第1章　AWSサービス全体の概要

1-1. AWS Well-Architectedフレームワーク

AWS Well-Architectedフレームワークの概要

- AWSが最初にクラウドサービスを開始したのは2006年
- Well-Architectedフレームワーク
    - ベストプラクティス集としてまとめたもの

Well-Architectedフレームワークの構成

- 「運用上の優秀性」
- 「セキュリティ」
- 「信頼性」
- 「パフォーマンス効率」
- 「コスト最適化」
- の５つの柱として、各項目のクラウド活用時の設計原則およびAWS上で構築するシステムがベストプラクティスに沿っているかを確認するためのチェックリスト（質問と回答）で構成

Well-Architectedフレームワークの５つの観点の概要

（省略）

５つの観点について簡単な説明

1-2. AWSインフラストラクチャの概要

AWSインフラストラクチャの概要

リージョンとアベイラビリティゾーン

（省略）リージョンとアベイラビリティゾーンの説明

- 複数のAZを利用することや、「AWSマネージド・サービス」と呼ばれるAWSがインフラストラクチャを管理しているサービスを利用することで、パブリッククラウドの恩恵を最大限に享受することができる

参考

- ローカルリージョン
  - Disaster Recoveryを目的としたリージョンがある
  - 大阪ローカルリージョンが該当
  - AWSがへ申請することで利用できる

エッジロケーション

- Amazon CloudFrontのコンテンツ
- Amazon Route53のDNSサービスを提供するためのデータセンタのこと

AWSサービスの範囲

AWSの各サービスは利用できる範囲によって以下に分類できる

グローバルサービス

リージョンに依存しないサービス。

- IAM
- CloudFront
- Route53

リージョンサービス

特定のリージョンのみで利用するサービス

- VPC
- DynamoDB
- Lambda

アベイラビリティゾーンサービス

特定のAZ内で利用するサービス

- EC2
- RDS

参考

- S3はグローバルサービスに分類されるが、データは特定のリージョン二保管される

1-3. アクセス制御サービス

AWSを利用するには、最初に操作するためのユーザを登録する必要がある。

IAMサービスの概要

- Identity and Access Management(IAM)サービスとは、AWSを利用するユーザに対して、AWSへのアクセスを安全に制御するための仕組み

AWSアカウント登録

- AWSアカウントを取得
- メールアドレスでコンソールへログイン
- このユーザをルートユーザという

IAMユーザ及びIAMグループの登録

- ルートユーザではAWSをのすべての操作を行うことができるため、ご操作の危険を伴います
- そこで操作用のユーザ、グループを作成

IAMポリシーによる権限の付与

- IAMユーザ、IAMグループは作成直後権限がなにも付与されていない
- IAMポリシーで適切な権限を付与してAWSの各種サービス、ストレージなどのリソースへのアクセス制御を行うことができる

IAMサービスを通じたAWSの操作方法

- WebブラウザでAWSマネジメントコンソールにログインする
- AWS CLIでWindowやLinuxからコマンド操作する
    - コマンド操作には、操作先のリージョンおよびIAMユーザ毎に作成できる
        - アクセスキー
        - シークレットアクセスキー
        - を事前に作成する必要がある
- AWS SDKでプログラムからAPIを利用する
    - Javaなどの主要なプログラミング言語向けにAWSから提供されているAPI群
    - AWS SDKを利用する場合も、
        - アクセスキーID
        - シークレットアクセスキー
        - による認証情報が必要となります。

試験対策

- IAMユーザ作成後にアクセスキーIDとシークレットアクセスキーを作成することで、AWSを操作するAPIがAWS SDKから使用可能となります

1-4. ネットワークサービス

VPCの構成要素

- サブネット

    - VPC内に構成剝るネットワークセグメントのことで、１つのVPCに対して１つ以上のサブネットで構成されます。
    - パブリックサブネットにするか、プライベートサブネットにするか、どのAZで構成するかなどを検討した上で作成

- インターネットゲートウェイ（IGW:Internet Gateway）

    - VPC内のリソースからインターネットにアクセスするためのリソース
    - インターネットゲートウェイをVPCにアタッチすることで、VPC内のリソースからインターネットにアクセスできる

- ルートテーブル

    - サブネット内のインスタンスに対して静的ルーティングをするもの
    - 設定はサブネット単位で行う
    - パブリックサブネット：デフォルトゲートウェイへのルーティングにインターネットゲートウェイを指定
    - プライベートサブネット：インターネットゲートウェイを指定していない

- 試験対策

    - パブリックサブネットとプライベートサブネットの定義を覚えておく

- NATゲートウェイ

    - NAT機能を有する
    - マネージド・サービス
    - AZ内で冗長化されているためEC2インスタンスを利用する方法より高い可用性が期待できる

- NATインスタンス

    - 実態はEC2インスタンス
    - 「送信元/送信先チェック」と呼ばれる機能を無効化し、自身の通信トラフィックを破棄する設定が必要
    - EC2インスタンスをNATサーバーとして利用するため、単一障害とならない様に冗長化する必要がある

- 試験対策

    - NATインスタンスの特徴を覚えておく

- バーチャルプライベートゲートウェイ（VPゲートウェイ）

    - Direct ConnectやインターネットVPN接続するためのゲートウェイ
    - オンプレミス環境と接続するVPCにアタッチして利用する
        - →コンソール上、仮想プライベートゲートウェイという形で記載されてた

- VPCフローログ

    - VPC内のネットワークインターフェース（Elastic Network Interface:ENI）で通信するトラフィック情報をキャプチャする
    - キャプチャした情報は、Amazon CloudWatch Logs(CloudWatch Logs)へ転送

    - 設定はENI単位、サブネット単位、VPC単位で指定することができる
        - →コンソールで確認したが実態は見えないみたい
        - →VPCの項目にFlow Logsというタブがあってそこから設定するみたい
        - Learn-more

- Elastic IP（EIP）

    - 固定のグローバルIPを提供するサービス
    - 料金について狙われる
        - 起動しているEC2インスタンスにアタッチされている場合には料金が発生しないが、EC2インスタンスにアタッチされていない場合や、アタッチされているEC2インスタンスが停止している場合は時間単位で料金が発生する

- 試験対策

    - EIPはどのような場合に課金されるか覚えておく

- VPCピアリング接続

    - 異なるVPC間をプライベート接続するサービス
    - VPCはAWS上に独立したプライベートネットワーク空間を作成するため、VPC同士は直接通信を行えませんが、VPCピアリング接続することでVPC同士はインターネットを介せず、AWSのプライベートネットワーク内で直接通信することができる
    - 異なるAWSアカウントでも可能
    - ネットワークアドレスが一致、または重複する場合は接続できない！

- VPCエンドポイント

    - 以下の２種類が存在する
    - ネットワークレイヤのゲートウェイ型
        - ルートテーブルに指定されたターゲットを追加することでS3やDynamoDBに接続する際、インターネットを経由せずAWS内のプライベート接続を実現
    - アプリケーションレイヤのインターフェース型
        - 「AWS Private Link」とも呼ばれ、AWSへのAPIコールに対して、インターネットを経由せずにプライベート接続を実現
    - →ゲートウェイは EC2等 => S3,DB に接続する線
    - →インターフェースは AWS内のAPIコール(EC2等=>EC2等)に対して接続する線
    - という理解をした

- セキュリティグループ

    - EC2インスタンス等に適用するファイアウォール機能
    - EC2インスタンスから出る通信：アウトバウンド
    - EC2インスタンスへ通信：インバウンド　の２つを制御する
        - デフォルトの設定値の場合、アウトバウンドはすべて許可。インバウンドはすべて拒否
        - インバウンド、アウトバウンドの別、プロトコル、ポート範囲、アクセス元またはアクセス先IPなどの項目で許可する通信ルールを最大６０まで定義可能
        - EC2インスタンスに複数のセキュリティグループの適用が可能
        - セキュリティグループの設定追加・変更は即座に反映される

        - ステートフルな制御が可能（ルールで許可された通信の戻りの通信も自動的に許可される）
            - →Outbound Rulesはそもそもデフォルトですべて許可されているから、いじらなければこの意識はあまりいらない気がする

- ネットワークACL

    - サブネット単位で設定するファイアウォール機能
    - 異なるサブネット間の通信を制御する際に利用する
        - VPC内に構成したサブネットごとに１つのネットワークACLを設定可能
        - インバウンド、アウトバウンドそれぞれに対して、許可、または拒否を明示した通信制御が可能
        - ステートレス（セキュリティグループとは異なり、インバウンドとアウトバウンドに対する通信制御が必要）

【ネットワークACLとセキュリティグループの主な違い】

項目	**セキュリティグループ**	**ネットワークACL**
適用範囲	インスタンス単位	サブネット単位
デフォルト動作	インバウンド> すべて拒否、アウトバウンド> すべて許可	インバウンド> すべて許可、アウトバウンド> すべて許可
ルールの評価	すべてのルールが適用	ルールの番号順で適用
ステータス	ステートフル	ステートレス

- 試験対策
    - セキュリティグループとネットワークACLの違いは重要！覚えておく

VPC内で利用できるネットワーク・サービス

- Elastic Load Balancing(ELB)

    - EC2や特定のIPアドレスへの通信トラフィックを分散するロード・バランシングサービス
    - 登録されたEC2インスタンスを「バックエンドインスタンス」と呼ぶ
    - 以下の３種類がある
        - Classic Load Balaner(CLB)
            - 標準的なロード・バランシングを提供する
        - Application Load Balancer(ALB)
            - CLBの機能に加え、リクエストのクエリを判断し、予め設定したパスのルーティングに従って振り分ける
            - コンテンツベースで処理を振り分けることができる
        - Network Load Balancer(NLB)
            - 低レイテンシで高いスループットを実現する場合に利用する
            - 送信元のアドレスを保持するため、レスポンスはクライアントへ直接返す

- ～以下、ELBにおける重要な特徴

- 高可用性
    - 複数のAZへ分散することができる
    - 正常なバックエンドインスタンスのみに処理を振り分けることができる
- 自動スケーリング
    - 通信トラフィックの負荷に応じて自動でスケーリングする機能を備えており、ELB自体も冗長性が確保されています
    - ELBにはVPCサブネットのアドレスが自動的に割り当てられます
    - IPアドレスは変動するため、通常、ELBへの接続は「エンドポイント」と呼ばれるELBに割り当てられたDNS名を使用する
- セキュリティ機能
    - SSL復号の機能を備えている
    - ELBがSSL復号を担うことで、バックエンドインスタンスの負荷軽減や証明書の一元管理が可能
    - ELB自体にもセキュリティグループが設定できるので、アクセス制御を行うこともできる
- ヘルスチェックとモニタリング
    - バックエンドのEC2インスタンスが正常に動作しているかヘルスチェックを行う
    - スティッキーセッション
        - １度セッションを確立したバックエンドインスタンスにユーザのリクエストを送信できるようになります
    - Connection Draining
        - バックエンドインスタンスをELBから登録解除する場合、サーバーで何かしらの処理を行っていると、処理が中断してしまうことが懸念される。サーバーの切断を処理完了まで遅延させる機能のことを[Connection Draining]という
    - 通信ログをS3へ保管できる
- 外部ELBと内部ELB
    - 外部ELB（インタネット公開向け）のInternet-facingか、内部ELBとして利用するInternalのいずれかで動作
    - 外部ELBの場合
        - ELB自体がパブリックサブネットに配置されているため、バックエンドインスタンスをパブリックサブネットに配置する必要がなく、プライベートサブネットでセキュリティを維持することができる
    - →ロードバランサーを作成する際に、「スキーム」という項目があり、
        - インターネット向け
        - 内部
        - の２つから選択することができることを確認した
- 参考
    - Internet-facingへの名前解決：パブリックIPが返却される
    - Internalへの名前解決：プライベートIPが返却される
- クロスゾーン負荷分散

    - クロスゾーン負荷分散を利用すると、ELBは複数のAZに登録されたすべての**インスタンスに対してリクエストを均等に分散**
    - 一方、クロスゾーン負荷分散が無効の場合、ELBはAZごとにリクエストを分散するため、AZによってインスタンス数が異なるとすべてのインスタンスにリクエストを均等に分散できない！
    - →理解した！

- Auto Scaling

    - EC2インスタンスを自動でスケーリングする
    - Auto Scalingは予め設定したAmazon Machine Image(AMI)からEC2インスタンスを起動するため、**AMIを最新化しておくことが大事**
    - ユーザデータを利用して、S3やGitからソースやスクリプトを取得することで、EC2インスタンスを最新の状態にすることが可能
    - AutoScalingを実行するには、以下の３つの設定を行う
        - スケーリングプラン
            - いつどこでどのような条件でAuto Scalingを実行するかを定義します。
            - Cloud Watchのメトリクス（CPUの使用率などのリソース状況）に応じて
            - スケジュールを指定して実行
            - 正常なインスタンス数を維持するように実行
            - 手動でスケーリングを実行
        - 起動設定
            - AutoScaling実行時に起動するEC2インスタンス情報を定義
            - 使用するAMI
            - インスタンスタイプ
            - セキュリティグループ
            - キーペア
            - などEC2を構築するために必要なパラメータを設定する
        - AutoScalingグループ
            - EC2インスタンスの管理を行う範囲のことを指し、Auto Scalingで実際に起動するEC2インスタンスの最小数や最大数を定義

