MODULE 1

クラウドコンピューティング

クラウド：コンピューティングリソース（サーバー、ストレージ、ネットワークなど）がインターネット経由で提供されるもの

**コンピューティング**

* コンピュータを使用してタスクを実行する
* 広い意味で、コンピュータ技術資源の開発と使用のあらゆる活動
* もともとコンピューティングという言葉は「計算」という意味であり、コンピュータは計算する人や機械を指すもの
* **クラウドコンピューティングにおけるコンピューティング**：ソフトウェアコンピューティングに関連する概念とオブジェクトすべてのプログラムのコンピューティングの成功に必要な処理性能、メモリ、ネットワーキング、ストレージ、その他のリソースを指すために使用される一般的な用語です。

**クラウドコンピューティング**

クラウドコンピューティングは、コンピューティングパワー、データベース、ストレージ、アプリケーション、その他のITリソースをインターネット経由でオンデマンドで提供し、使用した分だけ支払うことです。

**オンデマンド(On-demand)サービス**

経済的意味で、供給者中心ではなく需要に応じて製品やサービスが提供されるもので、消費者が望むものをすぐに提供する形のサービス

* 要件に応じてすぐに提供/供給する方法（注文型）
* 通信技術の発達で取引費用が減り、価格決定の主導権を需要者が持つ
* 顧客の要求があるとき、いつでもどこでも顧客中心でニーズを解決してくれること
* 供給ではなく、需要がすべてを決定するシステム
* 車両共有サービスウーバー、宿泊共有サービスエアアンドビー
* **クラウドコンピューティング展開モデル**

オンプレミス: 企業が独自にITインフラを所有、管理、運営する場合

この場合、社内ITチームはシステムの設計、構築、管理を担当します。まれに、サードパーティのベンダーが一部のコンポーネントを管理することもあります。ただし、ほとんどの場合、組織は場所、デバイス、ソフトウェア、およびアプリケーションを完全に所有および管理します。

**クラウドベースの展開**

* アプリケーションのすべての部分をクラウドで実行します。
* 既存のアプリケーションをクラウドに移行
* クラウドで新しいアプリケーションを設計および構築します。

**IT移行**

データまたはソフトウェアをあるシステムから別のシステムに移動する

**オンプレミス展開**

仮想化およびリソース管理ツールを使用してリソースをデプロイします。

アプリケーション管理と仮想化技術を使用してリソース使用率を向上させます。

オンプレミス展開はプライベートクラウド展開とも呼ばれます。このモデルでは、リソースは仮想化およびリソース管理ツールを使用してオンプレミスにデプロイされます。

たとえば、アプリケーションに必要なテクノロジのすべての要素がオンプレミスのデータセンターに保存されることがあります。このモデルは従来のITインフラストラクチャと非常によく似ていますが、アプリケーション管理と仮想化テクノロジが統合され、リソース使用率を向上させるのに役立ちます。

**ハイブリッド展開**

* クラウドベースのリソースをオンプレミスインフラストラクチャに接続する
* クラウドベースのリソースを従来のITアプリケーションと統合
* オンプレミスでよりよく維持されるレガシーアプリケーション
* 政府の規制に従って、ビジネスが特定のレコードをオンプレミスに保管する必要がある場合

**クラウドコンピューティングの利点**

* 先行コストを可変コストに置き換える
* データセンターの運用と保守に費用投資が不要
* 容量推定不要
* 規模の経済で得られる利点
* スピードと俊敏性を向上
* 数分で世界中に配布

MODULE 1 サプリメント

**コンピューティングリソース**

* コンピューティングタスクを実行するために必要なすべてのリソース
* ハードウェア/ソフトウェアで構成

**ハードウェアリソース**

* 中央処理装置（CPU）：データを処理してプログラムを実行する
* メモリ（RAM）：プログラムとデータを一時的に保存する
* ストレージデバイス（ハードディスク、SSDなど）：データを永久に保存

