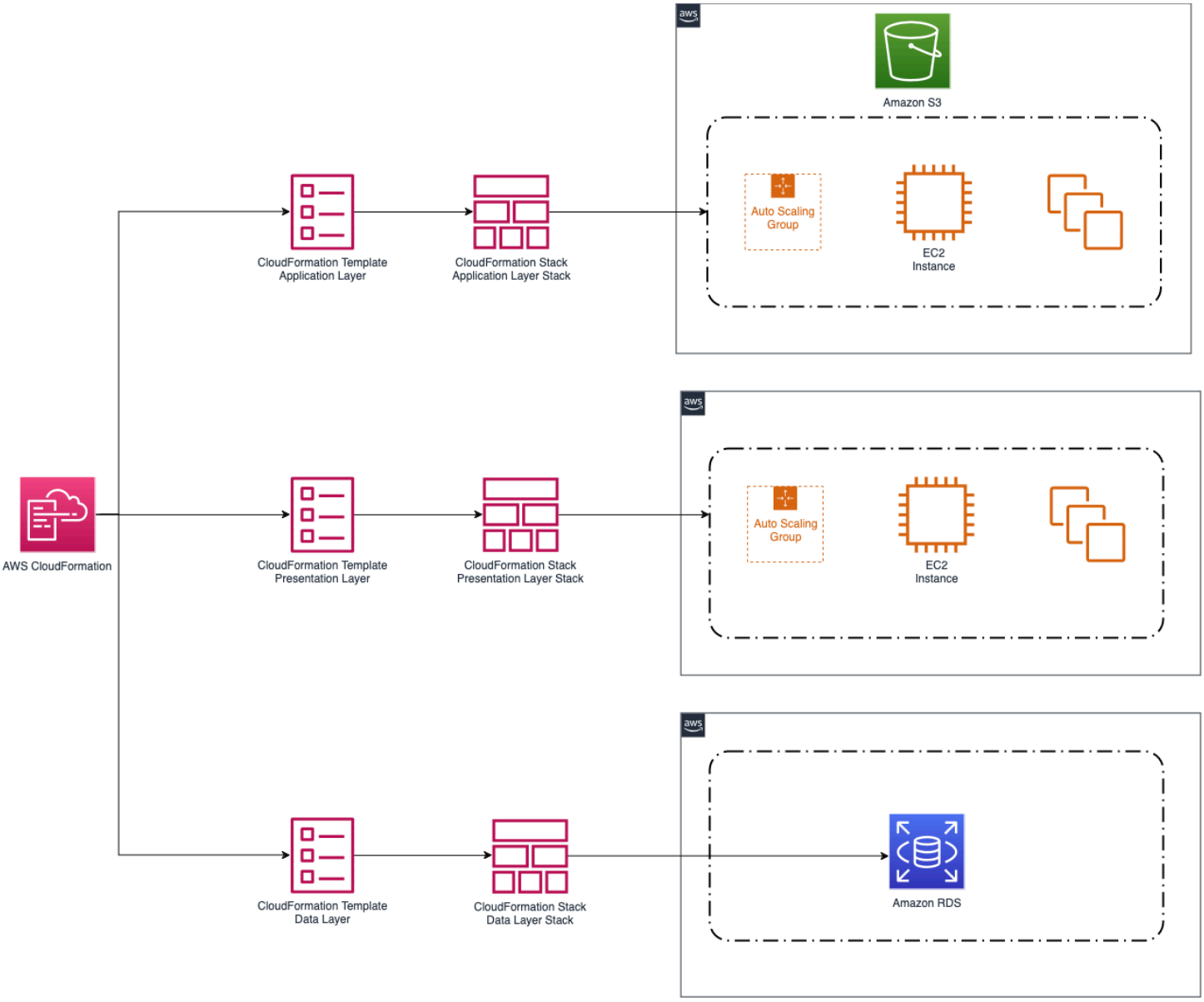
- AWS Serverless Application Model (AWS SAM) は、AWS でのサーバーレスアプリケーションの構築を簡素化するオープンソースフレームワークです。関数、API、データベース、イベントソースマッピングを表現するための省略構文を提供します。

表 1: AWS CloudFormation デプロイ機能

機能	説明
プロビジョニング	CloudFormation は、テンプレートで定義されているインフラストラクチャコンポーネントを自動的に作成および更新します。 AWS CloudFormation テンプレートを使用したインフラストラクチャの作成の詳細については、AWS CloudFormation 「ベストプラクティス」を参照してください。
構成する	AWS CloudFormation テンプレートは、すべてのインフラストラクチャコンポーネントをカスタマイズおよび更新するための広範な柔軟性を提供します。 テンプレートのカスタマイズの詳細については、「テンプレートAWS CloudFormation の構造」を参照してください。
デプロイ	AWS CloudFormation テンプレートを更新して、スタック内のリソースを変更します。アプリケーションアーキテクチャによっては、インフラストラクチャで実行されているアプリケーションバージョンを更新するために、追加のデプロイサービスが必要になる場合があります。 をデプロイソリューションとして使用する方法の詳細については、「 を使用した Amazon EC2 でのアプリケーションの AWS CloudFormationデプロイ」を参照してください。 AWS CloudFormation

機能	説明
スケール	AWS CloudFormation はユーザーに代わってインフラストラクチャのスケーリングを自動的に処理しませんが、 AWS CloudFormation テンプレートでリソースの自動スケーリングポリシーを設定できます。
モニタリング	AWS CloudFormation は、テンプレートで定義されたインフラストラクチャの更新の成功または失敗をネイティブにモニタリングし、テンプレートで定義されたリソースが仕様を満たさない場合にモニタリングするためのドリフト検出を提供します。アプリケーションレベルのモニタリングとメトリクスには、追加のモニタリングソリューションが必要です。 がインフラストラクチャの更新を監視する方法の詳細については、「スタックの更新の進行状況のモニタリング」を参照してください。AWS CloudFormation

次の図は、一般的なユースケースを示しています AWS CloudFormation。ここでは、シンプルな AWS CloudFormation 3 層ウェブアプリケーションの作成に必要なすべてのインフラストラクチャコンポーネントを定義するテンプレートが作成されます。この例では、 で定義されたブートストラップスクリプトを使用して、アプリケーションの最新バージョン AWS CloudFormation を Amazon EC2 インスタンスにデプロイします。ただし、追加のデプロイサービスを AWS CloudFormation （ インフラストラクチャ管理およびプロビジョニング機能 AWS CloudFormation にのみ使用) と組み合わせることも一般的な方法です。インフラストラクチャの作成には複数の AWS CloudFormation テンプレートが使用されます。この図では、 AWS CloudFormation を使用して、IAM ロール、VPCs、サブネット、ルートテーブル、セキュリティグループ、Amazon S3 バケットポリシーなど、すべてのインフラストラクチャコンポーネントを作成します。アプリケーションアーキテクチャの各ドメインを構築するには、個別の AWS CloudFormation テンプレートを使用します。



AWS CloudFormation ユースケース

AWS Elastic Beanstalk

AWS Elastic Beanstalk は、Java、.NET、.NET Core、PHP、Node.js、Python、Ruby、Go、または Docker で開発されたウェブアプリケーションとサービスを、Apache、Nginx、Passenger、IIS などの使い慣れたサーバーにデプロイおよびスケーリングするためのeasy-to-useサービスです。Elastic Beanstalk は完全なアプリケーション管理ソリューションであり、ユーザーに代わってすべてのインフラストラクチャとプラットフォームタスクを管理します。

Elastic Beanstalk を使用すると、インフラストラクチャを管理する運用上の負担をかけずに、アプリケーションを迅速にデプロイ、管理、スケーリングできます。Elastic Beanstalk は、ウェブアプリケーションの管理の複雑さを軽減し、AWS を初めて使用する組織や、ウェブアプリケーションをできるだけ早くデプロイしたい組織に適しています。

デプロイソリューションとして Elastic Beanstalk を使用する場合、ソースコードをアップロードするだけで、Elastic Beanstalk はサーバー、データベース、ロードバランサー、ネットワーク、自動スケーリンググループなど、必要なすべてのインフラストラクチャをプロビジョニングして運用します。これらのリソースはユーザーに代わって作成されますが、これらのリソースを完全に制御できるため、開発者は必要に応じてカスタマイズできます。Elastic Beanstalk は、ISO、PCI、SOC 1、SOC 2、および SOC 3 への準拠の基準と HIPAA 適格性の基準を満たしています。つまり、Elastic Beanstalk で実行されているアプリケーションは、規制対象の財務データまたは保護対象の医療情報 (PHI) を処理できます。