- 試験対策

    - Auto ScalingでAMIを最新化することや、ユーザーデータなどを利用する考え方は重要

その他のネットワークサービス

- Route53

    - 可用性の高いDNSを提供するマネージド・サービス
    - ホストゾーン
        - パブリックホストゾーン
        - プライベートホストゾーン
    - ELBやCloudFrontに対してZoneApexレコードをAレコードで指定することができる
    - 通信を制御するルーティングポリシーを使用することもできる
        - レコードを作成するときにRoute53がクエリにどう応答するかを設定できる

- 試験対策

    - Route53には、パブリックホストゾーンとプライベートホストゾーンの２種類の利用方法があることを覚えておく
    - Route53は、ELBなどのエンドポイントをエイリアスとして指定することができる。そのため、エンドポイントはCNAMEレコードではなく、Aレコードとして指定できる
    - Zone Apexレコードを Route53で利用するケースを覚えておく

- このへんよく狙われそうなのでWeb記事参照して理解を深める
    - CNAMEレコードにZone Apexをマッピングできない件について

- AWS Direct Connect(Direct Connect)
    - オンプレミス環境とAWS環境を専用線で接続するサービス
    - インターネットVPN接続もオンプレミス環境とAWS環境を接続できる。品質は低下するが比較的低コストでかつ短期間で導入可能

- 仮想プライベートゲートウェイ（VGW:Virtual Private Gateway）
    - オンプレミス環境とAWSを接続する際に利用するゲートウェイ

- CloudFront
    - エッジロケーションからコンテンツを配信するCDNサービス
    - CloudFront自体には、高可用性、高パフォーマンス、低レイテンシーなネットワークが用意されており、オリジン（コンテンツの提供元）としてELBやEC2、S3を指定することができる。
    - CloudFrontには主に以下の機能がある
        - SSLによる通信の暗号化
            - ユーザが用意したSSL証明書を導入することもできる
            - CloudFrontからバックエンドのAWSサービスまでの通信をSSL暗号化通信にすることも可能
        - 署名付きURL
            - 一定時間アクセスを許可するためのURLを発行し、限定的なユーザに飲み公開する機能
        - カスタムエラーページ
            - オリジンからエラーコードが返却された場合、予め設定したエラーページを表示する
        - 地域制限
            - CloudFrontに接続するユーザの地域情報に基づいて、アクセスを許可/拒否することができる
        - ストリーミング配信
            - Webサービスのコンテンツ配信に加えて映像や音声のストリーミング配信に対応

- CDN Content Delivery Network
    - 世界中に配置されたCDNネットワークから効率的にコンテンツを配信する仕組み。
    - 閲覧者に最も近い拠点から配信するため非常に効率がよい

- 試験対策
    - CloudFrontのオリジン二指定できるサービスを押さえておく。EC2、ELB、S3といったサービスが指定可能だが、オンプレミスのサーバも指定できる

    - 署名付きURLによるアクセス制限の機能は重要
    - CloudFrontは地域制限によるアクセス制御ができることを覚えておく

1-5. コンピューティングサービス

EC2

EC2はAWSが仮想サーバーを提供するサービスで、このサービスを利用して作成した仮想サーバーをEC2インスタンスという。ハイパーバイザ型の仮想化が利用されている。

簡単な操作で作成可能

- 操作手順
    - １．AMIの選択
    - ２．インスタンスタイプの選択
    - ３．インスタンスの詳細設定
    - ４．ストレージの設定
    - ５．タグの追加
        - タグ（AWSリソースにつけるラベル）を追加することで、運用の効率化や課金の振り分けを行うことができる
    - ６．セキュリティグループの設定
    - ７．キーペアの設定
        - 公開鍵認証方式
- Amazon Machine Image(AMI)
    - EC2インスタンスを作成する際に使用する仮想マシンイメージ
    - インスタンス
        - EBS-Backed
        - Instance Store-Backed
- 試験対策
    - EBS-Backedインスタンスはデータを永続化できますが、Instance Store-Backedインスタンスは揮発性のためインスタンス停止時にはデータが削除される
- EC2インスタンスのライフサイクル
    - pending→running
    - running→stopping
    - running→rebooting
    - running→shutting-down
    - shtting-down→terminated
    - terminated
    - rebooting→running
    - stopping→pendding
    - stopping→terminaged
    - stopped→pendding
- ユーザデータとインスタンスメタデータ

- ユーザデータ
    - EC2インスタンスの初回起動時に１回だけスクリプトを実行できる機能
    - 起動時に、S3から最新のコンテンツを反映するスクリプトをユーザデータに記載しておく例
- インスタンスメタデータ
    - EC2インスタンス自身に関するデータ
    - 実行中のインスタンスを設定または管理するために使用されます
    - リンクローカルアドレス「http://169.254.169.254/latest/meta-data/」にブラウザやカールコマンドでアクセスして取得
    - 取得できる情報は以下の通り
        - インスタンスID
        - プライベートIPｖ4アドレス
        - パブリックIPv4アドレス
        - ローカルホスト名
        - 公開鍵
- Lambda
    - 概要
        - サーバーなどのコンピューティングリソースを意識することなく、アプリケーションコードをデプロイしただけで実行することができる
        - 実行する際のリソースと、実行時間に課金される
        - 実行時間に制限があるため、時間を要する処理には不向き
    - ExecutionRole
        - LambdaにアタッチされたIAMロールのこと
        - LambdaはIAMロールの権限に従って各AWSサービスへアクセスするため、IAMロールによるアクセス制御設計が必要となります
    - ロギング
        - すべてのLambda関数の処理結果はCloudWatchにLogsに保存されます。
- API Gateway
    - 概要
        - APIの作成、配布、保守、監視、保護を簡単に行えるサービス
    - EC2インスタンスやLambdaと組み合わせることで、Webアプリケーションのバックエンドやデータ、ビジネスロジックにアクセスすることができる
- その他のコンピューティングサービス
    - Amazon Elastic MapReduce（EMR）
        - 概要
            - 大量のデータを迅速、容易に、かつコスト効果よく処理するためのWebサービス
            - HadoopやSparkなどの分散処理環境を構築、容易に実行可能
    - Amazon Elastic Container Service(ECS)
        - 概要
            - コンテナオーケストレーションサービス
            - Dockerコンテナを簡単に、実行、停止、管理することができる
    - VM Import/Export

        - 概要
            - VM Importはオンプレミス環境に作成した仮想サーバーのマシンイメージをAWSのEC2インスタンスにインポートすることができるEC2サービスの機能
            - VM ExportはVM ImportによってAWS上にインポートされたEC2インスタンスをオンプレミス環境で動作する仮想サーバーのマシンイメージへエクスポートすることができる
            - →この機能で既存のマシンを無駄にすることなく、容易にAWSへ移行することができる！

1-6. ストレージサービス

- Amazon S3
    - 概要
        - 高耐久・大容量のオブジェクトストレージサービス
        - オブジェクトストレージとは、データをオブジェクトとして扱い、IDとメタデータによってオブジェクトを管理する方式
    - 機能
        - 高耐久性
            - デフォルトで同一リージョン内の３箇所のAZへ自動的に複製
            - データの耐久性はナイン１１
            - データストレージとしてだけでなくバックアップストレージとしても使える
        - 大容量ストレージ
            - 利用できる容量に制限がない
            - 利用した分だけの料金が発生するため非常に導入しやすい
            - なお！１ファイルあたりのサイズには制限がある
                - １ファイル５GBまでらしいが、100 MB 以上のオブジェクトの場合は、Multipart Upload 機能を使うことをお考えください、だそうです
                - https://qiita.com/Esfahan/items/126c57d8ff234b50c8c6
        - バージョニング機能
            - オブジェクトの世代管理を提供
            - バージョンIDで管理される
            - デフォルトでは同名のオブジェクトをアップロードすると標準では上書きされる
        - 静的Webサイトホスティング
            - 静的なコンテンツを配信するサイトであればサーバーを構築することなくWebサイトを公開できる
            - **簡易Webサイト以外に、Route53のDNSフェイルオーバー機能と合わせて、WebサイトのSorryページとしても利用される！**
        - 暗号化
            - 以下の３種類の暗号化機能を提供
                - S3デフォルトキーによるAES256暗号化
                - AWS Key Management Service（AWS KMS）で管理されている鍵によるAES-256暗号化
                - ユーザーの任意の鍵によるAES-256暗号化
        - アクセス制御
            - ３つのアクセス制御
                - IAMポリシー

            - バケットポリシー
            - ACLによるアクセス制御
                - →ACLの設定あとで確認したい..
        - ストレージクラス
            - 種類
                - スタンダード
                - 標準低頻度アクセス
                - １ゾーン低頻度アクセス
                - 低冗長化ストレージ
                - Amazon Glacier
        - 柔軟なライフサイクルポリシー
        - 署名付きURL
            - AWS CLIやAWS SDKを利用して、署名付きURLを発行することができる
            - 署名付きURLとは、S3上のデータに対して一定時間だけアクセスを許可するためのURLを発行する機能
        - ログ保管
            - CloudTrailやELBなどAWSサービスのログの標準出力先として利用。
            - 近年ではデータレイクのストレージとして利用することも増えている
            - S3へのアクセスログを取得することも可能
        - マルチパートアップロード
            - 大量料のオブジェクトを複数のパートに分割してアップロードすることで効率よくS3へ転送する仕組み
        - クロスリージョンレプリケーション
            - S3に保存したデータはデフォルトでリージョン内の３箇所のAZへ自動的に保管されるが、クロスリージョンレプリケーションを有効化することで、別のリージョンのS3バケットにオブジェクトを複製する
            - **複製先には別のAWSアカウントを指定可能**

- 試験対策
    - S3の各機能で何ができるのかを押さえておく

- Amazon EBS
    - ※EC2のサービス画面から確認する
    - 概要
        - 永続可能なブロックストレージサービス
        - EC2インスタンスにアタッチすることで利用できる
    - 機能
        - 複数のストレージタイプ
            - 汎用SSD
            - プロビジョンドIOPS SSD
            - スループット最適化HDD
            - ColdHDD
            - マグネティック
        - 高可用性
            - AZ内で自動的に複製される仕組みを備えているため、単一のディスク障害を回避することができる

        - スナップショットによるバックアップ
            - スナップショットを用いてバックアップを取得することができる
            - スナップショットは自動的にS3に保管される耐久性を備えている
            - EBSスナップショットは取得を実行した時点のものが保管されるため、完了を待つ必要はない
            - また、スナップショットは初回はフルで行うが、以後は差分で取られる

- 試験対策

    - 各ストレージタイプの違いやスナップショットによるバックアップの仕組みなど、EBSの特徴を覚えておく

- その他のストレージサービス

    - インスタンスストア（エフェメラルディスク）
        - 特定のEC2インスタンスで利用できる無料のストレージサービス
        - 揮発性なので、インスタンス停止時にデータが消失
        - 重要なデータの保管方法は別途検討する必要
            - →AWSマーケットでAMIを見てみたが、インスタンスストアのものがなかったような...どこにあるかあとでみとく
    - Amazon Elastic File System
        - スケーラブルな共有ストレージサービス
        - 複数のEC2インスタンスから共有ファイルストレージとして利用することができる
        - ストレージ容量やパフォーマンスを自動的にスケーリングする機能も備えている
    - AWS Storage Gateway(SGW)
        - 概要
            - S3へNFS、SMB、、iSCSIといった標準プロトコルでアクセスできるサービス
            - EC2インスタンスだけでなくオンプレミスサーバーにS3をマウントして利用することも可能
                - →このサービス初めてしった！
        - キャッシュ型ボリュームゲートウェイ
            - データはS3へ保管するが、アクセス頻度の高いデータはローカルにキャッシュすることで高速にアクセスできる
            - インターフェースにはiSCSIを使用
        - 保管型ボリュームゲートウェイ
            - データをスナップショットとしてS3へ格納
            - インターフェースにはiSCSI
        - テープゲートウェイ
            - 物理テープ装置の代替としてデータをS3(Glacier)に格納
            - インターフェースにはiSCSI
        - ファイルゲートウェイ
            - データをS3に直接オブジェクトとして格納
            - インターフェースにはNFS
        - →SMB（Server Message Block）はここを参照
          http://www.atmarkit.co.jp/icd/root/27/77053027.html

- 試験対策

    - インスタンスストアはエフェメラルディスクとも呼ばれます
    - どちらの名称も覚えておく

- AWS Import/Export

    - ユーザから送付された物理ディスクのデータをAWSがAWS内のストレージへ転送したり、ユーザーの物理ディスクに書き出したりするサービス
    - ネットワーク転送では非常に時間がかかる大容量のデータ移行に非常に有効
    - しかし、移行のために高価なディスクを購入する必要があったり、送付時のディスクの耐久性やセキュリティに懸念事項があることから、次に説明する「SnowBall」という後継サービスが登場！

- AWS Import/Export Snowball

    - ペタバイト規模の大容量データをAWS内のストレージへ転送するサービス
    - Snowballで使用される筐体は、AWSが用意した物理耐久性が高いもので、AWSマネジメントコンソールからジョブを作成するだけで、登録した住所へ自動的に発送される
    - Snowball内のデータは自動的に暗号化されるなどセキュリティ面でも考慮されている
        - →コンソールで「AWS Snowball」で出てくる！
        - > AWS Snowball は、AWS 所有のアプライアンスを使用して、インターネットの速度よりも速くデータを転送したり、AWS クラウドの性能をローカルで利用したりできるサービスです。