**ネットワークリソース**

ネットワーク接続：データの転送と通信に使用

**ソフトウェアリソース**

オペレーティングシステム：ハードウェアを管理し、アプリケーションを実行するために必要です。

アプリケーションとライブラリ：特定のタスクを実行するためのプログラムと機能を提供します。

**ITリソース**

「コンピューティングリソース」を含めながら、より広い範囲の技術とリソースを表します。

**ネットワークインフラ**

LAN（Local Area Network）、完全なプロトコル（IP）、スイッチ、ルーターなどのネットワーク機器を含みます。

**データベースとデータの保存**

データを保存および管理するサーバー、データベースソフトウェアなどが含まれます。

**セキュリティシステム**

ファイアウォール、侵入検知システム、暗号化ソリューションなどのセキュリティ関連の機器およびソフトウェアを含む

**ソフトウェアアプリケーション**

ワークプロセスを自動化したり、特定の機能を実行するためのソフトウェアが含まれています

**コンピューティングリソースの補足**

クラウドコンピューティングでは、「コンピューティングリソース」は、物理的なハードウェアリソースではなく仮想化された形式で提供され、ユーザーが必要に応じて動的に調整できます。これは、従来のデータセンター環境とは多少異なる概念です。

**クラウド展開方式の具体例**

*オンプレミス展開 (On-Premises Deployment):*

大学Aは、独自のデータセンターに学校のWebサイトと学士号システムをホストしています。学校は、セキュリティとコンプライアンスの要件を満たすために、重要な学生情報を独自のデータセンターで管理しています。

*クラウドデプロイ (Cloud Deployment):*

会社Bは、オンライン販売のためにWebアプリケーションをクラウドサービスプロバイダのインフラストラクチャにデプロイします。このように、会社Ｂは、トラフィック変動に応じて自動的に拡張され、柔軟に対応することができる。

*ハイブリッド展開 (Hybrid Deployment):*

企業Cは、ビジネスアプリケーションの一部を独自のデータセンターで運営し、機密データを保護するためにオンプレミス環境を使用しています。しかし、企業Cは、顧客管理やコラボレーションツールなどの一部のサービスをクラウドで購読し、コストを削減し、柔軟性を高めます。

MODULE 2: クラウドコンピューティング

**Amazon EC2 インスタンスタイプ**

**汎用インスタンス**

* コンピューティング、メモリ、ネットワーキングリソースをバランスよく提供
* アプリケーションサーバー、ゲームサーバー、エンタープライズアプリケーション用のバックエンドサーバー
* 中小規模データベース
* コンピューティング、メモリ、ネットワーキングに必要なリソースがほぼ同じアプリケーション

**コンピューティング最適化インスタンス**

* パフォーマンスプロセッサを活用するコンピューティング集約型アプリケーションに最適
* 汎用インスタンスとは異なり、高性能Webサーバー、コンピューティング集約型アプリケーションサーバー、およびゲーム専用サーバーに適しています。
* 単一のグループで多数のトランザクションを処理する必要があるバッチワークロードにも使用できます。

**アクセラレートコンピューティングインスタンス**

* ハードウェアアクセラレータ、コプロセスを使用
* 浮動小数点計算、グラフィック処理、データパターンマッチング
* グラフィックアプリケーションゲーム、ストリーミング、アプリケーションストリーミング

**ストレージ最適化インスタンス**

* ローカルストレージの大規模データセットへの順次読み書きアクセスが必要
* 分散ファイルシステム、データウェアハウスアプリケーション、高頻度オンライントランザクション処理（OLTP）システム
* 遅延時間の短いランダムIOPSを提供
* IOPA要件の高いアプリケーションに適しています

**Amazon EC2 料金**

**オンデマンド**

* 不規則な短期ワークロードを持つアプリケーションに最適
* 使用したコンピューティング時間に対してのみ費用を支払う
* アプリケーション開発とテスト
* 予測不能な使用パターンを持つアプリケーションの実行
* 1年以上続くワークロードには不適切

**Amazon EC2 Savings Plans**

* 1年または3年の期間に約束
* オンデマンド料金に比べて最大72％までコストを削減
* 約定を超えた使用量には一般オンデマンド料金がかかります
* AWS Cost Explorerで使用量を管理し、カスタマイズされた推奨を知ることができる