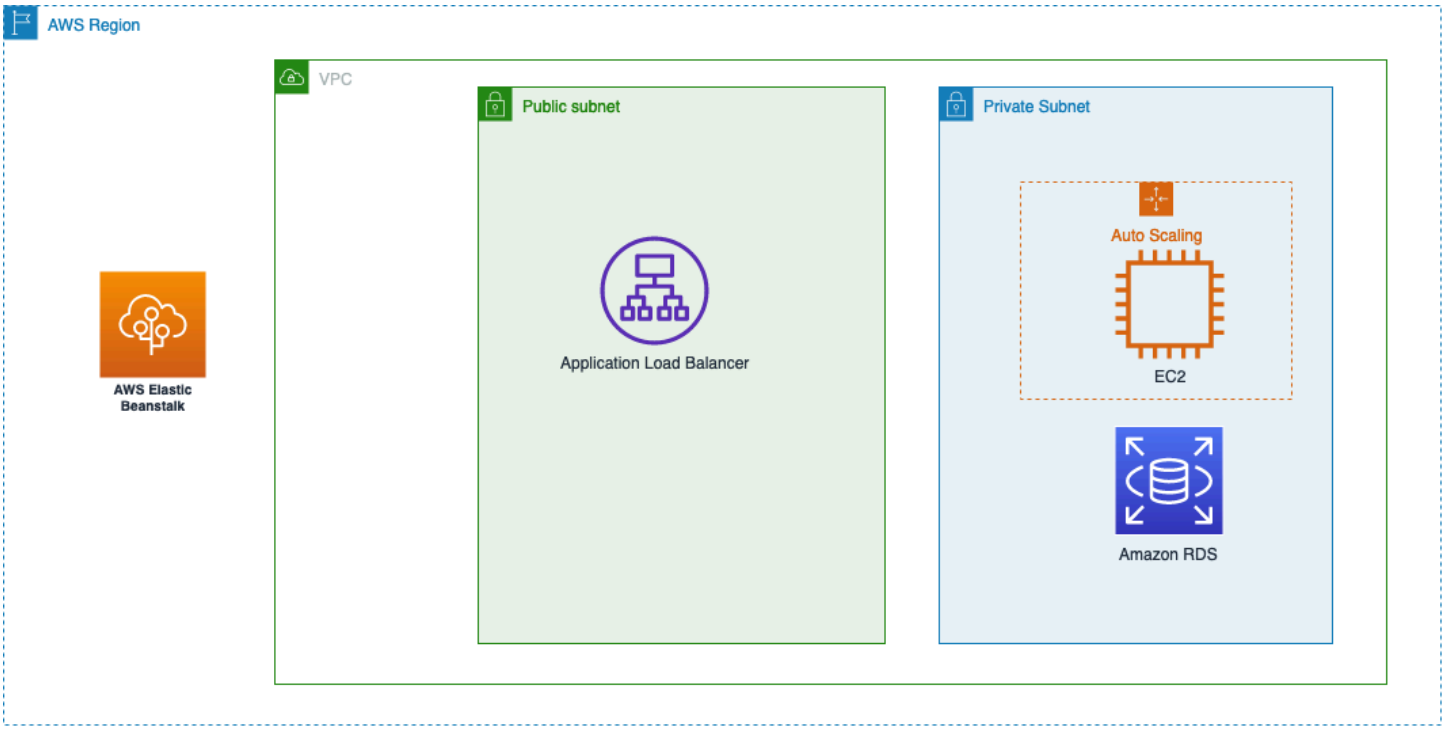
表 2: AWS Elastic Beanstalk デプロイ機能

機能	説明
プロビジョニング	Elastic Beanstalk は、サポートされているプラットフォームのいずれかで実行されるウェブアプリケーションまたはサービスを運用するために必要なすべてのインフラストラクチャコンポーネントを作成します。追加のインフラストラクチャが必要な場合は、Elastic Beanstalk の外部で作成する必要があります。 Elastic Beanstalk でサポートされているウェブアプリケーションプラットフォームの詳細については、「Elastic Beanstalk プラットフォーム」を参照してください。
構成する	Elastic Beanstalk には、環境内のリソースをカスタマイズするための幅広いオプションが用意されています。 Elastic Beanstalk によって作成されるリソースのカスタマイズの詳細については、「Elastic Beanstalk 環境の設定」を参照してください。

機能	説明
デプロイ	Elastic Beanstalk はアプリケーションのデプロイを自動的に処理し、既存のユーザーに影響を与えることなくアプリケーションの新しいバージョンを実行する環境を作成します。 Elastic Beanstalk を使用したアプリケーションのデプロイの詳細については、「 への AWS Elastic Beanstalkアプリケーションのデプロイ」を参照してください。
スケール	Elastic Beanstalk は、Elastic Load Balancing と Auto Scaling を使用して、特定のニーズに基づいてアプリケーションを自動的にスケールインおよびスケールアウトします。複数のアベイラビリティーゾーンを使用すると、アプリケーションの信頼性と可用性を向上させるオプションが提供されます。 Elastic Beanstalk による自動スケーリングの詳細については、Elastic Beanstalk 環境の Auto Scaling Group を参照してください。
モニタリング	Elastic Beanstalk には、デプロイの成功/失敗、環境ヘルス、リソースパフォーマンス、アプリケーションログなどのアプリケーションの組み込み環境モニタリングが用意されています。 Elastic Beanstalk を使用したフルスタックモニタリングの詳細については、「環境のモニタリング」を参照してください。

機能	説明
Graviton のサポート	AWS Graviton arm64 ベースのプロセッサは、Amazon EC2 で実行されているクラウドワークロードに最適な価格のパフォーマンスを提供します。Elastic Beanstalk の AWS Graviton を使用すると、ワークロードの最適化ニーズを満たす Amazon EC2 インスタンスタイプを選択でき、同等の x86 ベースのプロセッサよりも価格パフォーマンスが向上します。

Elastic Beanstalk を使用すると、ウェブアプリケーションを AWS で簡単にデプロイおよび管理できます。次の例は、シンプルなウェブアプリケーションのデプロイに使用される Elastic Beanstalk の一般的なユースケースを示しています。すべてのアプリケーションインフラストラクチャ (セキュリティグループ、IAM ロール、CloudWatch アラームを含む) は、Elastic Beanstalk によって作成および管理されます。Amazon EC2 インスタンスは、ランタイム環境とデプロイパッケージで自動的にプロビジョニングされます。Elastic Beanstalk 環境は、Elastic Beanstalk の外部で作成された Amazon Relational Database Service (Amazon RDS) などのリソースと統合できます。



AWS Elastic Beanstalk ユースケース

AWS CodeDeploy

AWS CodeDeploy は、Amazon EC2、Amazon Elastic Container Service (Amazon ECS)、AWS Lambda、オンプレミスサーバーなどのコンピューティングサービスへのアプリケーションのデプロイを自動化するフルマネージド型のデプロイサービスです。組織は CodeDeploy を使用してアプリケーションのデプロイを自動化し、デプロイプロセスからエラーが発生しやすい手動操作を削除できます。CodeDeploy は、コード、サーバーレス関数、設定ファイルなど、さまざまなアプリケーションコンテンツで使用できます。

CodeDeploy は、アプリケーションデベロッパーが既存のインフラストラクチャで実行されているソフトウェアをデプロイおよび更新するのを支援することに焦点を当てた構成要素サービスとして使用されることを目的としています。これはend-to-endのアプリケーション管理ソリューションではなく、完全な CI/CD パイプラインの一部としてAWS CodeStar、、AWS CodePipeline、その他の AWS 開発者ツール、サードパーティーサービス (製品統合の完全なリストについては「製品統合」を参照) などの他の AWS デプロイサービスと組み合わせて使用することを目的としています。 AWS CodeDeployさらに、CodeDeploy はユーザーに代わってリソースの作成を管理しません。

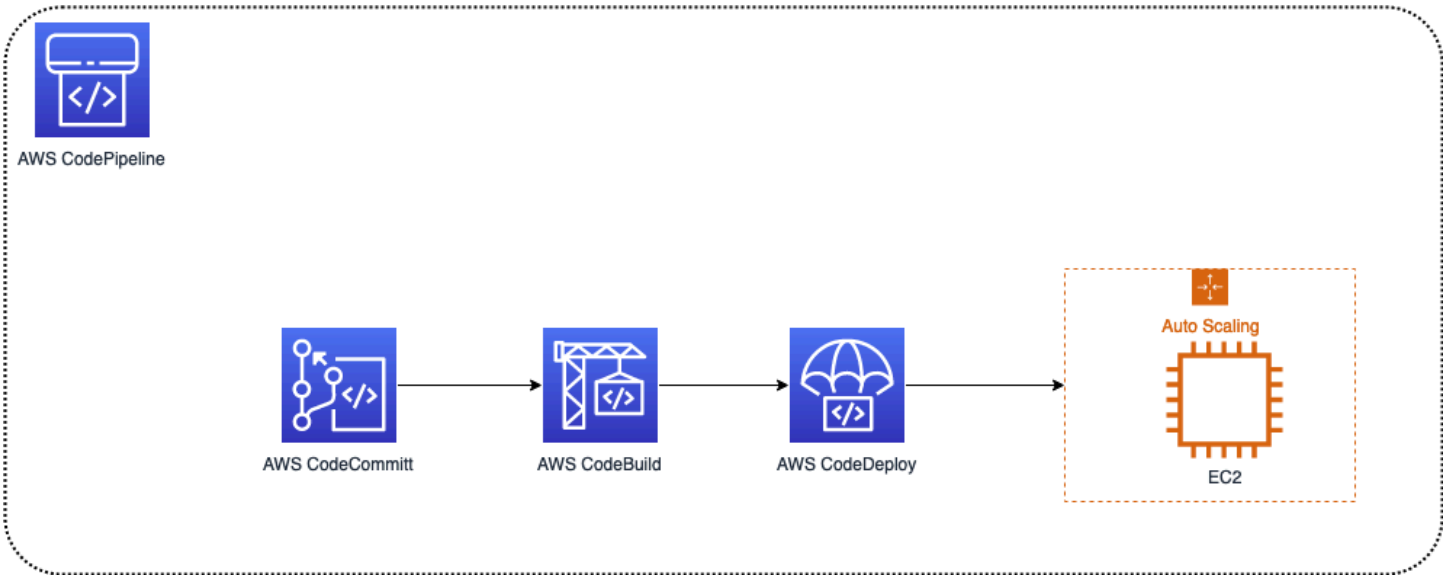
表 3: AWS CodeDeploy デプロイ機能

機能	説明
プロビジョニング	CodeDeploy は、既存のコンピューティングリソースでの使用を目的としており、ユーザーに代わってリソースを作成することはありません。CodeDeploy では、アプリケーションコンテンツをデプロイするために、コンピューティングリソースをデプロイグループと呼ばれるコンストラクトに整理する必要があります。 CodeDeploy をコンピューティングリソースにリンクする方法の詳細については、「CodeDeploy でのデプロイグループの操作」を参照してください。 CodeDeploy
構成する	CodeDeploy は、アプリケーション仕様ファイルを使用してコンピューティングリソースのカスタマイズを定義します。

機能	説明
	CodeDeploy でのリソースのカスタマイズの詳細については、「CodeDeploy AppSpec ファイルリファレンス」を参照してください。CodeDeploy
デプロイ	CodeDeploy が使用されるコンピューティングリソースのタイプに応じて、CodeDeploy はアプリケーションをデプロイするためのさまざまな戦略を提供します。 サポートされるデプロイプロセスのタイプの詳細については、CodeDeploy でのデプロイの使用」を参照してください。
スケール	CodeDeploy は基盤となるアプリケーションインフラストラクチャのスケーリングをサポートしていませんが、デプロイ設定によっては、ブルー/グリーンデプロイをサポートする追加のリソースを作成する場合があります。
モニタリング	CodeDeploy はデプロイの成功または失敗をモニタリングし、すべてのデプロイの履歴を提供しますが、パフォーマンスまたはアプリケーションレベルのメトリクスは提供しません。 CodeDeploy が提供するモニタリング機能のタイプの詳細については、「CodeDeploy でのデプロイのモニタリング」を参照してください。CodeDeploy