- 試験対策

    - AWS Import/ExportとVM Import/Exportは別のサービス
    - 混同しないように注意
        - ※VM Import/Exportとは
            - オンプレミスで作成したサーバイメージをAWS環境におくるもの。（前述！）

1-7. データベースサービス

AWSで提供されているデータベースサービスの概要を説明

- AWSのデータベースの特徴

    - AWSの特徴にマネージドサービスがある
    - AWSで提供されているデータベースサービスは**すべてマネージド型のデータベース**
    - マネージド型の理由
        - データを格納するために使用するものなので、テーブルやカラムの設計にできるだけ時間を費やしたいものです
        - 実際は、ハードウェアの調達やINSTALL、パッチ適用、バックアップ・リストア、DRなど、、
        - いろいろやることがあるので、マネージド型サービスとしてAWSが管理を担っている

- RDS

    - 概要
        - マネージド型のリレーショナルデータベースサービス
    - Amazon Aurora（Aurora）

        - Amazonが開発したリレーショナルデータベースサービス
        - MySQLやPostogreSQLと互換性があるため、これらのデータベースを使用したシステムはスムーズに移行できる
        - 性能は、両者よりAuroraのほうが優れている（両者よりスループットが高い）
        - データは３箇所のAZに保管され、２箇所のディスクに書き込まれ、合計で６箇所に保存される

- 試験対策

    - Amazon Auroraの特徴を覚えておく

- RDSの主な特徴

    - バッチ適用の管理負荷軽減
        - メンテナンスウィンドウで指定するだけでAWSが自動でパッチ適用
        - 通常は数ヶ月に１回のペースでパッチがリリースされるため、パッチリリース後、最初にメンテナンスウィンドウで設定した曜日や時間帯になった時点でパッチが自動適用される
    - バックアップと復元
        - 標準設定の場合、１日１回自動バックアップが実施され、データベースとトランザクションログが取得される
        - データベースのバックアップには、ストレージボリュームのスナップショットを作成し、データベースインスタンス全体を取得
        - 保存期間を設定することが可能で、最大３５日分を保存することができる
        - 自動ではなく、手動で任意の時間にバックアップを取得することもできる
            - →スナップショット取得中は短時間のI/O切断が起こる
            - マルチAZ配置だと大丈夫とのこと!https://docs.aws.amazon.com/ja_jp/AmazonRDS/latest/UserGuide/USER_CreateSnapshot.html

            - > Single-AZ DB インスタンスでこの DB スナップショットを作成すると、I/O が短時間中断します。この時間は、DB インスタンスのサイズやクラスによって異なり、数秒から数分になります。マルチ AZ DB インスタンスは、スタンバイ側でバックアップが作成されるため、この I/O 中断の影響を受けません。
        - 復元する際は、バックアップからデータベースインスタンスを作成して起動
        - トランザクションログも取得しているので、バックアップ取得時点に近いデータに復元することができる
            - →これを**ポイントインタイムリカバリ**といいます
        - RDSはデータベースのトランザクションログを５分毎にS3二保存している
        - したがって最新のデータを復元すると過去５分以内の状態にデータベースインスタンスを戻すことができる
        - 参考）RDSの復元方法
            - 通常の復元
            - ポイントインタイムリカバリ
            - →S3にスナップショットもトランザクションログも保存しているのでこれを使用して復元する

        - データベースを復元するとき、新しくデータベースインスタンスを作成・起動することになるため、アプリケーション側は新しいエンドポイントに接続する様に修正する必要がある
            - →これについて調査！
            - http://mzgkworks.com/post/aws-rds-restore-from-snapshot/ここに対策が記載されていた。
            - 復元したインスタンスのDBインスタンス識別子を変更するみたい
            - RDSを削除すると過去に撮ったスナップショットも削除されるっぽい
        - バックアップのスナップショットはリージョン間でコピーすることができるためDR対策にも利用できる
    - マルチAZ
        - RDSを複数のAZに構築することを指す
        - Aurora以外のデータベースはマルチAZインスタンスを作成すると、RDSはマスターのデータベースインスタンスを自動的に作成、それと同時に異なるAZにあるスレーブインスタンスにもデータを複製
    - リードレプリカ
        - 読み取り専用としてデータベースを構築したもの
        - 読み取り頻度が高い場合、スループットを向上させることができる
        - →スレーブとは別に、リードレプリカが存在するみたい！
    - SSL接続
        - SSLを使用してRDSのデータベースインスタンスに接続することができる
        - MySQLでSSL接続する例）
            - mysqlコマンドの実行時に、`--ssl-ca [公開鍵のフルパス]`オプションをつけて公開鍵を指定します

- 試験対策
    - RDSではSSL接続を利用できます
    - 接続に使用するコマンドのオプションを覚えておく！
        - →コマンドのオプションレベルで出題されるの...???

- DynamoDB
    - 概要
        - マネージド型のNoSQLサービス
        - NoSQLは非RDBMSの総称
        - 構造化データ以外の一部のデータを扱うことができる
    - キーバリュー型データモデル
        - NoSQLにはいくつかのデータモデルがあり、代表的なものにキーバリュー型があります
        - DynamoDBはキーバリュー型のデータベース
        - Jsonのような半構造データもバリューとして格納できるｚ
    - マルチAZ
        - DynamoDBのデータは自動的に３箇所のAZに保存される
    - 結果整合性モデル
        - 書き込んだデータは時間が経てば正しく反映されるがデータ読み取りのタイミングによっては書き込んだデータが反映されていない状態になる、こと
        - データを３箇所のAZに分散・保存されるが書き込み完了には時間差がある
        - まだ最新が反映されていないAZからデータを取得してしまう可能性がある

        - この様に「正しいデータが取得できない可能性があるので機能要件上利用することが難しい」という場合の対応策として「Consistent Read（読み取り一貫性）」というオプションが提供されています。
        - このオプションで、書き込みが反映された最新のデータを読み取ることができる！

- 試験対策
    - DynamoDBのデータの保持方法と、結果整合性の仕組みを理解しておく
- 参考
    - DynamoDBの３つのAZにデータを書き込む場合、２つのAZに正常に書き込みが完了した時点でDynamoDBに対してデータの書き込みが正常に完了したと判断して、成功の旨の応答を戻します。この２つのAZへの書き込みが完了し、残り１つのAZはデータ書き込み中の状態でまだ書き込みが完了していないAZで次のデータの読み取り処理が行われると、最新のデータが取得できない状態が発生する
- Redshift
    - 概要
        - ペタバイトクラスのデータも扱うことができるマネージド型のデータウェアハウスサービスでBIツールなどを利用した大量のデータの集計・分析に向いている
        - データウェアハウスとは：企業活動の中などで蓄積される膨大なデータに対して、ある特定の内容を分析するために利用するシステム、もしくはそのために利用する大量のデータを指す
        - Redshiftはノードと呼ばれるコンピューティングリソースの集まりで構成
        - リーダーノードが処理を受け、コンピュータノード二処理を依頼する。
        - ノードの集まりのことをクラスターという
    - スナップショット
        - スナップショットを利用することでバックアップを取得
        - スナップショットはクラスターのポイントインタイムバックアップ
        - 取得方法は自動と手動がある
        - 自動スナップショットの設定を有効にしている場合、８時間毎、もしくはノードあたり５GBのデータの更新があったときに自動取得
        - 自動スナップショットは設定した保存期間がすぎるとRedshiftによって自動的に削除されるため、永続的にバックアップを保存したい場合は手動で取得すること！
        - スナップショットの格納先は、標準ではRedshiftのクラスターが配置されているリージョンになるが、別のリージョンにコピーも可能（**クロスリージョンスナップショット**という）
- 試験対策
    - Redshiftのバックアップの特徴を覚えておく
- 参考
    - Redshiftの使用はPostgreSQL8.0.2に準拠している
    - PostgreSQLを操作するためのpsqlコマンドを利用してRedshiftを操作することができ、一般的なRDBの操作経験者には扱いやすいが、大きな違いもある

    - https://docs.aws.amazon.com/ja_jp/redshift/latest/dg/c_redshift-and-postgres-sql.html

- ElastiCache
    - Amazon ElastiCache(ElastiCache)はマネージド型のインメモリデータベース
    - メモリ上で処理を実行するため、高スループットかつ低レイテンシな処理を実現
    - キャッシュを使用しないアプリケーションは以下のような構成＝構成例の説明省略＝
    - EC2インスタンスとRDSの間にElastiCacheが介在するようなアーキテクチャ例

1-8. データ通知・連携処理サービス

データ通知・連携処理サービスの概要

- 通知処理
    - Amazon Simple Email Service(SES)
        - AWSで提供されるEメールサービス
        - WebアプリケーションからSESで提供されるAPIを実行することで、メールを利用して簡単に通知を行うことができる
    - Amazon Simple Notification Service（SNS）
        - AWSで提供されるメッセージ通知サービス
        - 任意のメッセージなどをHTTPSなどの様々なプロトコルでアプリケーションから簡単に送信することができる
        - また、CloudWatchやSQS、LambdaをはじめとしたAWSの様々なサービスへの通知・連携も可能
            - トピック
                - メッセージを送信し、通知を受信するためのアクセスポイントのこと
                - トピックを作成し、購読者がこのトピックを行動（サブスクライブ）することで、通知の送受信が可能となる
            - 購読者（サブスクライバ）
                - 対象となるトピックから発信されるメッセージの購読者（サブスクライバ）を設定します
                - 以下のようなプロトコルを選択できる
                    - HTTP/HTTPS
                    - Email
                    - SQS
                    - Lambda
                - メッセージの配信（パブリッシュ）
                    - 作成されたトピックに対して、アプリケーションなどからメッセージを送信します
                    - 送信されたメッセージは、SNSからサブスクライバへ配信（パブリッシュ）されます
    - 参考
        - SNSのようにメッセージを配信（パブリッシュ）・購読（サブスクライブ）する形式の通知の仕組みは「Pub/Sub」と呼ばれている

- メッセージキューイング処理

    - Amazon Simple Queue Service（SQS）というメッセージキューイングマネージドサービスを利用
    - メッセージをキューイングすることでバッチのような非同期かつ並列な分散処理が可能になる
    - 処理負荷の大きいビデオエンコーディングを複数のEC2に振り分けて処理する

- パイプライン処理

    - Amazon Data Pipeline（Data Pipeline）サービスでデータのETLなどの順次処理を行うことができる
    - 特徴
        - データのETL処理をスケジュール機能で自動化
        - データの順次処理をワークフロー形式で定義
        - 処理の成功・失敗などのイベントの通知が可能
        - オンプレミス環境との連携が可能
    - ユースケース
        - １日１回定期的にDynamoDBなどのデータベースからデータを取り出し、S3にバックアップしたあとで、取り出し元のテーブルをDynamoDBから削除
        - 企業の複数拠点にあるWebサーバーからアクセスログをS3に定期的に保存・加工し、S3からRedshiftへデータをロード
    - 試験対策
        - Data Pipelineを利用する主なユースケースを押さえておく

- ストリーミング処理

    - リアルタイムに流れてくるデータの処理（ストリーミング処理）には、AWSではAmazon Kinesis(Kinesis)を利用します。
    - IoTなどのデバイスからデータをリアルタイムに受信して分析する場合に利用される
    - 以下の３つのサービスから構成される
        - Kinesis Data Streams
        - Kinesis Data Firehose
        - Kinesis Data Analytics

1-9. 構成管理サービス

構成管理

- プロビジョニング

    - provision(供給)という単語から派生した用語
    - プロビジョニングとは
        - システムの利用の需要を予測し、必要に応じて設備やサービスなどのリソースを供給できるように備えること

- デプロイ

    - 実行ファイルやアセットを配布すること

CloudFormation

- this is..

    - AWS内のすべてのインフラストラクチャリソースを自動でプロビジョニングできるサービス

- Cloud Formationで重要な用語

    - テンプレート
        - CloudFormationの設定ファイルを指す
        - このファイルにプロビジョニングしたいAWSリソースの集合を指す
    - スタック
        - テンプレートを利用してCloudFormationによってプロビジョニングされるAWSリソースの集合を指す
            - →つまり、テンプレートに基づいて
            - プロビジョニングして
            - スタックを作成するのが
            - CloudFormation!!

Elastic Beanstalk

- this is..

    - Webアプリケーションやサービスをサーバーにデプロイでき、環境の管理も行うことができるサービス
    - アプリケーションを実行しているインフラストラクチャを深く理解していなくてもアプリケーションのデプロイとその管理を行うことができる

- 対応しているアプリケーション

    - Java
    - .NET
    - PHP
    - Nodejs
    - Python
    - Ruby
    - Go
    - ※Dockerにも対応している

- 対応しているサーバー

    - Apache HTTP Server
    - nginx
    - Passenger
    - Microsoft Internet Information Server(IIS)

- 試験対策

    - Elastic Beanstalkが対応しているアプリケーションとサーバーについて覚えておく

その他のサービス

- AWS OpsWorks(OpsWorks)

    - this is..