**予約インスタンス**

* **オンデマンドインスタンスを使用する場合に適用される**支払い割引オプション
* 1年～3年約定（標準予約、コンバーチブル予約、定期予約）
* 契約期間が終了してもオンデマンドインスタンスとして引き続き使用可能

**スポットインスタンス**

* 開始および終了時間が自由なワークロード
* ごみ箱を延長できるワークロード
* オンデマンド料金の最大90％までコストを削減できます。

**専用ホスト**

* 専用ホストは、**ユーザー専用の**Amazon EC2インスタンス容量を持つ物理サーバー
* 既存のソケットごと、コアあたり、またはVMごとのソフトウェアライセンスを使用する
* オンデマンド専用ホストと専用ホスト予約を購入可能
* 最高コスト

**Amazon EC2 拡張**

**拡張性**

**Amazon EC2 Auto Scaling**

* タイムリーにタイムアウトするウェブサイト
* 動的調整は需要の変化に対応
* 予測調整は、予測された需要に応じてEC2インスタンスを自動予約
* 柔軟な調整方法を使用: 新しいインスタンスを追加して必要がない場合は終了可能
* 最小 EC2 インスタンス数設定可能
* 最大容量制限可能

**Elastic Load Balancing**

* トラフィックを自動分散するサービス
* 単一接点の役割、着信量に合わせて追加または削除
* 1つのインスタンスだけが過剰労働することはありません。

**メッセージングとキュー**

* モノリシックアプリケーションとマイクロサービス
* DB、サーバー、UI、ビジネスロジックの密結合 – モノリシックアプリケーション
* ある場所で障害が発生すると他の場所でも発生する問題
* マイクロサービスは、アプリケーションコンポーネントが焼結

**Amazon Simple Notification Service (Amazon SNS)**

* 公開/購読サービス
* 購読者にメッセージを投稿します。
* コーヒーショップでレジ係が飲み物を作るバリスタに注文を伝える例
* Webサーバー、Eメールアドレス、AWS Lambda関数オプションなど
* 単一のトピック、複数のトピックからの更新の公開

**Amazon Simple Queue Service（Amazon SQS）**

* メッセージキューサービス
* ソフトウェアコンポーネント間でメッセージを送信、保存、受信
* アプリケーションがメッセージをキューに送信する
* 計算員とバリスタの間のカスタム版の例

**追加のコンピューティングサービス**

**サーバーレスコンピューティング**

1. インスタンス（仮想サーバー）をプロビジョニングします。

2. ユーザーコードをアップロードします。

3. アプリケーションの実行中に引き続きインスタンスを管理します。

**サーバーレス**

* コードはサーバー上で実行されますが、これらのサーバーをプロビジョニングまたは管理する必要はありません
* サーバーを維持するのではなく、新製品や機能を革新に集中
* アプリケーションを自動的に拡張できる柔軟性です。

**AWS Lambda**

* サーバーをプロビジョニングまたは管理することなくコードを実行できるサービス
* 使用したコンピューティング時間のみの支払い
* コード実行中にのみ課金
* 管理不要

**AWS Lambdaの仕組み**

* コード Lamda アップロード
* イベントソース（AWS、モバイルアプリ、HTTPエンドポイント）でトリガーするように設定
* Lambdaはトリガーされた場合にのみコードを実行します。
* 使用したコンピューティング時間の料金のみを支払う

**Amazon Elastic Container Service（Amazon ECS）**

* コンテナ式アプリケーションを実行して拡張可能
* 高性能コンテナ管理システムです。
* Dockerコンテナのサポート

**Docker**

* アプリケーションの構築、テスト、デプロイが可能なソフトウェアプラットフォーム
* Dockerはアプリケーションをコンテナにパッケージ化するために使用され、これは開発環境から本番環境まで一貫した実行環境を提供します。また、Dockerはアプリケーションをより効率的にデプロイおよび拡張し、複数の環境で一貫した開発および運用プロセスを構築するのに役立ちます。
* コンテナベースの仮想化は軽量で高速で効率的であるため、複数のコンテナを単一のホストで実行してリソースを効率的に活用できます。 Dockerは、開発者、システム管理者、DevOpsエンジニアなど、さまざまな役割を持つ人々に人気があり、クラウド環境でアプリケーションをデプロイして管理するために広く使用されています。