次の図は、完全な CI/CD ソリューションの一部としての CodeDeploy の一般的なユースケースを示しています。この例では、CodeDeploy は追加の AWS 開発者ツール、つまり AWS CodePipeline (CI/CD パイプラインを自動化)、 AWS CodeBuild (アプリケーションコンポーネントの構築とテスト)、 AWS CodeCommit (ソースコードリポジトリ) と組み合わせて使用され、Amazon EC2 インスタンスのグループにアプリケーションをデプロイします。CodeDeploy は、完全な CI/CD パイプラインの一部として他のツールとともに使用されます。CodeDeploy は、デプロイグループの一部で

あるコンピューティングリソースへのアプリケーションコンポーネントのデプロイを管理します。すべてのインフラストラクチャコンポーネントは CodeDeploy の外部で作成されます。



AWS CodeDeploy ユースケース

AWS CodeDeploy の AWS Lambda

AWS CodeDeploy for AWS Lambda を使用すると、サーバーレスデプロイを自動化できるため、アプリケーションリリースをより詳細に制御および可視化できます。CodeDeploy を使用して、サーバーレス関数の新しいバージョンを少数のユーザーまたはトラフィックにデプロイし、新しいバージョンに自信が持てるにつれてトラフィックを徐々に増やすことができます。CodeDeploy を使用すると、同じイベントソースからトラフィックを受信する一連の Lambda 関数を表すデプロイグループを定義できます。例えば、API Gateway または Amazon EventBridge ルールによって開始される一連の Lambda 関数のデプロイグループを作成できます。その後、CodeDeploy を使用してデプロイを作成できます。CodeDeploy は、指定したデプロイグループに新しいバージョンの Everless 関数をデプロイします。

CodeDeploy では、デプロイタイプ、デプロイ戦略、トラフィックシフトルールなど、デプロイの設定を指定するデプロイ設定を定義することもできます。Canary デプロイ戦略を使用して、サーバーレス関数の新しいバージョンを少数のトラフィックにデプロイし、新しいバージョンのヘルスとパフォーマンスをモニタリングしてから、トラフィックを増やすことができます。

サーバーレスに CodeDeploy を使用することで、デプロイプロセスを自動化し、アプリケーションの新しいバージョンのリリースに必要な時間と労力を削減し、サーバーレス関数の安定性と信頼性を向上させることができます。

Amazon Elastic Container Service

Amazon Elastic Container Service (Amazon ECS) は、フルマネージド型のコンテナオーケストレーションサービスで、Docker コンテナをサポートし、マネージドクラスターでアプリケーションを簡単に実行できます。Amazon ECS は、コンテナ管理インフラストラクチャをインストール、運用、スケーリングする必要をなくし、Security Groups、Elastic Load Balancing、 AWS Identity and Access Management (IAM) などの使い慣れた AWS コア機能を備えた環境の作成を簡素化します。

Amazon ECS でアプリケーションを実行する場合、Amazon EC2 インスタンスを使用するか、コンテナ用のサーバーレスコンピューティングエンジンである を使用してAWS Fargate、コンテナの基盤となるコンピューティング能力を提供するかを選択できます。いずれの場合も、Amazon ECS はユーザーが定義した設定に従って、コンテナをクラスターに自動的に配置してスケーリングします。Amazon ECS は、ユーザーに代わって Load Balancer や IAM ロールなどのインフラストラクチャコンポーネントを作成しませんが、Amazon ECS サービスは、Amazon ECS クラスターでのこれらのリソースの作成と使用を簡素化するために多数の APIs を提供します。

Amazon ECS を使用すると、デベロッパーはすべてのインフラストラクチャコンポーネントをきめ細かく直接制御できるため、カスタムアプリケーションアーキテクチャを作成できます。さらに、Amazon ECS は、アプリケーションコンテナイメージを更新するためのさまざまなデプロイ戦略をサポートしています。

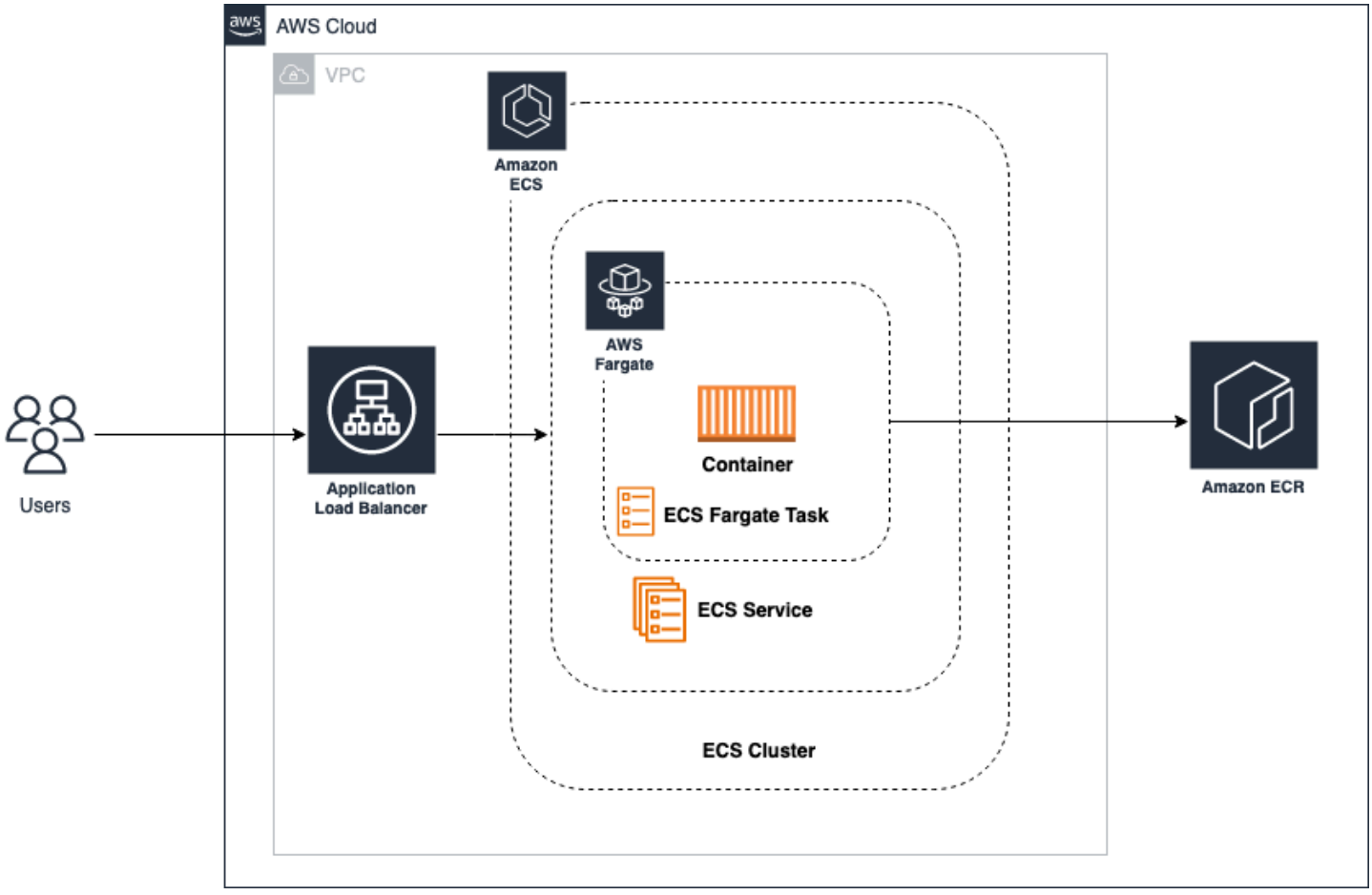
表 4: Amazon ECS デプロイ機能

機能	説明
プロビジョニング	Amazon ECS は、スケーリングポリシーと Amazon ECS 設定に基づいて、新しいアプリケーションコンテナインスタンスとコンピューティングリソースをプロビジョニングします。Load Balancer などのインフラストラクチャリソースは、Amazon ECS の外部で作成する必要があります。 Amazon ECS で作成できるリソースのタイプの詳細については、「Amazon ECS の開始方法」を参照してください。
構成する	Amazon ECS は、コンテナ化されたアプリケーションを実行するために作成されたコン

機能	説明
	ピューティングリソースのカスタマイズと、アプリケーションコンテナのランタイム条件 (環境変数、公開ポート、予約メモリ/CPU など) をサポートしています。基盤となるコンピューティングリソースのカスタマイズは、Amazon EC2 インスタンスを使用する場合にのみ使用できます。 コンテナ化されたアプリケーションを実行するように Amazon ECS クラスターをカスタマイズする方法の詳細については、「クラスターの作成」を参照してください。
デプロイ	Amazon ECS は、コンテナ化されたアプリケーションに対していくつかのデプロイ戦略をサポートしています。 サポートされるデプロイプロセスのタイプの詳細については、「Amazon ECS デプロイタイプ」を参照してください。
スケール	Amazon ECS を自動スケーリングポリシーで使用して、Amazon ECS クラスターで実行されているコンテナの数を自動的に調整できます。 Amazon ECS でコンテナ化されたアプリケーションの自動スケーリングの設定の詳細については、「Service Auto Scaling」を参照してください。

機能	説明
モニタリング	Amazon ECS は、CloudWatch によるコンピューティングリソースとアプリケーションコンテナのモニタリングをサポートしています。 Amazon ECS が提供するモニタリング機能のタイプの詳細については、「Amazon ECS のモニタリング」を参照してください。

次の図は、シンプルなコンテナ化されたアプリケーションの管理に使用されている Amazon ECS を示しています。この例では、インフラストラクチャコンポーネントは Amazon ECS の外部に作成され、Amazon ECS はクラスター上のアプリケーションコンテナのデプロイとオペレーションを管理するために使用されます。