        - 構築手順通りにサーバー構築手順を自動化する構成管理サービス
        - Elastic Beanstalk同様、デプロイ、プロビジョニング、リリース後の監視の機能がある
        - ChefやPuppetといったサーバー構成を自動化するツールを利用できる

- AWS CodeDeploy(CodeDeploy)
    - this is..
        - アプリケーションのデプロイを自動化するサービス
- 参考
    - AWSには、コード作成、管理用としてAWS CodeCommit、ビルドテスト用としてAWS CodeBuildというCodeシリーズサービスがある

1-10. 運用管理サービス

CloudWatchの概要

- this is..
    - AWS上で可動する様々なシステムやAWSのリソース情報を収集、監視、可視化するサービス
- EC2のメトリクス
    - CloudWatchで取得・監視する項目のことを「メトリクス」と呼ぶ
        - 標準メトリクス
        - カスタムメトリクス
        - ２種類が存在する
- 標準メトリクス
    - CPUUtilization
        - CPU使用率
    - DiskReadOps
        - インスタンスストアボリュームからの読み取り回数
    - DiskReadBytes
        - インスタンスストアボリュームから読み取られたバイト数
    - NetworkIn
        - すべてのネットワークインターフェースから受信されたバイト数
- カスタムメトリクス
    - メモリ使用率やディスク使用率などは、EC2インスタンス側で監視スクリプトを作成、設定し、CloudWatchに送ることができる
- 試験対策
    - EC2のメモリ使用率やディスク使用率を監視するには、カスタムメトリクスの設定が必要
- 参考

    - CloudWatchエージェントをLinuxやWindowsにINSTALLすることで、エージェントからカスタムメトリクスを設定、送信することもできる！

- CloudWatchの監視間隔と保存期間
    - CloudWatchでEC2を関しする場合、基本モニタリングと詳細モニタリングの２つのプランが用意されている
        - 基本モニタリング
        - 詳細モニタリング

- 試験対策
    - 基本モニタリングと詳細モニタリングでは、収集したデータの監視間隔が異なります

- 参考
    - 監視間隔はAWSサービスによって異なる
    - 例えばELBやEBS、RDSは１分間隔のデータが無料で閲覧できる

- 個人的な確認事項
    - 詳細モニタリングの有効化は各サービス（EC2）側で行う！
    - EC2ならインスタンスのモニタリングタブで確認できた

- CloudWatchアラームによるアラームとアクション設定
    - CloudWatchの状態
        - OK
            - 定義された監視閾値を下回っている状態
        - ALARM
            - 定義された監視閾値を上回っている状態
        - INSUFFICIENT-DATA
            - Cloudwatchに送信されるデータが不足しているため、正常か異常か判定できない状態

- 個人的な確認事項
    - EBS-backedなEC2インスタンスは終了操作を実行してterminatedになってからも一定時間残る

- CloudWatch Eventsによるイベント駆動型監視とアクション設定
    - Cloudwatchはメトリクスが対象だが、
    - CloudWatch Eventsは状態の変化を対象にアクションを実行する機能
        - 例えば
        - EC2インスタンスの状態変化
        - スケジュール
        - Auto Scaling　成功、失敗などの状態監視
    - 監視後のアクションは
        - Lambda関数
        - SNSトピック

        - Kinesisストリーム
        - SQSキュー
    - 活用例
        - スポットインスタンスが強制停止する状態変換を監視し、検知したらLambdaを実行して新たなスポットインスタンスを起動するなどの自動化が可能！
        - →なるほど腑に落ちた

- 試験対策

    - CloudWatchアラームとCloudWatch Eventsの特徴の違いを押さえておく！
        - →押さえれた！

その他の運用管理サービス

- CloudWatch Logs

    - CloudTrailやVPCフローログなど様々なAWSのログを統合的に収集するサービス

- CloudTrail

    - AWS操作をログとして記録するサービス

- VPCフローログ

    - VPC内のネットワークインターフェース間で行き来する通信の内容をキャプチャする機能
    - 意図しない通信が行われていないかどうかの監視・監査に利用

- AWS Config

    - AWSのサービスで管理されているリソースの構成変更を追跡するサービス

- AWS Systems Manager

- AWS Trasted Advisor

    - ベストプラクティス二基づいて、コスト最適化、セキュリティ、耐障害性、パフォーマンス、サービスの制限の５項目でユーザーのAWSの利用状況をチェックし、改善すべき事項を推奨するサービス

- 試験対策

    - 運用管理に関連するサービス群の役割の違いを覚えておく

1-11. 演習問題

- グローバルサービスに該当するものは次のうちどれか
    - S3
    - Route53
- リージョンサービスに該当するものは次のうちどれか
    - VPC
- IAMユーザを作成した直後に設定されているアクセス権限に関する説明で正しいもの
    - 権限は付与されていない

- AWS CLIやAWS SDKを利用するために必要な認証情報は次のうちどれか
    - IAMユーザーのアクセスキーIDとシックレットアクセスキー
- パブリックサブネットとプライベートサブネットの違いで正しいもの
    - ルートテーブルにインターネットゲートウェイのルーティングがあるかないか
- ロードバランシングサービスを提供するサービスは？
    - ELB
- Elastic IPに対して課金が発生しないのは次の打ちどれ
    - Elastic IPをアタッチしているEC2インスタンスが起動している場合
- セキュリティグループの説明で正しいもの
    - 何もルールを設定しないと通信が許可されない
    - ステータスを保持できる
    - 設定を変更したあと、**すぐにルールが適用される**
        - →ここ勉強になった
- AWSとオンプレミス環境を安定した閉域網で接続したいときに利用するサービス
    - AWS Direct Connect
- ハイパーバイザー型仮想化を利用したコンピューティングサービス
    - Amazon Elastic Compute Cloud（EC2）
- コンピューティングリソースを管理することなく、PythonやNodejsなどのアプリケーションコードをデプロイするだけ実行することができるサービス
    - AWS Lambda
- API作成から運用まで容易に行えるマネージドサービスは次の打ちどれ
    - API Gateway
- S3の特徴として正しくないもの
    - データは自動的に複数のリージョンへコピーされる
        - →自動的にコピーされるのは**複数のAZ**！
- NoSQLに属するマネージドデータベースサービスは次の打ちどれ
    - DynamoDB
- RDSの特徴として正しくないもの
    - データキャッシュが可能
        - →これはElasticache
- Lambdaにメッセージを通知・連携したいときに利用するサービスとして適切なもの
    - SNS
- 処理の重いビデオエンコーディングをバッチ処理として複数のEC2インスタンスに振り分けたいと考えています。適切なAWSサービスはどれ
    - AWS SQS
- 定期的にDyanamoDBのテーブルからデータを取り出しS3へ保存・加工したあとでDyanamoDB上のテーブルを削除する一連の動作の自動化に適切なサービス
    - Amazon Data Pipeline
- 複数のデバイスから送信されるデータをリアルタイムでストリーミング処理したいと考えています、適切なサービスはどれ
    - Amazon kinesis
- 「テンプレート」と呼ばれる定義ファイルに従って、AWSリソースをプロビジョニングするサービスは
    - AWS CloudFormation
- CloudFormationでプロビジョニングされるAWSリソースの集合をなんという
    - スタック

- Chefを用いてサーバー構築を自動化できるサービスは
    - AWS OpsWork
- EC2インスタンスに対してCloudWatchのカスタムメトリクスを使用する必要がある監視項目
    - メモリ使用率
- VPC内のネットワークインターフェース間の通信トラフィックを監視するサービスとして適切なもの
    - VPCフローログ

第2章　AWSにおける高可用アーキテクチャ

2-1. 高可用性の定義

一般的な可用性の定義

- 可用性とは、システムが正常に継続して動作し続ける能力
- 参考
    - AWSサービスの中にはSLAで稼働率が公表されているものがあるが、あくまでも努力目標値

AWSにおける可用性向上策

- システム障害の原因の多くはハードウェアによるもの
- AWSでは、Design For Failureという考え方がある
    - 障害を回避するのではなく障害が発生してもシステムが継続できる様に設計する
- リソースを冗長化する
    - オンプレミスと同様に単一障害点（SPOF）をなくすことが重要
- 地理的に離れた場所で冗長化する
- システムを疎結合で構成する
    - 可用性を高めるうえで、システムやコンポーネントを独立させる（疎結合にする）考え方は非常に重要
    - SQSとELBを用いた疎結合の構成例
        - SQSとELBはAWS内部で冗長されるらしい！ので単一障害点にはならない

2-2. ネットワークにおける高可用性の実現

ネットワークサービス

- Amazon Virtual Private Cloud(VPC)
    - AWS上に独立したプライベートネットワーク空間を作成できるサービス
    - アカウント作成時にデフォルトで各リージョンに１つ作成されている！ことを確認した！
- AWS Direct Connect(Direct Connect)

    - 「接続ポイント」と呼ばれるロケーションを経由してオンプレミス環境とAWSをの間を専用線で接続することができる
    - オンプレミス環境と接続ポイント間の可用性はユーザーが考慮する必要がある。
    - この回線の可用性は通信キャリアの提供プランによって異なるため、Direct Connectを契約する前に確認すること！

- Amazo Route53 (Route53)

    - ドメインネームシステムのマネージドサービス
    - DNSフェイルオーバー機能を備えたフェイルオーバールーティングを活用することで、障害が発生した場合のDNSの切り替えをすぐに行える

- route53で指定できるトラフィックポリシー

ルーティングポリシー	**説明**
レイテンシーベースルーティング	最もレイテンシーが低いリソースへルーティング
荷重ルーティング	複数のリソースに対して加重度を設定し、指定した比率に応じて処理を分散
位置情報ルーティング	接続クライアントの位置情報に基づいて、地理的に近い場所へ
フェイルオーバールーティング	ヘルスチェック結果に基づいてルーティング
シンプルルーティング	設定されたレコードの情報に従ってルーティングする

- 試験対策

    - 各ルーティングポリシーで何が実現できるかを覚えておく
    - 例えば、位置情報ルーティングでは、リージョンにかかわらず、あくまで地理的に近い場所へ名前解決を行う

- Route53で使用できるレコードタイプの一覧を紹介（割愛）

- 試験対策

    - Route53のレコードタイプの中で、ドメイン名からIPアドレスを参照するAレコードとドメイン名から別のドメイン名を参照するCNAMEレコードは特に重要

高可用ネットワークの構築

前述したサービスを利用して高可用ネットワークを構築するにはどうしたらいいか、という話題へ

- オンプレミス環境：AWS間の接続

    - Direct Connect
        - 安定した高速通信環境が必要な場合に利用する
        - コストがかかる
        - 開通まで時間がかかる、すぐに利用したい場合には向かない
    - インターネットVPN

        - 各メーカーのルータに最適化された設定ファイルをAWSマネジメントコンソールからダウンロードし、その設定ファイルを元にオンプレミスのルータの設定を行うだけで利用可能

- Direct Connect冗長化パターン
    - DirectConnectを２回線用意して冗長化することでネットワークの高可用性を実現
- 参考
    - Direct Connectには占有型と共有型の２つの接続タイプがある
        - 占有型：契約した物理接続に対してユーザー側で自由に論理接続を作成できる
        - 共有型：物理接続をキャリアが保有しているため、論理接続単位で料金が発生する
- Direct Connect とインターネットVPCの併用パターン
    - 障害発生時にVPCに切り替えるパターン
    - VPCにフェイルオーバーした場合、通信品質や帯域が異なるのでパフォーマンスに影響が出る可能性あり
- 試験対策
    - Direct ConnectとインターネットVPNの特性やユースケースを覚えておく
- VPC内通信
    - 万が一の有事に備えた設定を！
    - NATゲートウェイ、インターネットゲートウェイはデフォルトで冗長化されている
- VPCピアリング
    - 異なるAWSアカウントのVPCや、同一AWSアカウントの別リージョンのVPCを指定できたりする

2-3. コンピューティングにおける高可用性の実現

コンピューティングサービス

- Amazon EC2
    - オンプレミス同様にサーバーにクラスタリングソフトウェアを導入して高可用性を実現することもできるが、AWSではELBやAutoScaling、Route53などのマネージドサービスと組み合わせるのが一般的
- Auto Scaling
    - スケールアウト
        - スケーリングプランの設定値をトリガーに、Auto Scalingグループの設定値に従ってEC2インスタンス数を増減
    - スケールイン
        - デフォルトで以下の順で実行
            - １．EC2インスタンス数が最も多いAZ内からランダムにAZを選択
            - ２．AZのインスタンス数が同じ場合は、最も古い起動設定を使用したEC2インスタンスがあるAZを選択

        - ３．最も古い起動設定を使用したEC2インスタンスが複数ある場合は、次に課金が発生するまでの時間が最も短いEC2インスタンスを選択
        - ４．３．が複数ある場合は、この中からランダムでEC2インスタンスを選択

ただ単にAutoScalingを設定しただけでは意図したとおりに動作しないこともあるので、クールダウンやライフサイクルフックといったオプションを利用する

- クールダウン
    - AutoScalingが連続で実行されてしまわない様に、待ち時間を設定する機能
- ライフサイクルフック
    - AutoScalingによるEC2インスタンスの起動または終了を一時的に待機させ、指定したアクションを実行する
    - 例えば、
        - 終了時にログを外部へ出力する場合や、起動時に特定のデータをロードする場合などに利用します。
- CloudFront
    - エッジロケーションからコンテンツを配信するCDNサービス
    - CloudFront自体の可用性を考慮した設計を行う必要はない
        - 高可用性
        - 高パフォーマンス
        - 低レイテンシー　を備えているので！
    - 世界中からアクセスされる大規模なWebサーバーや、予期できないアクセス集中が発生しうるサービスを提供する際に利用されます
- AWS Lambda
    - 可用性やスケールはすべてLambdaで管理されている
    - 実行した分だけ料金が発生する
    - 処理時間に限りがあるため、時間を要するコードは実行できない
    - Lambdaの制限について
    - https://docs.aws.amazon.com/ja_jp/lambda/latest/dg/limits.html