**コンテナ**

* コード、依存関係を単一のオブジェクトにパッケージ化する
* セキュリティ、信頼性、確定性重要なプロセス、ワークフローの使用

**Amazon Elastic Kubernetes Service（Amazon EKS）**

* コンテナ型アプリケーションを大規模に展開、管理可能なオープンソースソフトウェア
* AWS Kubernetesの実行に使用するマネージドサービス

**AWS Fargate**

* コンテナ用サーバーレスコンピューティングエンジン
* Amazon ECSとAmazon EKSで動作
* サーバープロビジョニング、管理不要
* 自動的にサーバーインフラストラクチャを管理する
* アプリケーションイノベーション、開発に集中可能

MODULE 2サプリメント

**汎用インスタンス (General Purpose Instances):**

* さまざまなワークロードを処理するように設計されています。
* 一般的なコンピューティング要件

**コンピューティング最適化インスタンス (Compute Optimized Instances):**

* **CPU集約的な作業に特化**
* 高いスループットを提供するためのCPUパフォーマンスの最適化
* 例えば、バッチ処理、メディアエンコーディング、ゲームサーバなどに適している。
* エクセラレイテッド

**コンピューティングインスタンス (Accelerated Computing Instances):**

* 高性能コンピューティングと高速化されたワークロードに使用されます。
* GPUやFPGAなどのアクセラレータを含む並列処理と加速をサポート
* 機械学習、グラフィックレンダリング、科学、エンジニアリングシミュレーションなどに適しています。

**加速コンピューティング**は、特別なハードウェアを介して作業速度を大幅に向上させる方法であり、頻繁に繰り返される作業をまとめる並列処理技術がよく使用されます。通常、シリアルでジョブを実行しているCPUで問題になる可能性のあるトリッキーなジョブをオフロードする方法

**ストレージ最適化インスタンス (Storage Optimized Instances):**

* 大規模なデータセットを処理したり、I/O 集約的なタスクを実行するために特化されています。
* 高速ディスクI / Oと大容量ストレージを提供し、データ処理と分析に最適化されます。
* NoSQLデータベース、データウェアハウス、分析プラットフォームなどに適しています。

**コンピューティング最適化インスタンス/エクセラレートインスタンスの違い**

**1)目的**

コンピューティングの最適化:CPU集約タスクの特化

アクセラレートコンピューティング：加速タスク特化、GPU、FPGAなどのアクセラレータを使用して並列処理に特化加速されたタスクを処理するために特化

**2) ハードウェアリソース**

コンピューティングの最適化:高性能CPUの焦点

エクセラレートコンピューティング：CPUに加えて、GPU、FPGAなどのアクセラレータを含む

**3) 適合作業タイプ**

コンピューティングの最適化：バッチ処理、メディアエンコーディング、ゲームサーバー

エクセラレイテッド：機械学習、グラフィックレンダリング、科学、エンジニアシミュレーション

**加速**

これは、GPU（Graphics Processing Unit）やFPGA（Field Programmable Gate Array）などのアクセラレータを使用して並列化と加速をサポートすることを意味します。

**コプロセッサ(Coprocessor)**

コンピュータシステムで中央処理装置（ＣＰＵ）と連携して特定のタスクを実行する補助処理装置

CPU が扱いにくい複雑なまたは特殊なタスクを処理するために使用されます。これにより、システム全体のパフォーマンスを向上させることができます。

例：グラフィックス処理装置（GPU）

コンピュータグラフィックスタスクに特化しており、画像やビデオ処理、3Dグラフィックレンダリングなどのタスクを実行。

**データウェアハウス(Data Warehouse)**

企業や組織がさまざまなソースから大量のデータを収集、保存、管理、分析するために使用される中央データストア

主に意思決定支援システムにデータを提供するために使用

データウェアハウスは通常、ビジネスインテリジェンス（Business Intelligence）およびデータ分析目的で使用されます。

**OLTP**

オンラインバンキング、ショッピング、注文入力、テキストメッセージの送信など、同時に発生する複数のトランザクションを実行するデータ処理の種類。

基本的な目的は、集計されたデータを分析することです。

レポートを生成し、複雑なデータ分析を実行し、傾向を識別するために使用

**トランザクション**(Transaction以下トランザクション)

* データベースの状態を変更して実行する作業の単位
* データベースの状態を変化させるということ？
* 簡単に言えば、以下のクエリ言語（SQL）を使ってDBにアクセスすること