Amazon ECS のユースケース

Note

- アプリケーションインフラストラクチャ (Amazon Elastic Container Registry (Amazon ECR) リポジトリ、Amazon ECS 設定、ロードバランサーを含む) は、Amazon ECS デプロイの外部でプロビジョニングおよび管理されます。
- Amazon ECS は、Amazon ECS サービス内で実行されているアプリケーションコンテナのデプロイを、Amazon ECR などのコンテナレジストリから取得されるタスクとして管理します。

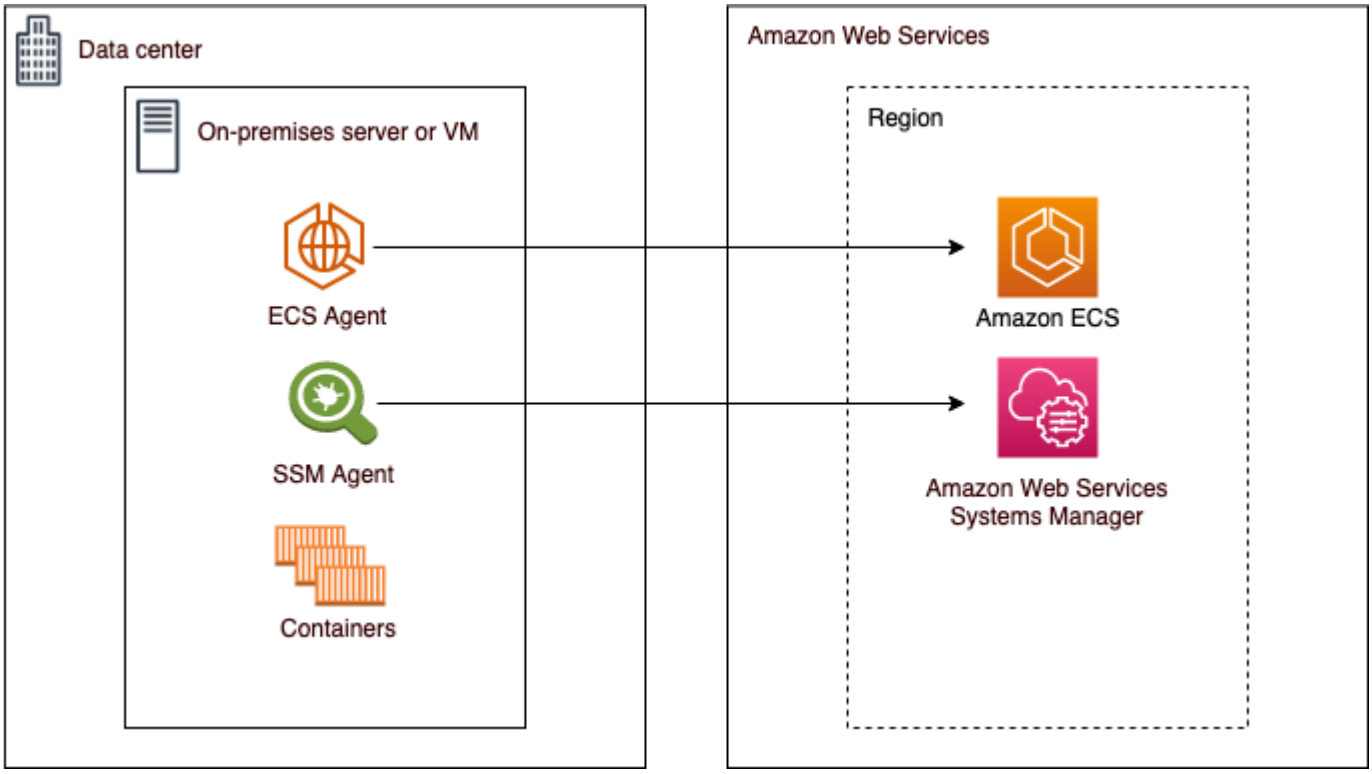
Amazon ECS は、Linux や Windows などの複数のコンテナインスタンスタイプと、Amazon ECS Anywhere を使用したオンプレミス仮想マシン (VM) などの外部インスタンスタイプをサポートしています。

Amazon ECS Anywhere

Amazon ECS Anywhere では、オンプレミスでも他のクラウド環境でも、Amazon ECS タスクをどこでも実行できます。Amazon ECS Anywhere を使用すると、一貫した運用エクスペリエンスを維持しながら、ハイブリッドインフラストラクチャ全体でコンテナ化されたアプリケーションを簡単にデプロイおよび管理できます。このサービスは、Amazon ECS プラットフォームを、オンプレミスのデータセンター、リモートオフィス、その他のクラウド環境など、あらゆる環境に拡張することで機能します。これにより、基盤となるインフラストラクチャについて心配することなく、使い慣れた同じ Amazon ECS APIs とツールを使用して、すべての環境にコンテナをデプロイおよび管理できます。

Amazon ECS Anywhere は Amazon ECS エージェントを使用してコンテナのデプロイとライフサイクルを管理し、 で使用しているのと同じ Amazon ECS タスク定義と設定ファイルを使用できます AWS クラウド。これにより、ハイブリッドインフラストラクチャ全体にコンテナをデプロイして管理するプロセスを簡素化し、手動の設定と管理に必要な時間と労力を削減できます。

Amazon ECS Anywhere では、IAM、 AWS CloudFormation、Amazon ECR などの他の AWS のサービスを活用して、コンテナ化されたアプリケーションを管理することもできます。これにより、アプリケーションの安全性、コンプライアンス、および他の AWS サービスとの統合を確保できます。



Amazon ECS Anywhere architecture

での Amazon Elastic Container Service AWS Outposts

の Amazon ECS AWS Outposts は、 で使用しているのと同じ APIsとツールを使用して、オンプレミスで Amazon ECS タスクを実行できるようにするフルマネージド AWS サービスです AWS クラウド。Amazon ECS on を使用すると AWS Outposts、オンプレミスでもクラウドでも、コンテナ化されたアプリケーションを一貫性のある使い慣れた方法でデプロイおよび管理できます。 AWS Outposts は、AWS インフラストラクチャ、サービス、APIs、ツールをオンプレミス環境に拡張するフルマネージドサービスです。Amazon ECS をオンにすると AWS Outposts、基盤となるインフラストラクチャについて心配することなく、組織専用のハードウェアで Amazon ECS タスクを実行できます。これにより、アプリケーションを安全かつ準拠した方法でデプロイできると同時に、クラウドの柔軟性とスケーラビリティを活用できます。

の Amazon ECS AWS Outposts は、一連の AWS サービスと APIs をオンプレミス環境にデプロイすることで機能します。これにより、専用のハードウェアで Amazon ECS タスクを実行できます。これには、コンテナのデプロイとライフサイクルを管理する Amazon ECS エージェントと、コンテナ化されたアプリケーションを実行するための安全で準拠した環境を提供する AWS Outposts インフラストラクチャが含まれます。Amazon ECS をオンにすると AWS Outposts、 で使用するのと同じ Amazon ECS APIs とツールを使用できるため AWS クラウド、コンテナ化されたアプリケーションを一貫性のある使い慣れた方法で簡単にデプロイおよび管理できます。これにより、手動の設定と管

理に必要な時間と労力が削減され、ハイブリッドインフラストラクチャ全体の一貫性と信頼性が向上します。の Amazon ECS は、IAM、 AWS CloudFormation、Amazon ECR などの他の AWS サービスと AWS Outposts 統合して、コンテナ化されたアプリケーションを管理します。これにより、アプリケーションの安全性、コンプライアンス、および他の AWS サービスとの統合を確保できます。

アマゾン エラスティックKubernetesサービス

Amazon Elastic Kubernetes Service (Amazon EKS) は、AWS での Kubernetes クラスターの構築、保護、運用、保守のプロセスを簡素化する、フルマネージドの認定された Kubernetes 準拠サービスです。Amazon EKS は、CloudWatch、Auto Scaling Groups、IAM などのコア AWS サービスと統合され、コンテナ化されたアプリケーションをモニタリング、スケーリング、ロードバランシングするためのシームレスなエクスペリエンスを提供します。

Amazon EKS は、Kubernetes ワークロード用のスケーラブルで可用性の高いコントロールプレーンを提供します。Amazon ECS と同様に、Amazon EKS でアプリケーションを実行する場合、Amazon EC2 インスタンスまたは を使用して、コンテナの基盤となるコンピューティング能力を提供できます AWS Fargate。

Amazon VPC Lattice は、AWS ネットワークインフラストラクチャに直接構築されたフルマネージド型のアプリケーションネットワークサービスで、複数のアカウントおよび仮想プライベートクラウド (VPCs) でサービスを接続、保護、モニタリングするために使用できます。Amazon EKS では、Kubernetes Gateway API の実装である AWS Gateway API Controller を使用して VPC Lattice を活用できます。VPC Lattice を使用すると、シンプルで一貫性のある方法で、標準の Kubernetes セマンティクスとのクラスター間接続を設定できます。

Amazon EKS は、次のいずれかのデプロイオプションで使用できます。

- Amazon EKS Distro – Amazon EKS がクラウドにデプロイしている同じオープンソースの Kubernetes ソフトウェアおよび依存関係のディストリビューションです。Amazon EKS Distro は Amazon EKS と同じ Kubernetes バージョンのリリースサイクルに従っており、オープンソースプロジェクトとして提供されています。詳細については、Amazon EKS Distro を参照してください。
- Amazon EKS on AWS Outposts – オンプレミス施設でネイティブ AWS サービス、インフラストラクチャ、運用モデル AWS Outposts を有効にします。Amazon EKS では AWS Outposts、拡張クラスターまたはローカルクラスターの実行を選択できます。拡張クラスターでは、Kubernetes コントロールプレーンは で実行 AWS リージョン され、ノードは で実行されます AWS Outposts。ローカルクラスターでは、Kubernetes コントロールプレーンとノードの両方を含む AWS Outposts Kubernetes クラスター全体がローカルで実行されます。

- Amazon EKS Anywhere — Amazon EKS Anywhere は、オンプレミスの Kubernetes クラスターを簡単に作成および運用できる Amazon EKS のデプロイオプションです。Amazon EKSと Amazon EKS Anywhere はどちらも Amazon EKS Distro上に構築されています。Amazon EKS Anywhere の詳細については、「Amazon Amazon EKS Anywhere Amazon EKS Anywhere の概要」、Amazon EKS Anywhere と Amazon EKS の比較」を参照してください。