VPCリソースにおける高可用性コンピューティング

- ELB
    - ロードバランサー
    - EC2の高可用性に利用される
    - ELB自体は自動的にスケールされるが、IPアドレスは不定期に変わるので、ELBへ接続するにはエンドポイントを指定する
    - ELBはVPC内のリソースなので、**複数のAZをまたぐことは可能ですが、リージョンをまたぐことはできない**
    - 複数のAZへEC2インスタンスを配置することが大切（マルチAZ）
- 一般的なWebアプリケーションにおける構成例

    - ELB+EC2+AutoScalingGroup

- 一般的なWebアプリケーションにおける障害発生時の挙動

    - ヘルスチェックの説明をしているだけ
    - ダウンしたEC2にはアクセスしないよう制御

- 参考

    - ELB+AutoScalingで自動的にスケールすることは可能だが、短時間でアクセスが急増するとスケールアウトが間に合わず、システムが過負荷な状況になる可能性がある
    - CloudFrontを活用してキャッシュから応答することで予期できないアクセス集中に対応できるようなアーキテクチャも重要

- 参考

    - EC2インスタンスがダウンしたからと言って、セッションを失うような作りにしては行けない。ステートレスなアプリケーションを推奨！

- データベースを持つWebアプリケーションにおける構成例

    - マスタースレーブ構成の説明

- 参考

    - EBSは複数のEC2インスタンスからアクセスする共有ディスクとして使用できない！

- RDSインスタンス障害発生時の挙動

    - シングルAZのRDSに障害が発生した場合
        - 障害発生時、自動的に再起動
        - RDS内のデータに付いては、ユーザー側で復元する必要があるため、ポイントインタイムリカバリ機能などを使う
    - マルチAZのRDSに障害が発生した場合
        - 以下の順ですべて自動！
        - １．障害発生
        - ２．スレーブをマスタに昇格
        - ３．マスターが利用していたDNSレコードをスレーブのDNSレコードへ切り替える
        - ４．データは同期されているので意識しなくてよい
        - ５．フェイルオーバー中はわずかにダウンタイムがある

- 試験対策

    - マルチAZのRDSでフェイルオーバーが発生したときの挙動は試験にでる！！

- データベース内のレコードが破損した場合

    - RDSの自動再起動やフェイルオーバーでは復旧できないため、バックアップからリストアする必要がある
    - リストア方法例
        - １．障害が発生したRDSエンドポイント名を控え、別のエンドポイントへリネーム
        - ２．バックアップから変更前のエンドポイント名でリストアする

        - 3．エンドポイント名を変更したRDSインスタンスを削除する

- インメモリデータベースを持つWebアプリケーションにおける構成例

    - Amazon Elasticacheを利用した場合の構成
        - マルチAzを利用できるRedisエンジンを前提として高可用性を実現
        - ※フェイルオーバーの動きはほぼ、RDSと同じ。
        - エンドポイントを変更する必要がないくらい

グローバルサービスにおける高可用性コンピューティング

ここまではVPC内での一般的な高可用性アーキテクチャを見てきたが、VPCリソースではない、グローバルサービスを利用した可用性を見る！

- Route53を利用したWebアプリケーションにおける構成例

    - DNSフェイルオーバーの構成例

- 参考

    - S3は静的ホスティングでSorryページを表示することができる
    - サイト名を独自ドメインで公開する場合は、バケット名をあわせるっ必要がある

- Route53によるマルチAZの構成例

    - EC2にEIPを割り当てておき、ルーティングポリシーに従って対象のEIPへAレコードで解決

- Route53によるマルチリージョンの構成例

    - 東京とシンガポールに同じ環境を設けて、Route53でルーティングする例

- CloudFrontを利用したWebアプリケーションにおける構成例

    - １．CloudFrontがまずリクエストを受付、CloudFrontにコンテンツがキャッシュされていればコンテンツを返し、なければバックエンドのオリジンへリクエストをルーティングする
    - ２．オリジンは複数登録でき、ルーティング先はパスパターンによって指定できる

2-4. ストレージにおける高可用性の実現

ストレージサービス

- EBS
    - AZ内で複製されているため単一障害にはならない
    - ただしAZ障害時にデータが消失してしまう可能性があるため、S3へのバックアップを適宜取得するケースはよくある！
- インスタンスストア（エフェメラルディスク）
    - 無料で利用できる高パフォーマンスなストレージサービス
- Amazon Elastic File System（EFS）
    - スケーラブルな共有ストレージ
    - 自動で複数のAZで冗長化されているため、ユーザ側で可用性を考慮する必要がありませんが、**EBSのようにスナップショットを取得することはできない**
- S3

    - デフォルトで３箇所のAZで自動的に複製されるためデータの耐久性は高い
    - 可用性という観点ではSLAが99.99%
- Amazon Glacier（Glacier）
    - アーカイブ目的

コンピューティングサービスにおけるストレージの選択

- EBSとインスタンスストアの関係
    - アプリケーションやデータの特性に応じて適切に選択する必要がある
    - （比較表が記載されているが割愛）
- EBSとインスタンスストアの利用手段
    - OSのルート領域として利用可能
        - EBS-backedインスタンス
        - Instance Store-Bakcedインスタンス
            - インスタンスの停止ができない。ターミネートするしかない
- 参考
    - インスタンスストアでは、EC2インスタンス停止時にデータが消失してしまうが、再起動では消失しない
    - これはEC2インスタンスの**停止、起動時の挙動として、同一のホストコンピュータでの起動を担保していないことに起因する！**
- S3をコンピューティングサービスとして利用
    - 静的ホスティングができるよ！っていう説明
    - メリット
        - スケールする必要がなく、アクセス集中につよい
        - 運用にかかるコストをへらすことができる
- EBSとEFSの違い

ー	ー	**EBS**	**EFS**
	接続	EC2インスタンスとネットワーク接続	EC2インスタンスとネットワーク接続
差	**接続台数**	１台のEC2インスタンスと接続可能	複数台のEC2インスタンスと接続可能
	データ永続性	永続化可能	永続化可能
差	**耐久性**	単一のAZで冗長化	複数のAZで冗長化
差	**拡張性**	手動でボリューム拡張可能	自動でボリューム拡張可能
	パフォーマンス	高速	高速
差	**料金体系**	ギガバイトあたりの重量課金（安）	ギガバイトあたりの重量課金（高）

- 試験対策
    - EBSとEFSの違いを覚えておく！

- 参考
    - EFSではスナップショットのようなバックアップ機能を提供していないため、別途バックアップを行う仕組みを検討する必要あり

各ストレージのバックアップ

- EBSのバックアップとリストア
    - スナップショットを用いる
    - S3に保管
    - 増分バックアップ
    - スナップショットはオンラインで取得することも可能ですが、データ整合性が求められるシステムではI/Oを停止してバックアップ
    - スナップショットはリージョン間でコピー可能なのでDRも可能
- 試験対策
    - EBSスナップショットには、取得開始時点のものが保管される
    - 取得開始から終了までに変更があったものはスナップショットに反映されない
- 参考
    - EBSはAZ単位で作成されるので、同一のAZにあるEC2にしかアタッチできない
- EC2のバックアップとリストア
    - EBSスナップショットを取ることでEC2のデータのバックアップを行えるが、殆どの場合AMIを用いている
- 参考
    - AMIは共有したり公開できる
    - AWSが提供しているAMIを使用するのが一般的だが、AWS Marketplaceにはベンダから多数のAMIが提供されている
- S3の機能
    - 注意点
        - S3に保存されたデータは複数のAZへ自動的に複製されることで高い耐久性が確保されているが、データの更新、削除は結果整合性モデルが採用されている
        - 古いデータが見えたり、消したのに画面上に表示されている時間がある
- S3のアクセス制御
    - 1-6ストレージサービスで説明した３つのアクセス制御

2-5. 演習問題

- Route53を利用してZone ApexレコードをELBに関連付けるにはどのレコードを指定すればよい？
    - Aレコード
- EBSとインスタンスストアの違いとして正しいもの

    - EBSはEC2インスタンス停止・起動後、データが消失しない
- ELBに関する説明として正しいもの
    - 複数のAZへ分散することができる
- 最も可用性が高いと考えられる構成はどれですか？
    - 略
- システムへの予期できないアクセス集中に有効なAWSサービスはどれ？
    - Amazon CloudFront
- EBSのスナップショットの特徴に関する説明として誤っているものはどれ
    - 単一のEC2インスタンスにしかアタッチできない
- AutoScalingを設定する際に、起動するEC2のインスタンスタイプを定義するものはどれですか
    - AutoScaling起動設定
- S3に関する説明で正しいもの
    - 結果整合性についての問題

第3章　AWSにおけるパフォーマンス

パフォーマンス効率のよいシステム構築の設計原則について説明

3-1. AWSにおけるパフォーマンスの考え方

AWSにおけるパフォーマンス効率の設計原則

- パフォーマンスを高めるための設計原則
    - 最新技術の導入
    - グローバルな環境
    - サーバーレスアーキテクチャの使用
        - S3の静的ホスティングやLambdaなどのイベント駆動型アプリ
    - 比較テストの実施
    - 適切な技術の利用

3-2. ネットワークサービスにおけるパフォーマンス

ネットワークにおけるパフォーマンスの考え方

地理的なレイテンシへの考慮

CloudFront

- CDNサービス
- 最も近い拠点からコンテンツを配信
- 接続ポイント
    - エッジロケーション
        - コンテンツをキャッシュしておくことができるキャッシュサーバー
    - リージョン別エッジキャッシュ
        - エッジロケーションよりも大容量のデータをキャッシュ可能なキャッシュサーバー

- 構成
    - １．閲覧者がコンテンツにアクセスすると、近いエッジロケーションにアクセス
    - ２．エッジロケーションにデータがない場合、リージョン別エッジキャッシュへアクセス
    - ３．リージョン別エッジキャッシュにデータがない場合、CloudFrontへアクセス
    - ４．CloudFrontはオリジンサーバからコンテンツを取得する
- 参考
    - CloudFrontにはエッジロケーションとリージョン別エッジキャッシュが存在するが、システム構成図を記載するときには、接続ポイントを記載しないことが一般的
- CloudFrontのユースケース
    - 静的コンテンツ配信：S3組み合わせ
    - 動的コンテンツ配信：EC2組み合わせ

Route53

- ルーティングポリシー
    - レイテンシベースルーティングが存在するのでこの機能を利用する
    - 地理的な近さではなく、低いレイテンシでアクセスできる方を優先する

プレイスメントグループ

- 単一のAZ内のEC2インスタンスを論理的にグルーピングしたもの
    - 異なるリージョン間で通信するよりより高速な通信が実現できるみたい
    - **クラスタコンピューティングを実装するような用途に最適！**

3-3. コンピューティングサービスにおけるパフォーマンス

EBS最適化インスタンス

- →理解済のため省略

Auto Scalingによる最適化の実現

- Amazon Cloud WatchとAutoScalingを組合せることで、システムのパフォーマンスを維持しつつ利用コストを押さえたシステムを構築することができる
- →理解済み
- 参考
    - CloudWatchとAutoScalingを利用することでパフォーマンスを効率的に管理できることを覚えておく

Lambdaのレイテンシ対策

- Lambdaはコンテナ上で動いているため、初回起動はコンテナ起動時間を考慮する必要がある
- その後一定期間は低レイテンシを保てるが、一定時間後、コンテナは停止する

- LambdaはVPCに配置することができるが、VPCエンドポイントでElastic Network Interface(ENI)を確立する必要があるため、VPC外にLambdaを構築
- レイテンシを低くするためには、SQSなどを用いて、ホットスタンバイの状態を維持する必要がある

3-4. ストレージにサービスにおけるパフォーマンス

EBS

- 各EBSボリュームタイプのスループットとIOPS
    - ５種類のボリュームタイプを用意
        - 汎用SSD
        - プロビジョンドIOPS SSD
        - スループット最適化HDD
        - コールドHDD
        - マグネティック
- パフォーマンス特性を記載した表（省略）
- 試験対策
    - EBSでパフォーマンスが低い場合の改善方法を押さえておく
- 試験対策
    - EBSの各ボリュームタイプの特徴とユースケースを押さえておく
- 参考
    - EBSボリュームタイプ選択時には、IOPS重視であればSSDタイプ、スループット重視で低コストで利用したい場合はHDDタイプを選択するとよい
- RAIDの設定
    - EBSでは標準的なRAIDはすべて使用できる！
    - →EBSに関するRAIDの話は初めて知った！
        - RAID0
        - RAID1
        - RAID5
        - RAID6

RAID構成	最小構成のディスク数	処理の特徴	備考
RAID0	2	ストライピング	高い耐障害性と高速な処理
RAID1	2	ミラーリング	非常に高い耐障害性
RAID5	3	ストライピング＋パリティデータ	AWSでは推奨されていない
RAID6	4	ストライピング＋パリティデータ	AWSでは推奨されていない

- IOPSがパリティデータ書き込み処理に消費されるため、AWSではRAID5、RAID6の使用を推奨していない
- 試験対策
    - 処理速度を早くすることができるRAID構成を覚えておく
- 参考
    - EBSでは標準的なRAIDは使用可能ですが、そのための条件として、EC2インスタンスで選択したOSがRAID設定をサポートしている必要がある