VM：仮想マシン

**モノリシックアーキテクチャ**

シングルコードベースのアプリケーション

オンプレミス：サーバーのハードウェア、ソフトウェア部分の両方を直接管理

クラウド：しかし、まだサーバーのソフトウェア部分は、ユーザーが直接管理する必要があります

OSをアップデートし、データをバックアップし、セキュリティを気にし、ソフトウェアをインストールする必要があります。

**サーバーレス**

クラウドサービスプロバイダが特定のコードを実行するために必要なコンピュータリソースとストレージのみを動的に割り当て、その部分に対してのみ請求するクラウド実行モデル

1) 開発者がサーバーレスにアップロードした関数は、24時間を通して戻るのではなく休止状態に入る

2）それからユーザー要求が来ている間、サーバーレスは関数を目覚めさせて実行し、要求されたタスクを実行します。

バックエンドをサーバーにアップロードするのではなく、バックエンドを小さな関数ステージに分割してAmazonで直接管理する

**ECS、EKSの違い**

**コンテナオーケストレーションサービス**

Amazon ECS：Amazonが独自に開発したコンテナオーケストレーションサービス。 ECS は AWS の他のサービスと簡単に統合でき、シンプルな設定、スケーリング、管理機能を提供します。

Amazon EKS: Kubernetes に基づく完全管理の Kubernetes サービスです。 Kubernetesは、CNCF（クラウドネイティブコンピューティング財団）によって管理されるオープンソースコンテナオーケストレーションプラットフォームであり、コミュニティで広く使用されています

**ユーザー体験**

Amazon ECS: ECS は Kubernetes より**シンプルで直感的なユーザーインターフェイスを**提供します。 AWSの他のサービスとの統合はうまく機能し、AWS CLIまたはコンソールを介して簡単に管理できます。

Amazon EKS：EKSはKubernetesの標準APIと互換性があるため、 **Kubernetesに精通しているユーザーに精通しています**。 Kubernetes CLI(kubectl)を使用してクラスタを管理し、Kubernetesの豊富な機能を活用

**クラスタ管理**

Amazon ECS：ECSクラスターは、ECSコンテナインスタンス（通常はEC2インスタンス）で実行されます。 ECSはクラスタ管理用のECS Optimized AMIを提供し、ユーザーが管理する必要がある部分を最小限に抑えます。

Amazon EKS: EKS クラスターは、EC2 インスタンスや AWS Fargate などのサービスで実行されます。 EKSはKubernetesコミュニティで開発されたバージョンのKubernetesを実行し、ユーザーはクラスタを管理するために追加の作業が必要です。

**コミュニティとエコシステム**

Amazon ECS：ECSはKubernetesよりもシンプルなモデルを提供しますが、Kubernetesと比較して比較的小さなコミュニティとエコシステム

Amazon EKS：Kubernetesは大きなコミュニティと豊富なエコシステムを持ち、複数の企業や開発者が積極的に貢献します

**コンテナとは？**

コンテナは、あらゆる環境で実行するために必要なすべての要素を含むソフトウェアパッケージ

* アプリケーションコード：コンテナ内で実行されるアプリケーションの実際のコードと実行可能ファイル。
* ランタイム：コンテナ内でアプリケーションを実行するために必要なランタイム環境たとえば、Javaアプリケーションの場合、Java仮想マシン（JVM）がランタイム環境になることがあります。
* ライブラリと依存関係：アプリケーションの実行に必要なすべてのライブラリと依存関係ファイル。これはコンテナイメージに含める必要があります。
* 環境変数と構成：アプリケーションの設定と環境変数の値が含まれます。これにより、アプリケーションの動作を設定できます。
* 実行環境設定：アプリケーションの実行時に必要な環境設定と構成が含まれます。これは、コンテナ内でアプリケーションを実行するときに必要なすべてを定義します。