Kubernetes クラスターに使用するデプロイオプションを選択する場合は、以下の点を考慮してください。

表 5: Kubernetes デプロイ機能

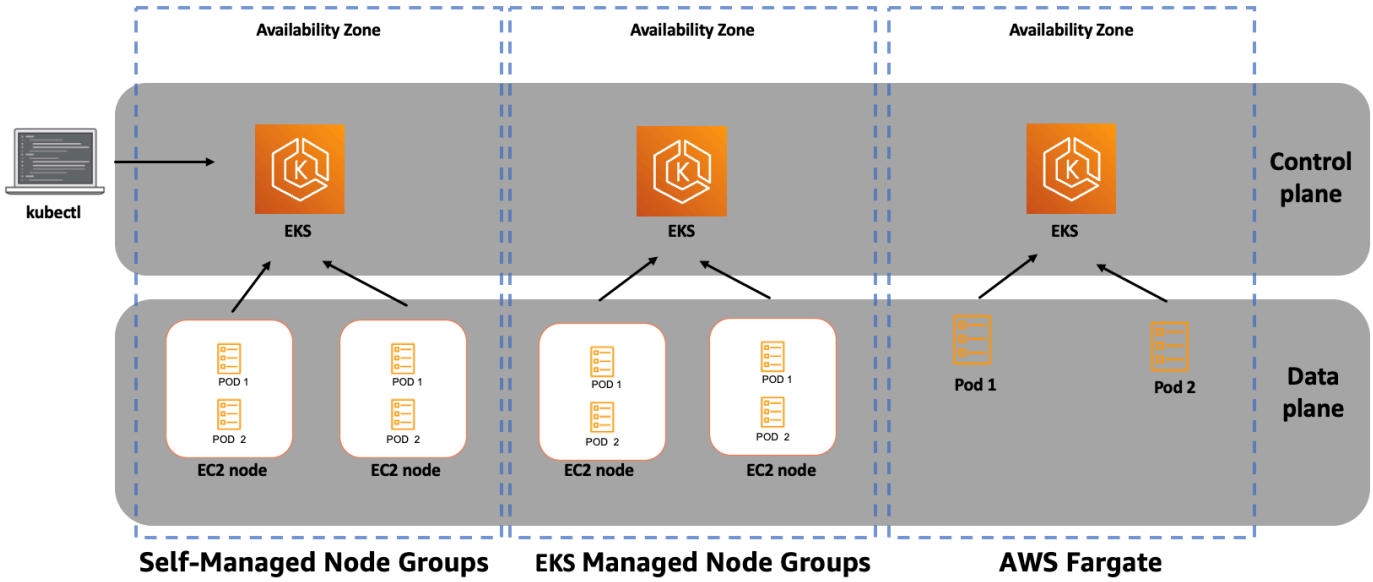
機能	Amazon EKS	での Amazon EKS AWS Outposts	Amazon EKS Anywhere	Amazon EKS Distro
ハードウェア	AWS 提供	AWS 提供	お客様が提供する	お客様が提供する
デプロイの場所	AWS クラウド	お客様のデータセンター	お客様のデータセンター	お客様のデータセンター
Kubernetes コントロールプレーンの場所	AWS クラウド	AWS クラウドまたはデータセンター	お客様のデータセンター	お客様のデータセンター
Kubernetes データプレーンの場所	AWS クラウド	お客様のデータセンター	お客様のデータセンター	お客様のデータセンター
サポート	AWS サポート	AWS サポート	AWS サポート	OSS コミュニティのサポート

表 6: Amazon EKS デプロイ機能

機能	説明
プロビジョニング	Amazon EKS は、コンテナ化されたアプリケーションをサポートするために特定のリソースをプロビジョニングします。 • 必要に応じてロードバランサー • コンピューティングリソースまたはワーカー (Amazon EKS は Windows と Linux をサポート) • アプリケーションコンテナインスタンス、またはポッド Amazon EKS クラスターのプロビジョニングの詳細については、「Amazon EKS の開始方法」を参照してください。
構成する	Amazon EC2 インスタンスを使用してコンピューティング能力を提供する場合、Amazon EKS はコンピューティングリソース (ワーカー) のカスタマイズをサポートします。Amazon EKS は、アプリケーションコンテナ (ポッド) のランタイム条件のカスタマイズもサポートしています。 詳細については、ワーカーノードと Fargate Pod Configuration のドキュメントを参照してください。
デプロイ	Amazon EKS は、Kubernetes と同じデプロイ戦略をサポートしています。詳細については、「Writing a Kubernetes Deployment Spec -> Strategy」を参照してください。
スケール	Amazon EKS は、Kubernetes Cluster Autoscaler を使用してワーカーをスケーリングし、Kubernetes Horizontal Pod Autoscaler

機能	説明
	と Kubernetes Vertical Pod Autoscaler を使用してポッドをスケーリングします。Amazon EKS は、オープンソースの柔軟で高性能な Kubernetes クラスターオートスケーラーである Karpenter もサポートしており、アプリケーションの負荷の変化に応じて適切なサイズのコンピューティングリソースを迅速に起動することで、アプリケーションの可用性とクラスター効率を向上させます。
モニタリング	Amazon EKS コントロールプレーンログは、監査と診断情報を CloudWatch Logs に直接提供します。Amazon EKS コントロールプレーンは と統合 AWS CloudTrail して、Amazon EKS で実行されたアクションを記録します。 詳細については、「Amazon EKS のログ記録とモニタリング」を参照してください。

Amazon EKS を使用すると、組織はオープンソースの Kubernetes ツールとプラグインを活用でき、既存の Kubernetes 環境で AWS に移行する組織に適しています。次の図は、一般的なコンテナ化されたアプリケーションの管理に使用されている Amazon EKS を示しています。



Amazon EKS use case

Amazon EKS Anywhere

Amazon EKS Anywhere では、独自のインフラストラクチャで Kubernetes クラスターを作成して運用できます。Amazon EKS Anywhere は、Amazon EKS Distro の長所に基づいて構築されており、最新のパッチが適用されたオープンソースソフトウェアを提供するため、セルフマネージド型の Kubernetes 製品よりも信頼性の高いオンプレミスの Kubernetes 環境を構築できます。

Amazon EKS Anywhere は、選択したプロバイダーに対してオンプレミスで Kubernetes クラスターを作成します。サポートされているプロバイダーには、ベアメタル (Tinkerbell 経由) 、CloudStack、vSphere などがあります。そのクラスターを管理するには、Ubuntu または Mac 管理マシンからクラスターの作成および削除コマンドを実行します。

AWS App Runner

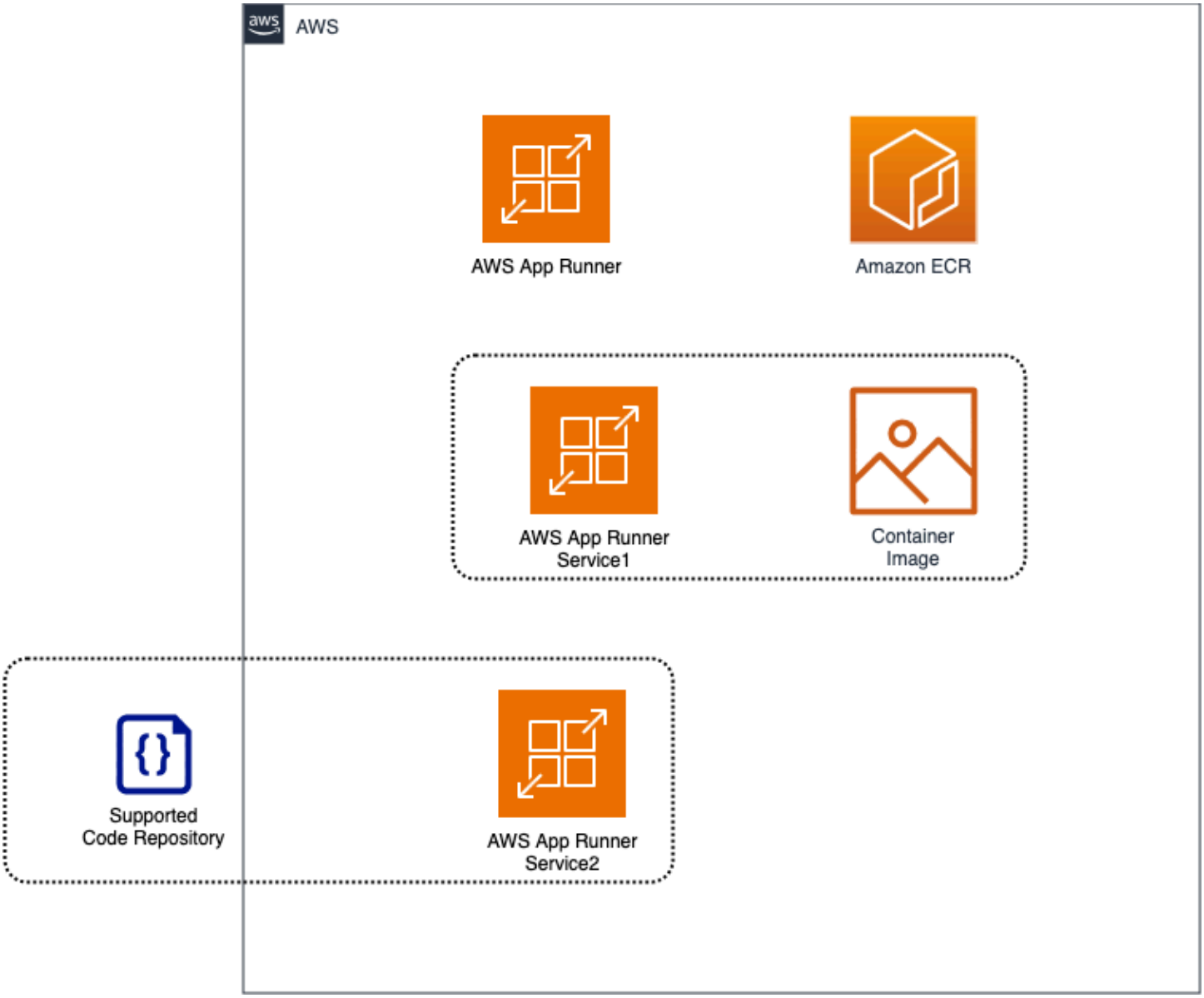
AWS App Runner はフルマネージド型のコンテナアプリケーションサービスで、コンテナ化されたウェブアプリケーションと API サービスを構築、デプロイ、および実行できます。インフラストラクチャやコンテナに関する経験は必要ありません。App Runner は、コードまたはイメージリポジトリに直接接続します。フルマネージド型のオペレーション、高パフォーマンス、スケーラビリティ、セキュリティを備えた自動統合および配信パイプラインを提供します。