3-5. データベースサービスにおけるパフォーマンス

RDS

- RDSのユースケース
    - リレーショナルデータベースの機能を必要とするとき
- RDSのパフォーマンス
    - スペック（インスタンスやストレージ種類）とRDS固有の機能であるリードレプリカによって高速化可能

DynamoDB

Key-Value型のNoSQLデータベースサービス

- DynamoDBのユースケース
    - メリット
        - 高い信頼性
        - パフォーマンス
    - 可用性が求められ、大量のデータ処理が必要な場合に向いている
    - 例）
        - モバイルゲームやアドテクノロジー（広告配信システム）、IoTでのセンサーデータのデータベースとして利用
- DynamoDBとクライアントとの通信
    - DynamoDBとクライアントとのやり取りはAPIで行われる
    - インターフェース
        - AWS SDK
        - AWS Command Line Interface(AWS CLI)
- 試験対策
    - DynamoDBの特徴とユースケースを覚えておく
- DynamoDB Accelerator
    - Amazon DynamoDB Acccelerator(DAX)は、DynamoDB用インメモリキャッシュ
    - DynamoDBとDAXを一緒に利用すると、処理速度をあげられる
        - 構成イメージは以下を参照

        - https://dev.classmethod.jp/cloud/aws/dax-preview/

- 試験対策
    - DAXの特徴を覚えておく

Redshift

- ペタバイト規模のデータを標準SQLで分析可能なマネージド型のデータウェアハウスサービス
- SQLでアクセスできる点は一般的なRDBと同じだが
    - RDB：行指向型データベース
    - Redshift：列指向型データベース
        - という点で異なる
- Redshiftのユースケース
    - 大量のデータの保持、データウェアハウス、BIツールなどでの分析に向いている

ElastiCache

- キーバリュー型のNoSQLデータベースサービス
- インメモリデータベースのMemcachedとRedisをデータストアとして利用することができるが内容に違いがある
- 表の記載があるが割愛
- https://docs.aws.amazon.com/ja_jp/AmazonElastiCache/latest/mem-ug/SelectEngine.html
- 試験対策
    - ElastiCacheが利用されるユースケースを覚えておく
- 参考
    - RedisとMemcachedの違い詳細
    - https://docs.aws.amazon.com/ja_jp/AmazonElastiCache/latest/mem-ug/SelectEngine.html

データベースサービスの比較

- 割愛

3-6. 演習問題

- 単一AZ内に配置したEC2インスタンス間で低レイテンシの通信を行う必要がある、適切なサービスは？
    - プレイスメントグループ
- 世界中に会員がいるコンテンツ配信サービスがある。どの地域からアクセスしても低レイテンシで配信できる方法として適切なもの（特定の人へ）
    - CloudFrontを使用して、署名付きURLを発行する

- 汎用SSDボリュームがアタッチされた単一のEC2インスタンス上で実行されているアプリケーションがある。また、利用しているEC2インスタンスはEBS最適化インスタンスを有効化できるインスタンスタイプ。この環境でEBSのスループットが低下した場合に、パフォーマンスを向上させる手段として適切なものは？
    - 汎用SSDボリュームをプロビジョンドIOPSSSDに変更する
- アプリケーションで使用するデータベースとして、NoSQLデータベースをEC2上に構築することを検討している。アプリケーションの要件として、高スループットが必要です。NoSQLデータベースのパフォーマンスを発揮するのに最適なEBSストレージオプションはどれ
    - プロビジョンドIOPS SSDボリューム
- スケーラブルなWebアプリケーションを設計しており、ユーザーセッション情報を管理するサービスを検討している。次の打ち最適なサービスは
    - Amazon ElastiCache
- CPUとネットワークの使用率に基づいて、Webサーバーの数を自動的に増減できる様にしたいと考えている。この要件を満たすサービスはどれ
    - Auto Scaling
    - CloudWatch
- EC2インスタンスで実行されているWebアプリケーションがある。EC2インスタンスのCPU使用率が１００%に近づいている。負荷を軽減する適切な方法は？
    - Auto Scalingを利用して、負荷に応じたEC2インスタンス数に増減される様にする
- 現在、EC2インスタンスのアプリケーション負荷分散を行うためにAuto Scalingを検討している。CPU使用率を判断材料とし、スケーリングを実施したいと考えている。EC2のサーバーリソースをモニタリングするサービスとして正しいものは
    - Amazon CloudWatch
- プレイスメントグループとはどのようなサービスですか？
    - 単一AZ内のEC2インスタンス間の通信を広帯域幅・低レイテンシでやり取りできる

第4章　AWSにおけるセキュリティ設計

4-1. AWSにおけるセキュリティ設計の考え方

AWS責任共有モデルによるセキュリティ方針

- パブリッククラウドの利用を検討する際に、一番の懸念事項として取り上げられるのがクラウドの「セキュリティ」
- どの様にデータが配置されているか、ユーザから見えないため不安が生じる
- これに対してAWSはユーザとセキュリティ対策の責任境界を明確化することで、分担・協力しながらセキュリティを強化していく方針をとっている
- この考え方は「AWS責任共有モデル」と呼ばれている

AWSのセキュリティ責任・対策

- データセンター、物理ハードウェア
    - 物理的な場所を秘匿にしている
    - 監視カメラのセキュリティ対策
    - ストレージを破棄する際に決められたプロセスに従っている

- ネットワーク
    - DDoSなどの分散攻撃やIPなりすまし、パケット盗聴などの一般的なネットワークセキュリティに対しては、AWSが緩和、保護対策を行っている
- 仮想化インフラストラクチャ（ハイパーバイザー、ホストOS）
    - AWSがパッチ管理や、アクセス管理やログ監査などのセキュリティ運用を行っている

ユーザのセキュリティ責任・対策

- 一方、AWSを利用するユーザは、基本的にOSから上のレイヤーの管理・運用責任を持っている
- WellArchitectedFramework
    - アイデンティティ管理とアクセス管理
        - 最小権限の原則：必要最低限の権限をユーザに対して与える
        - AWS Identity and Acceess Management(IAM)による権限管理について重点的に説明 4-2
    - ネットワークセキュリティ
        - 4-3で説明
    - データの保護
        - 4-4で説明
    - セキュリティ監視
        - 4-5で説明
- 試験対策
    - 責任共有モデルにおけるAWSとユーザーのセキュリティに関する責任・分担の考え方は重要

4-2. アイデンティティ管理とアクセス管理

AWSにおけるアカウント

- AWSを利用するには最初にAWSアカウントを作成する必要がある
- 最初に作成するユーザを「ルートユーザ」と呼ぶ
    - フルアクセス権限を持っている
- ルートユーザは使うべきではない
- 参考
    - ルートユーザの通常利用を避けるだけでなく、強度の高いパスワード設定やアカウント自体へのログインを多要素認証（MFA:Multi-Factor Authentication）にするなどのセキュリティ強化策も必須

IAMユーザとIAMグループの概要

- IAMユーザー
    - AWSを操作するユーザをIAMサービスから作成
    - １つのAWSアカウントで5000ユーザまで作成可能
    - グループに所属させて、グループに対して権限を付与すると管理しやすい

- IAMグループ
    - IAMユーザを束ねるグループを作成することで、ユーザーや権限などの管理を統合的に行うことができる
    - １つのAWSアカウントで最大300グループを作成できる。IAMユーザあたり１０までのグループに所属することができる
    - IAMグループにIAMポリシーを割り当てることで権限管理を便利に行える
- 試験対策
    - IAMユーザあたりのグループ所属数には上限（１０）があるので注意が必要

IAMポリシーによるアクセス権限管理

- 「最小権限の原則」
- IAMポリシーの記述方法
    - IAMポリシーは、どのユーザーやグループが、どのAWSサービスまたはリソースに対して、どの操作を許可、または拒否するかをJSON形式で記述
    - デフォルト設定ではユーザは何もできないのでポリシーで明示的に許可を行う必要がある
- 試験対策
    - IAMユーザとIAMグループの作成直後は、権限が付与されていないので注意が必要
- 参考
    - IAMポリシー設定用のビジュアルエディタも提供されており、JSON構文を直接記述せずに容易にポリシー定義が行える
- IAMポリシーの種類
    - AWS管理ポリシー
        - AWSが管理する事前に定義されたポリシーのテンプレート
        - **Administrator Access**
            - 管理者相当のIAMユーザ
        - **PowerUser Access**
            - IAMサービスを除くその他すべてのアクセス権限保持
    - カスタマー管理ポリシー
        - AWSユーザが自身でカスタマイズすることが可能なポリシー
        - AWS管理ポリシーで標準提供されているものではセキュリティ用件を満たせない場合に使用する
    - インラインポリシー
        - IAMユーザ、IAMグループ、IAMロールに埋め込まれているポリシーに対して、ポリシー設定を個別に反映できる
- 試験対策
    - AWS管理ポリシーで提供されている代表的なポリシーは重要！覚えておく！

IAMによる認証方式

- IAMユーザ名とパスワード
    - EC2を起動する、ためにはコンソールへユーザ名とパスワードを入力してログインする必要がある
- 参考
    - 通常は、強度の高いパスワードのみを許可する、パスワードの有効期限を設定するなどのパスワードポリシーをIAMポリシーで設定することでログイン時のセキュリティを強化
- 他要素認証（MFA：Multi-Factor Authentication）
    - パスワード流出、なりすましを防ぐためMFAにより認証時のセキュリティ強化
- アクセスキーIDとシークレットアクセスキー
    - IAMユーザは「アクセスキーID」と「シークレットアクセスキー」のペアを作成することができる
    - これらの認証情報はコマンドラインで利用するAWS CLIまたはプログラム経由から利用するAWS SDKの認証情報として使うことができる
        - ただ！プログラムソースコードにキーを指定するのは流出の危険が高い！ので、IAMロールで認証するのが推奨！
- IAMロール
    - AWSサービスやアプリケーションに対してAWSの操作権限を付与する仕組み
    - IAMロールに対して直接IAMポリシーを付与することで、権限を管理
- 試験対策
    - アクセスキーIDとシークレットアクセスキーを用いた認証と、IAMロールによる認証方式の違いを押さえましょう

IAMによるIDフェデレーション

- IDフェデレーションとは、企業や組織などですでに導入されている認証の仕組みとAWSの認証を紐付け、シングルサインオンを実現する機能
- AWSのIAMに全社員の認証情報を登録する必要なく、企業ですでに導入されているLDAPなどの認証の仕組みと連携することでAWSサービスを簡単に利用可能
- SAML2.0を使用したIDフェデレーション
    - IAMでは、SAML(Security Assertion Markup Language)2.0やOpenIDConnectと互換性のある認証プロトコルをサポートしている
    - SAMLやOpenID Connecrtはシングルサインオンを簡単に実現するためのプロトコル
    - 認証したユーザ毎に一時的なアクセスキーを発行することで認証連携を実現する
    - 一時的なアクセスキーは、AWSのSecurity Token Service（STS）と呼ばれるサービスで管理・発行される
- 試験対策
    - SAML2.0によるAWSとのIDフェデレーション連携は重要

- Web IDフェデレーションによるソーシャルサービスとの連携
    - GoogleやFacebookなど、OpenID Connectをサポートしているソーシャルサービスとの IDフェデレーションすることで、AWSへのシングルサインオンが可能
    - これを「Web IDフェデレーション」と呼ぶ
    - Webアプリケーションやモバイルアプリからの認証では、Amazon Cognitoサービスを利用することで、各ソーシャルサービスとの認証連携が簡単に実現できる
- Directory ServiceによるMicrosoft ADとの認証連携
    - AWS Directory ServiceはAWSのクラウド内で管理されるマネージド型のMicrosoft ADです。
    - 主に３つのディレクトリタイプがある
        - Microsoft AD
        - Simple AD
        - AD Connector
            - AD Connectorは既存のActive Directory環境へ接続するためのプロキシサービス
- 試験対策
    - Directory ServiceのAD Connectorを利用することで、企業内の既存のActive Directoryとの認証連携を行うことができる

4-3. ネットワークセキュリティ

VPCにおけるネットワークセキュリティの考え方を説明

VPCによるネットワーク構成

- 標準的なネットワーク構成について図で説明

セキュリティグループとネットワークACLによるアクセス制御

- セキュリティグループ
    - EC2インスタンスなどに適用するファイアウォール機能
    - 詳細は理解済みのため割愛
- 試験対策
    - セキュリティグループの特徴を押さえましょう。特に、設定が即座に反映される点、またステートフルな制御が可能である点は重要
- 参考
    - 実際のセキュリティグループの適用では、インバウンド通信は最低限必要な通信のみを許可し、アウトバウンド通信は特別な用件がない限りすべて許可しておくケースが一般的
- ネットワークACL
    - VPC内に構成されたサブネットに対するファイアウォール機能
    - 詳細は理解済みのため割愛

- 試験対策
    - ネットワークACLはステートレスであるため、インバウンド通信とアウトバウンド通信の双方で通信制御を行うことが重要
- 参考
    - ネットワークACLのデフォルト状態では、
        - ルール番号１００で「すべてのトラフィック通信を許可」
        - ルール番号＊（最後の行）で「この行より上に記載したルールに一致しないすべての通信を拒否」
        - ルールは番号順に適用されるため、明示的な拒否の場合は１００より小さな数字を指定します
- セキュリティグループとネットワークACLを両方とも適用している場合の注意点
    - 双方のルールで許可されていないと拒否となるため注意が必要
- 試験対策
    - セキュリティグループとネットワークACLの双方で許可されていない場合は、全て拒否となることを押さえおく！