**コンテナオーケストレーション**

コンテナオーケストレーションは、大規模なコンテナ化されたアプリケーションを効率的に管理および実行するためのツールです。

プロビジョニング、展開、ネットワーキング、拡張、可用性、ライフサイクル管理を自動化

**プロビジョニング**

ユーザーが要求したITリソースを利用可能な状態で準備することを指します。サーバーリソースのプロビジョニング、OSプロビジョニング、ソフトウェアプロビジョニング、ストレージプロビジョニング、アカウントプロビジョニングなどがあります。

**クバネティス (Kubernetes)**

Kubernetisは、コンテナ化されたアプリケーションの自動デプロイ、スケーリングなどを提供する管理システムで、オープンソースベースです。もともとGoogleによって設計され、現在Linux財団によって管理されています

MODULE 3グローバルインフラストラクチャと安定性

**AWSグローバルインフラストラクチャ**

**リージョンの選択**

* データガバナンスおよび法的要件の遵守
* 顧客との近接性
* リージョン内で利用可能なサービス
* 料金

**利用可能領域**

* リージョン内の単一のデータセンターまたはデータセンターのグループは、  
  数十マイル離れています。
* 遅延時間が短いほど近いが、災害防止の可能性を減らすほど離れている

**エッジロケーション**

Amazon CloudFront がコンテンツのコピーをキャッシュするために使用するサイトです。

AWS リソースをプロビジョニングする方法

**AWSマネジメントコンソール**

* Amazon サービスへのアクセスと管理のためのウェブベースのインターフェイス
* 最近使用したサービスにすばやくアクセス
* 名前、キーワード、または略語で他のサービスを検索する
* *モバイルアプリケーション：リソース監視、アラーム表示、支払い情報*
* **AWS コマンドラインインターフェイス (AWS CLI) を**使用可能、コマンドラインから直接複数の AWS サービスを制御可能
* CLI: スクリプトによる作業の自動化
* **SDK** ：APIを介してAWSサービスをより簡単に利用可能

**AWS Elastic Beanstalk**

* 容量調整
* ロードバランシング
* 自動調整
* アプリケーション状態の監視

**WS CloudFormation**

* インフラストラクチャをコードベースで管理およびプロビジョニングするためのAWSのサービス
* JSONまたはYAML形式のテンプレートを作成してAWSリソースを定義してデプロイする
* インフラストラクチャをコードとして定義することで、スケーラビリティ、信頼性、管理の容易さを向上
* 変更の追跡、ロールバック可能
* 繰り返し作業の自動化

MODULE 3サプリメント

**コマンドラインインタフェース**

コマンドラインインターフェイス（英語：コマンドラインインターフェイス、CLI、コマンドラインインターフェイス）またはコマンドラインインターフェイス、コマンドインターフェイスは、仮想端末または端末を介してユーザーとコンピュータがやり取りする方法を意味します。

**AWS Elastic Beanstalk**

開発者がWebアプリケーションとサービスを簡単にデプロイおよび管理できるサービス

Elastic Beanstalkは、開発者がアプリケーションコードをアップロードすると、そのコードを実行するために必要なインフラストラクチャを自動的に設定管理します

**インフラストラクチャをコードとして定義する？**

インフラストラクチャを構成するサーバー、ネットワーク、データベース、ストレージなどをコード形式で定義および管理することを指します。

たとえば、AWS CloudFormationでは、JSONまたはYAML形式のテンプレートを使用してAWSリソースを定義およびデプロイできます。

**SDK** ：ソフトウェア開発キットは、通常、ソフトウェア技術者が使用して特定のソフトウェアパッケージ、ソフトウェアフレームワーク、ハードウェアプラットフォーム、コンピュータシステム、ゲーム機、オペレーティングシステムなどのアプリケーションなどを作成できるようにする開発ツールのセットです。