App Runner は、リポジトリからソースコードまたはソースイメージを取得し、 で実行中のウェブサービスを作成して維持します AWS クラウド。通常、サービスを作成するには、App Runner アク

ション `CreateService`を 1 つだけ呼び出す必要があります。ソースイメージリポジトリでは、App Runner がウェブサービスを実行するためにデプロイできるready-to-useコンテナイメージを提供します。ソースコードリポジトリでは、ウェブサービスを構築して実行するためのコードと手順を提供し、特定のランタイム環境をターゲットにします。App Runner は、複数のプログラミングプラットフォームをサポートしています。各プラットフォームには、プラットフォームのメジャーバージョン用の 1 つ以上のマネージドランタイムがあります。App Runner は、コンテナイメージだけでなく、Node.js や Python などのランタイムやウェブフレームワークもサポートしています。App Runner は、アプリケーションに送信された同時リクエストの数をモニタリングし、リクエストボリュームに基づいて追加のインスタンスを自動的に追加します。アプリケーションが受信リクエストを受信しない場合、App Runner はコンテナをプロビジョニングされたインスタンスにスケールダウンします。プロビジョニングされたインスタンスは CPU スロットリングされたインスタンスで、数ミリ秒以内に受信リクエストを処理できます。

現時点では、App Runner は GitHub リポジトリからソースコードを取得したり、 の Amazon ECR からソースイメージを取得したりできます AWS アカウント。

次の図は、App Runner サービスアーキテクチャの概要を示しています。この図には、2 つのサンプルサービスがあります。1 つは GitHub からソースコードをデプロイし、もう 1 つは Amazon ECR からソースイメージをデプロイします。



App Runner use case

App Runner は、HTTP プロトコルと HTTPS プロトコルを使用するフロントエンドウェブアプリケーションとバックエンドウェブアプリケーションの両方を含む、フルスタック開発をサポートします。これらのアプリケーションには、API サービス、バックエンドウェブサービス、ウェブサイトが含まれます。App Runner は、コンテナイメージだけでなく、Node.js や Python などのランタイムやウェブフレームワークもサポートしています。

Amazon Lightsail

Amazon Lightsail は、小規模企業、スタートアップ、個人がクラウドにアプリケーションを簡単にデプロイおよび管理できるようにする、シンプルで費用対効果の高いクラウドサービスです。基盤とな

るインフラストラクチャ管理の多くを抽象化し、クラウドでのアプリケーションの起動と実行を容易にする、使いやすいインターフェイスを提供します。を使用するとLightsail、仮想プライベートサーバー (VPS)、データベース、ストレージインスタンスをすばやくデプロイおよび管理できます。このサービスは、WordPress、Drupal、Joomla などのさまざまなワークロード用に最適化された事前設定されたインスタンスを提供します。これにより、環境のセットアップと設定に必要な時間と労力を削減できます。 は、統合されたロードバランサーと自動スケーリングLightsailも提供するため、手動による介入なしにトラフィック需要の変化を処理できます。このサービスでは、モニタリングとアラートも提供されるため、アプリケーションの状態とパフォーマンスを常に把握できます。

の主な利点の 1 つは、そのシンプルさと使いやすさLightsailです。このサービスは、最小限のクラウドコンピューティングエクスペリエンスでユーザーにアクセスできるように設計されているため、クラウドですぐに開始したい中小企業や個人に適しています。さらに、 Lightsailは費用対効果が高く、コンピューティング、ストレージ、データ転送などの予測可能な料金です。

Amazon Lightsail コンテナ

Amazon Lightsail コンテナは、コンテナ化されたアプリケーションをクラウドに簡単にデプロイおよび管理できるフルマネージド型のコンテナサービス AWS です。Docker や Kubernetes などの一般的なコンテナ管理ツールを使用してコンテナを起動および実行するシンプルで費用対効果の高い方法を提供します。

Lightsail コンテナは、コンテナ化されたアプリケーションを構築、テスト、デプロイするための統合環境を提供します。基盤となるインフラストラクチャ管理の大部分を抽象化する使いやすいインターフェイスを提供することで、コンテナのデプロイと管理のプロセスを簡素化します。

Lightsail コンテナを使用すると、コンテナ化されたアプリケーションを数回クリックするだけで VPC にデプロイできます。このサービスは、Node.js、Python、Ruby、Java などの一般的なプログラミング言語用に事前設定されたコンテナイメージを提供します。これにより、コンテナ環境のセットアップと設定に必要な時間と労力を削減できます。

Lightsail コンテナは、コンテナインスタンス全体にトラフィックを自動的に分散できる統合されたロードバランサーも提供し、アプリケーションの可用性とスケーラビリティを向上させます。さらに、このサービスではコンテナインスタンスの自動スケーリングが提供されるため、手動による介入なしにトラフィック需要の変化を処理できます。

Lightsail コンテナを使用すると、組み込みのメトリクスとログを使用して、コンテナ化されたアプリケーションのパフォーマンスをモニタリングできます。また、Amazon S3、Amazon RDS、 などの他の AWS サービスと統合して AWS CodePipeline、コンテナ化されたアプリケーション用に完全に自動化された統合 CI/CD パイプラインを作成することもできます。

Red Hat OpenShift Service on AWS

Red Hat OpenShift Service on AWS (ROSA) は、AWS マネジメントコンソールから利用できるマネージドサービスです。ROSA を使用すると、Red Hat OpenShift ユーザーとして、AWS でコンテナ化されたアプリケーションを構築、スケーリング、管理できます。ROSA を使用して、Red Hat OpenShift APIs とツールを使用して Kubernetes クラスターを作成し、AWS のサービスの全範囲と深さにアクセスできます。ROSA は、オンプレミスの Red Hat OpenShift ワークロードの AWS への移行を合理化し、他の AWS サービスとの密接な統合を提供します。また、Red Hat OpenShift のライセンス、請求、サポートに AWS から直接アクセスすることもできます。

各 ROSA クラスターには、フルマネージド型のコントロールプレーンとコンピューティングノードが付属しています。インストール、管理、メンテナンス、アップグレードは、Red Hat と Amazon の共同サポートにより Red Hat SRE によって実行されます。クラスターサービス (ログ記録、メトリクス、モニタリングなど) も利用できます。ROSA では、Red Hat Enterprise Linux CoreOS (RHCOS) ワーカーのみがサポートされています。

ROSA は、さまざまな AWS コンピューティング、ストレージ、データベース、分析、機械学習、ネットワーキング、モバイル、およびさまざまなアプリケーションサービスと統合され、世界中のオンデマンドでスケールする AWS サービスの堅牢なポートフォリオからメリットを得ることができます。これらの AWS ネイティブサービスに直接アクセスして、同じ管理インターフェイスを介してサービスをすばやくデプロイおよびスケーリングできます。

AWS Local Zones

AWS Local Zone は、ユーザーに近い地理的な場所 AWS リージョン にある の拡張です。Local Zones は、インターネットへの独自の接続を持ち、 AWS Direct Connectをサポートします。Local Zones で作成したリソースは、低いレイテンシーの通信をローカルユーザーに提供できます。ローカルゾーンは、リージョンコードに続けて、その場所を示す識別子 (us-west-2-lax-1a など) で表されます。

Amazon ECS は、低レイテンシーまたはローカルデータ処理が必要な場合に、ローカルゾーンを使用するワークロードをサポートします。Amazon ECS コントロールプレーンは常に で実行されます AWS リージョン。

Amazon EKS では、Local Zones の特定のリソースをサポートしています。これには、セルフマネージド Amazon EC2 ノード、Amazon EBS ボリューム、Application Load Balancer が含まれます。Amazon EKS が管理する Kubernetes コントロールプレーンは、常に AWS リージョンで実行されます。Amazon EKS が管理する Kubernetes コントロールプレーンは、Local Zone で実行できま

せん。 Local Zones は VPC 内のサブネットとして表示されるため、Kubernetes は Local Zone のリソースをそのサブネットの一部として認識します。

AWS Wavelength

AWS Wavelength は、5G-connectedユーザーとデバイスに近いワークロードをデプロイできる AWS インフラストラクチャです。Wavelength を使用して、Amazon EC2 インスタンス、Amazon EKS クラスター、および AWS Marketplace で利用可能な一連のサポートされているパートナーソリューションをデプロイできます。Wavelength Zones は、通信プロバイダーのネットワーク内で論理的に隔離されたデータセンターであり、冗長性、低レイテンシー、高スループットの接続を介して AWS リージョンに接続されます。

Wavelength の主な機能には、Wavelength Zones で Amazon EC2 インスタンス、Amazon EBS ボリューム、Amazon VPC サブネット、キャリアゲートウェイを作成する機能が含まれます。Amazon EC2 Auto Scaling、Amazon EKS クラスター、Amazon ECS クラスター、Amazon EC2 Systems Manager、Amazon CloudWatch、Application Load Balancer など、Amazon EC2、Amazon EBS AWS CloudTrail AWS CloudFormation、Amazon VPC をオーケストレーションまたは操作するサービスを使用することもできます。 Amazon CloudWatch Application Load Balancer Wavelength サービスは、Amazon DynamoDB や Amazon Relational Database Service (Amazon RDS) などのサービスに簡単にアクセスできるように、AWS リージョンへの信頼性の高い高帯域幅接続を介して接続された VPC の一部です。

追加のデプロイサービス

Amazon Simple Storage Service (Amazon S3) は、静的コンテンツおよびシングルページアプリケーション (SPA) のウェブサーバーとして使用できます。Amazon CloudFront と組み合わせることで、静的コンテンツ配信のパフォーマンスを向上させることができます。Amazon S3 を使用すると、静的コンテンツを簡単かつ強力にデプロイおよび更新できます。このアプローチの詳細については、ホワイトペーパーの「静的ウェブサイトのホスティング AWS」を参照してください。