VPC内でのネットワークセキュリティ設定例

どの様に、セキュリティグループとネットワークACLを使い分けるかを説明！！

- 書籍を見たほうがよい。割愛

踏み台サーバーによるセキュアなリモートアクセス制御

- 踏み台サーバーを利用する（英語で「Bastion(砦)」）
- ケース
    - VPC内に配置されたEC2インスタンス上のサーバーにメンテナンスなどの運用のためにSSHやRDPを利用したりリモートからのアクセスを許可したいケースがある
    - VPC内のすべてのサーバーのSSHを常時許可しておくのは好ましくない
    - メンテナンス時の踏み台として、アクセス可能な踏み台サーバーを用意しておき、踏み台サーバーにログインしたあとで、各サーバーにアクセスすることでセキュリティリスクを低減する
    - このケースのポイント！
        - メンテナンスなどでSSHやRDPなどのリモートアクセスが必要となる場合のみ、踏み台サーバーを起動する（通常は停止しておく）
        - 踏み台サーバーはパブリックサブネットに設置し、パブリックIPを設定する
        - 踏み台サーバーに適用するセキュリティグループには、インターネットから踏み台サーバーに対するSSHやRDPアクセスの許可ルールを設定
        - その際に、インターネットからのアクセス元IPは不特定多数ではなく、メンテナンスに利用するPCなどの特定のIPやサブネットを指定することで、よりセキュアなアクセス制限が可能
        - 踏み台サーバー経由でアクセスされるEC2インスタンスには、必要に応じて「踏み台サーバーからのSSHやRDPのみ許可」をセキュリティグループに設定しておく

- 試験対策
    - 踏み台サーバー（Bastion）の役割と構成は重要
- 参考
    - 通常、Linuxサーバーへのリモートアクセスは SSH(22)
    - Windowsサーバーへのリモートアクセスは RDP(3389)

4-4. データの保護

暗号化によるデータの保護

- データの保護の基本的な考え方は「暗号化」です
- 一般的には以下の２種類の暗号化を検討する必要がある
- 通信の暗号化
    - ユーザとAWSの間を通る通信経路（インターネットなど）における盗聴からデータを保護
    - 具体的にはSSL/TLSなどの暗号化の仕組みをユーザとAWSの間で導入
- 試験対策
    - ELBやRDSなどのサービスとの通信時には、SSL/TLSによる通信の暗号化が利用できます
- 参考
    - SSL/TLS利用時にはSSL/TLS証明書が必要となります。
    - ユーザが証明書を持ち込むことも可能ですが、AWS Certificate Manager(ACM)を利用することでAWSサービスで利用する証明書の作成・管理が用意になります。
- 保管するデータ自体の暗号化
    - EBSやS3などに保管されるユーザーデータが悪意のある第三者からアクセスされることを防ぐ

データ暗号化の方式と場所

どこで暗号化をするか・鍵の管理はどこで行うかの２つが重要

- クライアントサイドでの暗号化（CSE：Client Side Encryption）
    - S3にアップロードする前に暗号化
    - EBSでは、OSが提供する暗号化機能を用いてからEBSに保管
- サーバーサイドでの暗号化（SSE：Server Side Encryption）
    - S3やEBS、Redshift、Glacierなどのサービスで暗号化
    - 暗号化に必要な鍵は次のいずれか
        - ユーザーが管理している鍵
        - S3で自動生成された鍵
        - 後述するAWS KMSなどのサービスと連携して生成された鍵
    - EBSではボリュームの作成時に暗号化設定が可能

    - ただし、既存のEBSボリュームを暗号化ボリュームに変更するには、以下の作業を行う必要がある
        - １．既存のボリュームのスナップショットを作成
        - ２．作成したスナップショットを複製する際に暗号化オプションを指定
        - ３．暗号化されたスナップショットからEBSボリュームを再作成
        - ４．EC2インスタンスから既存のEBSボリュームをデタッチ
        - ５．EC2インスタンスへ暗号化されたEBSボリュームをアタッチ

- 試験対策

    - **クライアントサイド暗号化とサーバサイド暗号化の特徴の違いを覚える、特にEBSとS3のユースケースは重要！！**

暗号化に必要な鍵の管理

データの暗号化・復号には、鍵の管理と保管が必要

鍵の管理とは、鍵自体の作成、有効化や無効化、定期的なローテーションなどを指す

管理方法は以下の３つに分けられる

（割愛）

AWS KMSとCloudHSMによる鍵の管理・保管

鍵の管理、保管を行うには、AWS KMSとCloudHSMのサービスを利用する

- AWS KMSによる鍵の管理・保管

    - AWS KMSはAWS上で鍵管理を提供するマネージド・サービス
    - 主に暗号化鍵の作成や有効・無効の管理、ローテーション、削除を行うことができる
    - 鍵自体はAWSに保管される
    - **AWS KMSと連携した暗号化処理が可能な主なAWSサービスは以下のものがある**
        - AWS SDKやCLIを利用したクライアントアプリケーション
        - S3、EBS、RDS、Redshiftなどのストレージやデータベースサービス

- 試験対策

    - ユーザの鍵をAWS KMSで管理することで、サーバーサイド暗号化やクライアントサイド暗号化が可能

- CloudHSMによる鍵の管理・保管

    - もう一つの手段として、CloudHSMサービスが有る
    - HSM（Hardware Security Module）は、AWSのデータセンター内に配置されるユーザ占有のハードウェアアプライアンス
    - AWSがHSMのアプリライアンス自体を管理し、ユーザーは自身だけが薗アプリライアンスに保存される鍵を管理することができます
    - HSM自体がユーザのVPC内に配置され、他のネットワークから隔離されることや、国際的なセキュリティ基準に準拠していることなどから、**セキュリティコンプライアンス用件が厳しい場合に適用します。**

    - **対応サービス**
        - Redshift
        - RDS for Oracle

- 試験対策

    - AWS KMSとCloudHSMで暗号化処理の連携がサポートされているAWSサービスは重要

4-5. セキュリティ監視

セキュリティ監視の関連サービス

- セキュリティ監視で押さえておくべき代表的なサービス以下の５つ

    - CloudTrail
    - VPCフローログ
    - Amazon CloudWatch Logs
    - AWS Config
    - Trusted Advisor

- CloudTrail

    - AWSアカウントで利用された操作をログとして記録するサービス
    - 特徴
        - AWSアカウントを取得した時点で有効化され、過去90日間のサービスに対する操作を表示
        - 取得されたログはデフォルトでS3に保存され、後述のCloudWatchLogsへの連携も可能
        - AWSの様々なサービスをサポート・連携してログ記録を行う
        - 記録内容は、サービスのAPIコール元、時間、送信元IPアドレス、呼び出しAPI、対象となるリソースなど
    - 例
        - 管理コンソール：AWSアカウントのルートユーザによるログイン履歴
        - EC2：インスタンスの操作履歴
        - KMS：KMSで管理されている鍵の使用や削除履歴

- 試験対策

    - CloudTrailでは様々なサービス（EC2やIAM、KMS、ELBなど）の操作ログを収集することで、セキュリティインシデントに関連する操作を監視できる点が重要

- VPCフローログ

    - VPC内のネットワークインターフェース間で行き来する通信の内容をキャプチャする機能
    - 意図しない通信が行われていないかなどの監視・監査に利用
    - VPCフローログを有効にすることで、後述するCloudWatch Logsを使用してログを保存・可視化できます。
    - 記録されるログ内容は、送信元IP・ポート・宛先IP・ポート・プロトコル番号などのネットワーク通信に関する詳細な内容

- CloudWatch Logs

    - CloudTrailやVPCフローログなど、AWSの様々なログを統合的に収集するサービス
    - AWSのログだけでなく、Linuxなどのサーバー OSにログを収集・通知するエージェントをインストールすることで、様々なログをCloud Watch Logsに送信することも可能
    - 収集したログに対して、たとえばあるログ内のWarning文字列をフィルタリングして監視し、CloudWatchのアラーム機能でアラートをメール通知するなどのセキュリティ監視対応が可能となる

- 試験対策
    - CloudWatch Logsによる各サービスからのログ収集とログ監視・通知の連携は重要

- AWS Config
    - AWSサービスで管理されているリソースの構成変更を追跡位するサービス
    - EC2インスタンスの作成や削除などの構成変更の履歴を取得できる
    - 意図しないリソースの変更をメールで通知したりできる

- 参考
    - Cloud Trailはユーザ（人）の操作（APIコール）を追跡するサービスですが、
    - AWS Configはリソース（モノ）の構成変更履歴に特化した監視サービス

- Trusted Advisor
    - AWSのベストプラクティスに基づいて、コスト最適化、セキュリティ、耐障害性、パフォーマンス、サービスの制限の５項目でユーザーのAWS利用状況をチェックし、改善スベき事項を推奨するサービス
    - セキュリティ診断観点
        - セキュリティグループ-開かれたポート
            - 特定のポートに対して、無制限アクセス（0.0.0.0/0）を許可しているセキュリティグループのルールをチェック
        - ルートアカウントのMFA
            - AWSアカウントのルートユーザでMFA（多要素認証）が有効されていない場合にアラート

AWS上でのセキュリティ侵入テスト

- 「AWS脆弱性/侵入テストリクエストフォーム」から申請し許可を得ることでユーザー側でこれらのテストを実施することができる

- テスト可能なリソース一覧
    - EC2
    - RDS
    - Amazon Aurora
    - CloutFront
    - API Gateway
    - AWS Lambda
    - Amazon Lightsail
    - DNS Zone Walking

- 試験対策
    - ユーザー側でAWSへの脆弱性スキャンや侵入テストを行うには、事前申請が必要
    - インスタンスの種類によってはテストできない点も押さえておく

4-6. 演習問題

- AWSの責任共有モデルについて、AWSの責任範囲は次のうちどれか
    - AWSが保持するデータセンターの物理的なアクセス制御
    - ストレージなどの物理ハードウェアの適切な処分・破棄
    - 仮想化インフラストラクチャ、ハイパーバイザーのアップデートやパッチ管理
- EC2インスタンス上のアプリケーションから、アクセスキーIDとシークレットアクセスキーを使わずに安全に他のAWSサービス（S3など）にアクセスしたいと考えています。適切な方法はどれか
    - IAMロールをEC2インスタンスに割り当てる
- 認証基盤とAWS IAMとの間で認証連携（IDフェデレーション）をする場合に使う技術として適切なもの
    - SAML2.0
- 複数のEC2インスタンスにWebサーバーを構築している。SSL証明書を利用してSSL通信を可能にする方式として適切なものはどれか
    - EC2インスタンスの手前にELBを作成し、SSL証明書をインストールしてSSLの通信設定を行う
- RDS for OracleとRedshiftでデータベース管理をしています。両者に保存されているデータをユーザーが作成した鍵を連携して暗号化するには、どのサービスで鍵を管理すればよいか
    - AWS ClloudHSM
    - AWS KMS
- EBS上に保存されるファイルを暗号化したいと考えている。適切な方法は？
    - ＊割愛＊
- S3上に保存されるファイルを暗号化したいと考えている。次の方式のうち、適切でないものはどれ？
    - S3との通信にSSL/TLSを利用する
- 会社のネットワークからインターネットを経由してVPC内のWebサーバーにSSHでリモートアクセスし、メンテナンスを行いたいと考えています。最も適切な方法は？
    - 踏み台サーバー（Bastion）をVPC内のパブリックサブネットに構成し、セキュリティグループで会社のネットワークからのSSHのみ許可する
- 10.1.3.2/32から、あるEC2インスタンスへのSSHアクセスを許可したいと考えています。次のうち正しい設定はどれ
    - セキュリティーグループのインバウンド通信で、アクセス元IPが10.1.3.2/32のSSHを許可する
- EC2インスタンスで可動しているLinuxサーバーのあるログを収集して監視したいと考えております。適切なサービスはどれか
    - Amazon CloudWatch Logs

第5章　AWSにおけるコスト最適化

クラウドサービスでは柔軟な料金でITリソースを活用できる反面、適切なコスト管理を行わなければ、想定外の料金になってしまう

コスト最適化について理解すること！

5-1. AWSにおけるコスト最適化の考え方

クラウドにおけるコスト最適化の重要性

- 不要なリソースにコストを消費しているケースが多々あるよていう説明

AWSにおけるコスト最適化の指針

コスト最適化の５つの設計原則

- 必要なリソースを必要なときに必要な分だけ利用する
- 全体的なコスト効果を測定する
- データセンター運用への投資を不要に
- 投資効果の要因分析
- マネージド・サービス活用により所有コストの削減

コスト最適化の４つのベストプラクティス

設計原則を具体的に実現していく手段として、以下の４つのベストプラクティス

- コスト効果が高いリソースの選定
- ITリソースの需要とAWSサービスの適切な供給によるコスト最適化
- 全体的なコストの管理
- 継続的なコスト最適化の活動

5-2. コスト効果が高いリソースの選定

AWSサービスの購入オプション

- EC2,RDS,Redshiftなど、主に利用した時間に対して課金されるコンピューティングリソースを利用するサービスでは、以下の３つの購入オプションが提供されている
    - オンデマンドインスタンス
    - リザーブドインスタンス
    - スポットインスタンス
- オンデマンドインスタンス
    - 標準ではオンデマンドインスタンスが適用されます
    - 一定のレートの料金で使用する
    - 利用期間の制約はない
    - 用途
        - 開発・テスト環境のような決められた時間帯（平日の日中など）しか使われないサーバ
        - キャンペーン時の一時的なWebサイトなど
- 参考
    - オンデマンドインスタンスの最新の料金一覧は、AWSのサイトに掲載されている
- 参考
    - EC2やEMRなどのサービスでは、秒単位で使用した分課金されます。