AWS Proton

AWS Proton は、マイクロサービスとコンテナベースのアプリケーションのデプロイと管理のプロセスを簡素化および自動化するフルマネージドサービスです。一般的な DevOps ツールやサービスと統合される統一された一貫したデプロイエクスペリエンスを提供し、アプリケーション開発の管理と合理化を容易にします。Proton を使用すると、デベロッパーはインフラストラクチャ、コード、パイプラインなどのアプリケーションコンポーネントを再利用可能なテンプレートとして定義して作

成できます。これらのテンプレートは、開発、テスト、本番稼働など、複数の環境を作成するために使用でき、チームや組織間で共有できます。このアプローチは、マイクロサービスやコンテナベースのアプリケーションのデプロイと管理の複雑さを軽減するのに役立ちます。これは時間がかかり、エラーが発生しやすくなります。

AWS Proton には、ウェブアプリケーション、APIs、データベースなどの一般的なタイプのマイクロサービス用の構築済みテンプレートが用意されており、特定のニーズに合わせてカスタマイズできます。また、AWS CodePipeline、AWS CodeCommit、AWS CodeBuild などの一般的な DevOps ツールと統合して、継続的な統合とデプロイ (CI/CD) ワークフローを可能にします。

AWS Proton を使用することで、デベロッパーはマイクロサービスとコンテナベースのアプリケーションのデプロイと管理に必要な時間と労力を削減できます。このアプローチにより、チームはデプロイと管理プロセスに時間を費やすのではなく、アプリケーションの開発と改善に集中できます。

AWS App2Container

AWS App2Container は、Java および .NET ウェブアプリケーションをコンテナ形式に移行およびモダナイズするためのコマンドラインツールです。App2Container は、ベアメタル、仮想マシン、Amazon EC2 インスタンス、またはクラウドで実行されているアプリケーションのインベントリを分析して構築します。コンテナ化するアプリケーションを選択するだけで、App2Container はアプリケーションアーティファクトと識別された依存関係をコンテナイメージにパッケージ化し、ネットワークポートを設定し、ECS タスクと Kubernetes ポッド定義を生成します。App2Container は、仮想マシンで実行されているサポートされている ASP.NET および Java アプリケーションを識別し、環境内のすべてのアプリケーションを包括的にインベントリします。App2Container は、Linux、スタンドアロン、または JBoss、Apache Tomcat、Springboot、IBM Websphere、Oracle Weblogic などのアプリケーションサーバーで実行されている Windows または Java アプリケーションで IIS で実行されている ASP.NET ウェブアプリケーションをコンテナ化できます。

AWS Copilot

AWS Copilot は、AWS でコンテナ化されたアプリケーションをすばやく起動および管理するために使用できるコマンドラインインターフェイス (CLI) です。Amazon ECS、Fargate、App Runner でのアプリケーションの実行を簡素化します。AWS Copilot は現在、Linux、macOS、および Windows システムをサポートしています。Copilot を使用すると、負荷分散されたウェブサービスなどのサービスパターンを使用してインフラストラクチャをプロビジョニングし、テストや本番稼働などの複数の環境にデプロイし、自動デプロイに AWS CodePipeline リリースパイプラインを使用することもできます。

AWS Serverless Application Model

AWS Serverless Application Model (AWS SAM) は、サーバーレスアプリケーションを構築するためのオープンソースフレームワークです。関数、API、データベース、イベントソースマッピングを表現するための省略構文を提供します。リソースごとに数行で、必要なアプリケーションを定義し、YAML を使用してモデル化できます。デプロイ中、SAM は SAM 構文を AWS CloudFormation 構文に変換して拡張し、サーバーレスアプリケーションをより迅速に構築できるようにします。

AWS SAM CLI は、AWS でサーバーレスアプリケーションを簡単に開発、テスト、デプロイできるオープンソースのコマンドラインツールです。これは、AWS CloudFormation の拡張機能である AWS SAM 仕様を使用してサーバーレスアプリケーションを構築するためのコマンドラインインターフェイスです。

AWS SAM CLI を使用すると、デベロッパーはサーバーレスアプリケーションを AWS にデプロイする前にローカルで定義してテストできます。AWS Lambda と API Gateway をシミュレートするローカルテスト環境を提供し、デベロッパーがクラウドにデプロイする前にコードと設定をテストできるようにします。

AWS SAM CLI には、自動コードデプロイ、ログ記録、デバッグ機能など、さまざまな便利な機能も含まれています。これにより、デベロッパーは 1 つのコマンドでアプリケーションを構築、パッケージ化、デプロイできるため、サーバーレスアプリケーションのデプロイと管理に必要な時間と労力が削減されます。

さらに、AWS SAM CLI では、Node.js、Python、Java、.NET Core など、さまざまなプログラミング言語がサポートされています。これにより、開発者は好みのプログラミング言語とツールを使用してサーバーレスアプリケーションを構築およびデプロイできます。

AWS SAM CLI は、AWS CodePipeline や AWS CodeBuild などの他の AWS サービスと統合され、サーバーレスアプリケーションに完全に自動化され統合された CI/CD パイプラインを提供します。また、デベロッパーは Amazon S3、Amazon DynamoDB、Amazon SNS などの他の AWS サービスをサーバーレスアプリケーションの一部として使用できます。

AWS Cloud Development Kit (AWS CDK)

AWS Cloud Development Kit (AWS CDK) (AWS CDK) は、最新のプログラミング言語でクラウドインフラストラクチャをコードとして定義し、AWS CloudFormation を通じてデプロイするためのオープンソースのソフトウェア開発フレームワークです。AWS Cloud Development Kit (AWS CDK) は、一般的なプログラミング言語を使用してアプリケーションをモデル化することで、クラウド開発を加

速します。AWS CDK を使用すると、プログラミング言語の表現力が大幅に向上し、信頼性が高く、スケーラブルで、費用対効果の高いアプリケーションをクラウドで構築できます。

AWS CDK は、最新のプログラミング言語の全能力を活用して AWS インフラストラクチャをコードとして定義し、開発者中心のツールキットと考えてください。AWS CDK アプリケーションが実行されると、完全に形成された CloudFormation JSON/YAML テンプレートにコンパイルされ、プロビジョニングのために CloudFormation サービスに送信されます。AWS CDK は CloudFormation を活用しているため、安全なデプロイ、自動ロールバック、ドリフト検出など、CloudFormation が提供するすべての利点を享受できます。

このアプローチには、次の内容を含む多くの利点があります。

- AWS リソースに適切で安全なデフォルトを自動的に提供し、より少ないコードでより多くのインフラストラクチャを定義する高レベルのコンストラクトで構築します。
- パラメータ、条件、ループ、構成、継承などのプログラミングイディオムを使用して、AWS やその他の が提供する構成要素からシステム設計をモデル化します。
- インフラストラクチャ、アプリケーションコード、設定をすべて 1 か所に配置し、マイルストーンごとに完全なクラウドデプロイ可能なシステムを確保します。
- コードレビュー、ユニットテスト、ソース管理などのソフトウェアエンジニアリングプラクティスを採用して、インフラストラクチャをより堅牢にします。
- AWS Solutions Constructs は、AWS CDK のオープンソースライブラリ拡張機能です。AWS ソリューションコンストラクトは、AWS Well-Architected フレームワークによって確立されたベストプラクティスを使用して構築された、検証済みのマルチサービスアーキテクチャパターンのコレクションを提供します。

AWS Serverless Application Model と AWS CDK はどちらも AWS インフラストラクチャをコードとして抽象化するため、クラウドインフラストラクチャを簡単に定義できます。AWS SAM はサーバーレスのユースケースとアーキテクチャに特に焦点を当てており、インフラストラクチャをコンパクトで宣言的な JSON/YAML テンプレートで定義できます。AWS CDK は、すべての AWS のサービスに幅広く対応しており、最新のプログラミング言語でクラウドインフラストラクチャを定義できます。

Amazon EC2 Image Builder

EC2 Image Builder は、AWS またはオンプレミスで使用する VM およびコンテナイメージの構築、テスト、デプロイを簡素化します。VM とコンテナイメージup-to-dateに保つと、時間がかかり、リソースを大量に消費し、エラーが発生しやすくなります。現在、お客様は VMsを手動で更新してスナップショットを作成するか、イメージを維持するために自動化スクリプトを構築するチームを用意

しています。Image Builder は、シンプルなグラフィカルインターフェイス、組み込みのオートメーション、AWS が提供するセキュリティ設定を提供することで、イメージup-to-date状態に保ち、安全に保つ労力を大幅に削減します。Image Builder では、イメージを更新するための手動手順や、独自の自動化パイプラインを構築する必要はありません。Image Builder は、イメージの作成、保存、共有に使用される基盤となる AWS リソースのコストを除き、無料で提供されます。

EC2 Image Builder は、Amazon EC2、コンテナ、オンプレミスサーバーで使用するカスタムイメージの作成と管理のプロセスを簡素化することで、AWS でのデプロイを容易にします。このサービスは、イメージの作成と管理プロセスを合理化する自動ビルドパイプラインにより、カスタムイメージを作成および管理するためのシンプルで柔軟な方法を提供します。

EC2 Image Builder は、基盤となるインフラストラクチャ管理の大部分を抽象化する使いやすいインターフェイスを提供するため、開発者はカスタムイメージを簡単に作成および管理できます。EC2 Image Builder を使用すると、デベロッパーはイメージに含めるオペレーティングシステム、アプリケーション、パッケージを指定でき、更新、パッチ、セキュリティ修正など、イメージの構築とテストのプロセスを自動化します。自動ビルドパイプラインにより、デベロッパーはイメージの作成と管理プロセスを合理化し、手動イメージの作成とテストに必要な時間と労力を削減できます。これにより、一貫性の向上、エラーの軽減、イメージup-to-date、安全性、コンプライアンスの確保に役立ちます。

EC2 Image Builder の利点を以下に示します。

- イメージ作成の簡素化: EC2 Image Builder は、Amazon EC2、コンテナ、オンプレミスサーバーで使用するカスタムイメージを作成するためのシンプルで柔軟な方法を提供します。これにより、カスタムイメージの作成と保守に必要な時間と労力が削減され、アプリケーションの開発やテストなど、デプロイの他の側面に集中できます。
- 自動イメージビルドパイプライン: EC2 Image Builder は、カスタムイメージの構築、テスト、デプロイのための自動パイプラインを提供します。これは、イメージの作成と管理プロセスを合理化するのに役立ちます。これにより、イメージがup-to-date、安全、準拠していることが保証され、手動イメージの作成とテストに必要な時間と労力が削減されます。
- AWS サービスとの統合: EC2 Image Builder は、Amazon Elastic Container Registry (ECR) や Amazon Elastic Kubernetes Service (EKS) などの他の AWS サービスと統合され、コンテナで使用するカスタムイメージを構築できます。これにより、コンテナの構築とデプロイのプロセスを合理化し、アプリケーション、ライブラリ、設定を含むカスタムイメージを構築できます。
- 柔軟なイメージ作成: EC2 Image Builder では、カスタムイメージを柔軟に作成できるため、イメージに含めるオペレーティングシステム、アプリケーション、パッケージを指定できます。これにより、イメージが特定のユースケースや要件に合わせて調整され、デプロイ中のエラーや非互換性のリスクが軽減されます。

- イメージのセキュリティとコンプライアンスの向上: EC2 Image Builder を使用すると、脆弱性スキャンやコンプライアンススキャンなどのイメージテストを自動化して、イメージの安全性とコンプライアンスを確保できます。これにより、セキュリティ侵害のリスクを軽減し、コンプライアンスを向上させ、アプリケーションを自信を持ってデプロイできます。

デプロイ戦略

アプリケーションコードを更新し、インフラストラクチャをサポートする適切なツールを選択することに加えて、適切なデプロイプロセスを実装することは、完全で適切に機能するデプロイソリューションの重要な部分です。アプリケーションの更新に選択するデプロイプロセスは、制御、速度、コスト、リスク許容度、およびその他の要因の望ましいバランスによって異なります。

各 AWS デプロイサービスは、多くのデプロイ戦略をサポートしています。このセクションでは、デプロイソリューションで使用できる汎用デプロイ戦略の概要を説明します。

プリベーキングとブートストラップの AMIs

アプリケーションが Amazon EC2 インスタンスへのアプリケーションのカスタマイズやデプロイに大きく依存している場合は、ブートストラップとプリベーキングのプラクティスを通じてデプロイを最適化できます。

Amazon EC2 インスタンスが起動されるたびにアプリケーション、依存関係、またはカスタマイズをインストールすることを、インスタンスのブートストラップと呼びます。複雑なアプリケーションや大規模なダウンロードが必要な場合、デプロイやスケーリングイベントが遅くなる可能性があります。

Amazon マシンイメージ (AMI) は、インスタンスを起動するために必要な情報 (オペレーティングシステム、ストレージボリューム、アクセス許可、ソフトウェアパッケージなど) を提供します。単一の AMI から複数の同一のインスタンスを起動できます。EC2 インスタンスが起動されるたびに、テンプレートとして使用する AMI を選択します。プリベーキングは、アプリケーションアーティファクトの大部分を AMI に埋め込むプロセスです。

アプリケーションコンポーネントを AMI にプリベーキングすると、Amazon EC2 インスタンスの起動と運用にかかる時間を短縮できます。プリベーキングとブートストラップのプラクティスは、デプロイプロセス中に組み合わせて、現在の環境にカスタマイズされた新しいインスタンスをすばやく作成できます。

ブルー/グリーンデプロイ

ブルー/グリーンデプロイは、2 つの異なる同一の環境を作成するデプロイ戦略です。1 つの環境 (青) が現在のアプリケーションバージョンを実行し、1 つの環境 (緑) が新しいアプリケーションバージョンを実行しています。Blue/Green デプロイ戦略を使用すると、デプロイが失敗した場合にロー

ルバックプロセスを簡素化することで、アプリケーションの可用性が向上し、デプロイのリスクが軽減されます。グリーン環境でのテストが完了すると、ライブアプリケーショントラフィックはグリーン環境に転送され、ブルー環境は廃止されます。

多くの AWS デプロイサービスは、Elastic Beanstalk、OpsWorks、CloudFormation、CodeDeploy、Amazon ECS などのブルー/グリーンデプロイ戦略をサポートしています。アプリケーションのブルー/グリーンデプロイプロセスを実装するための詳細と戦略については、「AWS でのブルー/グリーンデプロイ」を参照してください。

ローリングデプロイ

ローリングデプロイは、アプリケーションが実行されているインフラストラクチャを完全に置き換えることで、アプリケーションの以前のバージョンをアプリケーションの新しいバージョンに徐々に置き換えるデプロイ戦略です。例えば、Amazon ECS のローリングデプロイでは、アプリケーションの以前のバージョンを実行しているコンテナは、アプリケーションの新しいバージョンを実行しているコンテナに one-by-one置き換えられます。

ローリングデプロイは、通常、ブルー/グリーンデプロイよりも高速ですが、ブルー/グリーンデプロイとは異なり、ローリングデプロイでは、古いアプリケーションバージョンと新しいアプリケーションバージョンの間に環境分離はありません。これにより、ローリングデプロイをより迅速に完了できますが、デプロイが失敗した場合のロールバックのプロセスも複雑になります。

ローリングデプロイ戦略は、ほとんどのデプロイソリューションで使用できます。CloudFormation を使用したローリングデプロイの詳細については、CloudFormation 更新ポリシーを参照してください。Amazon ECS を使用したローリングデプロイの詳細については、Amazon ECS を使用したローリング更新を参照してください。Elastic Beanstalk を使用したローリングデプロイの詳細については、Elastic Beanstalk を使用したローリング環境設定の更新を参照してください。OpsWorks を使用したローリングデプロイの詳細については、 でのローリングデプロイの使用 AWS OpsWorksを参照してください。 CloudFormation

Canary デプロイ

Canary デプロイは、リスクがより高い Blue/Green デプロイ戦略の一種です。この戦略では、トラフィックをアプリケーションの新しいバージョンに 2 段階的に移行する段階的なアプローチを取ります。最初の増分はトラフィックのごく一部で、Canary グループと呼ばれます。このグループは新しいバージョンをテストするために使用されます。成功すると、トラフィックは 2 回目の増分で新しいバージョンに移行されます。

Canary デプロイは 2 つのステップで実装することも、直線的に実装することもできます。2 ステップのアプローチでは、新しいアプリケーションコードがデプロイされ、トライアル用に公開されます。承諾すると、環境の残りの部分または線形にロールアウトされます。線形アプローチでは、すべてのトラフィックが新しいリリースに流れるまで、アプリケーションの新しいバージョンへのトラフィックを段階的に増やします。

インプレースデプロイ

インプレースデプロイは、インフラストラクチャコンポーネントを置き換えることなくアプリケーションバージョンを更新するデプロイ戦略です。インプレースデプロイでは、各コンピューティングリソースのアプリケーションの以前のバージョンが停止し、最新のアプリケーションがインストールされ、アプリケーションの新しいバージョンが起動および検証されます。これにより、基盤となるインフラストラクチャへの障害を最小限に抑えながら、アプリケーションのデプロイを続行できます。

インプレースデプロイでは、新しいインフラストラクチャを作成せずにアプリケーションをデプロイできます。ただし、アプリケーションの可用性は、これらのデプロイ中に影響を受ける可能性があります。このアプローチにより、新しいリソースの作成に関連するインフラストラクチャコストと管理オーバーヘッドも最小限に抑えられます。

CodeDeploy でのインプレースデプロイ戦略の使用の詳細については、「インプレースデプロイの概要」を参照してください。

デプロイサービスの組み合わせ

AWS には「1 つのサイズがすべてに適合」なデプロイソリューションはありません。デプロイソリューションを設計する場合、どの AWS サービスが最も適切かを決めることができるため、アプリケーションのタイプを考慮することが重要です。アプリケーションのプロビジョニング、設定、デプロイ、スケーリング、モニタリングを行う完全な機能を提供するには、多くの場合、複数のデプロイサービスを組み合わせる必要があります。

AWS のアプリケーションに共通するパターンは、CloudFormation (およびその拡張機能) を使用して汎用インフラストラクチャを管理し、より特殊なデプロイソリューションを使用してアプリケーションの更新を管理することです。コンテナ化されたアプリケーションの場合、CloudFormation を使用してアプリケーションインフラストラクチャを作成し、Amazon ECS と Amazon EKS を使用してコンテナのプロビジョニング、デプロイ、モニタリングを行うことができます。

AWS デプロイサービスは、サードパーティーのデプロイサービスと組み合わせることもできます。これにより、組織は AWS デプロイサービスを既存の CI/CD パイプラインまたはインフラストラク

チャ管理ソリューションに簡単に統合できます。例えば、OpsWorks を使用してオンプレミスノードと AWS ノード間で設定を同期し、CodeDeploy を完全なパイプラインの一部として多数のサードパーティー CI/CD サービスで使用できます。

結論

AWS には、インフラストラクチャのプロビジョニングとアプリケーションのデプロイを簡素化および自動化するためのツールが多数用意されています。各デプロイサービスは、アプリケーションを管理するためのさまざまな機能を提供します。デプロイアーキテクチャを正常に構築するには、アプリケーションと組織のニーズに照らして、各サービスの利用可能な機能を評価します。

寄稿者

本ドキュメントの寄稿者は次のとおりです。

- Manikandan Chandrasekaran、プリンシパル技術者
- Anil Nadiminti、シニアソリューションアーキテクト
- Bryant Bost、AWS ProServe コンサルタント

詳細情報

詳細については、次を参照してください。

- AWS ホワイトペーパーページ
- AWS での DevOps の概要 - デプロイ戦略

ドキュメントの改訂

このホワイトペーパーの更新に関する通知を受け取るには、RSS フィードにサブスクライブしてください。

変更	説明	日付
ホワイトペーパーの更新	最新のデプロイサービスと戦略のために 全体で更新	2024 年 5 月 31 日
マイナーな更新	ブルー/グリーンデプロイのセクションがわかりやすく改訂されました。	2021 年 4 月 8 日
ホワイトペーパーの更新	最新の サービスと機能で更新されました。	2020 年 6 月 3 日
初版発行	ホワイトペーパーの初回発行	2015 年 3 月 1 日

注意

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

© 2024 Amazon Web Services, Inc. またはその関連会社。All rights reserved.

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。