    - ただし、AWSのサービスによって課金単位は異なる

- リザーブドインスタンス

    - 予め決められた利用期間分を購入することで最大７５%オフの割引価格が適用される
    - 支払い方法
        - 全額前払い
        - 一部前払い
        - 前払いなし
        - の３種類のオプションから選択可能
    - 途中でキャンセルしても払い戻しなし
    - 用途
        - あらかじめ最低限の台数が必要なWebサーバーやアプリケーションサーバ
        - 常時起動しておく必要があるデータベース・サーバーなど
    - EC2ではリザーブドインスタンスがよくりようされ、
        - Standardタイプ
        - Convertibleタイプ　の２種類が提供されている
    - Standardタイプ
        - リージョンやアベイラビリティゾーンを指定してインスタンスを購入できる
        - 同じインスタンス構成であれば購入時に指定したRegion内、またはAZ内でインスタンスの配置変更が可能ですが、それ以外の場所に変更する場合は変更手続きが必要
    - Convertibleタイプ
        - あらかじめ指定したインスタンス構成に依存せず、柔軟に構成変更が可能
        - その分割引率はスタンダードより低く設定されている

- 試験対策

    - リザーブドインスタンスの主な用途、支払い方法と、２種類のタイプ（standardとconvertible）の違いを理解しましょう

- 参考

    - EC2などの期間購入のリザーブドインスタンスと同様の考え方で、Amazon CloudFrontやAmazon DynamoDBでは「リザーブドキャパシティ」と呼ばれる、あらかじめ決められた期間分のキャパシティを購入する割引オプションがある

- スポットインスタンス

    - 最大で９０%オフとなる、最も割引率が高い購入オプション
    - 希望の価格で入札する
    - スポット価格が高騰して入札額を上回ると、インスタンスが強制終了される
    - 用途
        - 処理が中断されても特に支障なく再実行が可能なシステムに適用できる
        - メディアエンコードのタスク処理
        - EMRのタスクノード
    - 2種類のオプションを必要に応じて選択できる
        - スポットブロック
            - あらかじめ決められた時間の起動を保証するオプション

        - 割引率は通常のスポットインスタンスより低いが、処理途中における想定外の終了を回避できる
    - スポットフリート
        - あらかじめ指定した性能キャパシティを満たすようにスポットインスタンスを構成するオプション
        - スポット価格の高騰で指定したスポットインスタンスが使用不可になっても、代替構成で全体の性能キャパシティを維持

- 試験対策
    - スポットインスタンスの主な使用用途と、強制終了される場合の条件（スポット価格が入札価格を上回る）は重要です
- 参考
    - 実際には、強制終了前に２分間の警告期間があり、この時間内にインスタンスの状態保存など、終了タスクの設定が可能

リソースの適切なサイジング

大小様々なリソースタイプを提供している

EC2に代表されるインスタンスとS3に代表されるストレージのサイジングについて説明

- 適切なインスタンスタイプの選定
    - EC2やRDS、Redshift、Amazon Elasticsearch Service（ES）などのサービスでは、様々なサイジングのインスタンスタイプの選択肢が提供されている
    - インスタンスタイプの構成
        - １．インスタンスファミリー
        - ２．インスタンス世代
        - ３．インスタンスサイズ
            - https://aws.amazon.com/jp/ec2/instance-types/
- 試験対策
    - 用途に合わせた適切なインスタンスファミリーの選択と、インスタンス世代・サイズの変更によりコスト最適化を実現できる
- 適切なストレージ選定
- S3のストレージクラス
    - １．スタンダード
        - データを複数箇所に保持
        - ナインイレブンの耐久性
    - ２．標準低頻度アクセス
        - ナインイレブンの耐久性
        - 格納コストはスタンダードと比較して安価
        - データの読み出し容量に対しても課金されるため、データへのアクセス頻度が低い場合に適している

    - ３．Glacier
        - アーカイブ目的で利用される
        - ナインイレブンの耐久性
        - データの取り出しに３～５時間かかる
        - バックアップ目的などめったにアクセスしない場合に適している
        - 復元方法は３種類から選択できる
            - 迅速：緊急時に少数のファイルを１～５分で復元可能だが、次の２つの方法と比較すると高コスト
            - 標準：取り出しに３～５時間程度かかる。標準的な方法
            - 大容量：取り出しに５～１２時間かかるが、大容量のデータを最も低コストで復元可能
    - ４．冗長化ストレージ
        - ９９．９９%耐久性
        - 上記３クラスから比較すると耐久性は低くなる
        - 用途は、Glacierから取り出したデータの一時置き場など、データを安価に一時格納する場合
    - ５．１ゾーン低頻度アクセス
        - １つのAZのみにデータを保存する
        - 複数AZにデータを複製するスタンダードクラスと比較すると、コストを約２０%削減できる
        - データへのアクセス頻度が低く、高い耐久性を必要とせず、かつ必要に応じてすぐに取り出したい場合に適している
    - 参考https://aws.amazon.com/jp/s3/storage-classes/

- 試験対策

    - S3の５種類のストレージクラスの用途・特徴に応じた使い分けを整理しておく

- EBSのストレージタイプ

    - １．汎用SSD
        - 費用対性能が高く、I/O性能も最大10，000IOPS
        - 用途
            - OSのブート領域
            - 中規模のデータベース用途
            - など、汎用的に使える
    - ２．プロビジョンドIOPS
        - 最もパフォーマンスの高いEBSで最大32，000IOPSまで指定できる
        - 用途
            - 大規模データベース
        - このタイプのみストレージ容量だけでなく、指定したIOPS数に対しても課金されるため注意が必要
    - ３．スループット最適化HDD
        - EMRによるログ分析など、主にファイルへのシーケンシャルアクセスが多い場合に高いスループットを提供するタイプ
        - 用途
            - ビッグデータ処理などに向いている
        - 価格もSSDの２タイプと比較して安価

    - ４．コールドHDD
        - 高いスループットが不要の場合はより低価格なコールドHDDのタイプも使用可能
        - ログファイルの保管など、高スループットを求められないバックアップ領域などが主な用途
    - ５．マグネティック
        - 旧世代のEBSのタイプ
        - 汎用SSDが登場する前はデフォルトのタイプでしたが現在も利用可能

- 試験対策

    - EBSの５種類のタイプの特徴と用途の違いを押さえておく
    - 特にSSDタイプとHDDタイプの用途の違いは重要

- 試験対策

    - プロビジョンドIOPS SSDでは、実際に利用しているストレージ容量に対する課金に加え、IOPS数にも課金されるため、コスト面で注意が必要！

5-3. 需要と供給のマッチングによるコスト最適化

需要の変化に応じたITリソースの供給

使いたいと思ったときに簡単にリソースを調達可能

あまり使われていないのにAWSのサービスを利用し続けると無駄な料金がかさむ

以下、需要と供給のマッチングを実現するサービスについて紹介します

ELBとAuto Scalingによる柔軟なリソース供給

- Elasticity(伸縮性)という言葉がある
- 伸縮性を生かしてAWSでは通信トラフィックなどの需要の増減に応じて柔軟にリソースを増減させることが可能
- AWSではAuto Scalingというサービスを利用することでこの伸縮性をうまく実現している
- ELB+Auto Scalingの組み合わせ方法

    - AutoScalingは通常、様々な通信トラフィックをEC2インスタンスに負荷分散するためにELBとともに利用され、負荷状況に合わせてEC2インスタンスの増減を制御する
    - 通信トラフィックなど、負荷状況に合わせた柔軟な制御を行うため、Amazon CloudWatchの監視メトリクスも合わせて使用する
    - ELBとAutoScalingを組み合わせた拡張方式の詳細は「2-3.コンピューティングにおける可用性の実現」を参照！

SQSやKinesisによるバッファリング

バッファリングにより処理を非同期化することで需要と供給を管理する方法もある

必ずしも同期的に処理する必要がない場合には、バッファリングなどの仕組みにより需要（負荷）を一時的に蓄積しておき、処理する側の必要に応じた間隔で非同期的に処理を行うという方法もある

- Amazon Simple Queue Service(SQS)
    - メッセージキューのマネージド・サービス
    - アプリケーション間でメッセージをキューイングすることで疎結合なアーキテクチャを実現
    - よく使われるSQSの機能・パターンは以下の３つ
        - ロングポーリング（Long Polling）
            - 通常、SQSキューに対して受信者がメッセージ取得要求を送ると、キューが空の場合でもEmptyメッセージが返送されます
            - この方式をショートポーリングと呼ぶ
            - SQSはリクエスト単位で課金されるため、空振りが多い場合にはこの方式は非効率的
            - ロングポーリングではキューが空の場合、メッセージが取得できるまで待つ時間を設定することでメッセージ取得要求の数をへらすことができる
        - 可視性タイムアウト
            - ある受信者がメッセージを取得した場合に、他の受信者にはそのメッセージをある一定期間（デフォルトで30秒間）見せない様にすることで、処理の重複を防ぐ、またリクエストをへらすことが可能
        - スポットインスタンスとの組み合わせ
            - 可視性タイムアウトとスポットインスタンスを組み合わせることで更にコスト効果が高まる
            - メッセージ受信者のアプリケーションにスポットインスタンスを使用していて、仮に強制終了しても、可視性タイムアウト時間（デフォルトで３０秒）を超過すると他の受信者がメッセージを取得できる様になるため、処理の再実行が可能となる

- Amazon Kinesis（Kinesis）
    - 主にストリーミングデータの収集・処理・リアルタイム分析に利用される
    - サーバレスかつ処理ボリュームに応じて拡張するため、AutoScaling+EC2などを自前で構築するよりも簡単かつ低コストでストリームデータのバッファリング処理が可能になる
    - SQSとの主な違いは複数のコンシューマが同時に同じメッセージを取得して処理できる点
    - Kinesisは以下の３サービスから構成される
        - Amazon Kinesis Data Streams
            - ストリーミングデータをほぼリアルタイムで保存することができ、登録されたデータはEMRやAmazon Lambdaなどのサービス上に構築された独自アプリケーションで処理することができる
            - また流れてくる大容量のデータを効率的に処理するために、「シャード」と呼ばれる単位でデータを分割して並列処理を行うことができます
        - Amazon Kinesis Data Firehose（Kinesis Data Firehose）
            - 独自にアプリケーションを構築することなく、ストリームデータをAmazonの各サービスに簡単に保存できるサービス
            - たとえば、ストリーミングされるデータを分析用データとしてS3やRedshiftに蓄積するケースなどえ利用される
        - Amazon Kinesis Data Analytics
            - ストリーミングデータに対してSQLクエリを実行し、リアルタイム分析を行うサービス
            - SQLクエリを利用できるため、例えば１分ごとのストリーミングデータの合計値や平均値などを簡単に計算できる

- 試験対策
    - SQS、Kinesisのバッファリング処理の特徴と利用用途の違いは重要
- 試験対策
    - Kinesisはリアルタイムに処理する
    - Kinesisの３つのサービスの違いを抑えておきましょう

5-4. コストの管理

継続的にコスト最適化をしていくために必要なコスト管理方法や可視化・レポーティングについて説明

AWSにおけるコスト管理

AWSではアカウント毎にサービスを利用した分のコストが発生する

- 主なコスト管理のタスク

タスク	タスク概要	主なAWSサービス
コストの請求・支払い	複数のAWSアカウントをまとめて支払い管理する	Consolidated Biling(一括請求)
コスト状況の詳細把握	コストとリソース使用量の詳細をレポート・分析する	AWS Cost And Usage Report（コストと使用状況レポート）
コストの可視化・傾向分析	コストの可視化や深掘り分析・将来予測を行う	AWS Cost Explorer
過剰利用の監視	予算策定や予算超過のアラートを管理する	AWS Budgets（予算設定）
コスト最適化の検討	コスト最適化の余地があるリソースを分析・改善する	AWS Trusted Advisor

→上記各サービスに、「請求ダッシュボード」からアクセスできることを確認した。

Consolidated Bilingによる支払い管理

Consolidated Bilingは複数のAWSアカウントに対する請求を１つに統合し、まとめて支払いできる機能

「AWS Organizations」と呼ばれる、複数のAWSアカウントを一元管理するサービスの１機能として提供されている

→AWS Organizationsというサービスが存在することを確認した

> AWS Organizations により、AWS クラウド内の複数のアカウントにポリシーベースのコントロールを一括して適用できます。 AWS アカウントをすべて 1 つの組織内に統合し、すべての AWS アカウントを個別の組織単位に整理できます。

- この機能での利点
    - 複数のAWSアカウントのコストが１つの請求書に統合されることに寄る支払業務の効率化

    - 各AWSアカウント（たとえば業務部門・IT部門や開発環境、本番環境）の使用状況を統合的に把握可能
    - 複数のAWSアカウントの使用量を統合することで、ボリュームディスカウントによるコスト削減が可能

- マスターアカウントに対し、メンバーアカウントの請求が統合できる
- 試験対策
    - Consolidated Bilingにおけるアカウントの種類（マスターアカウントとメンバーアカウント）と、利用した場合の利点を抑えておきましょう
- 参考
    - 通常、請求先のマスターアカウントは支払い専用のAWSのアカウントとして作成します！←そうなんだ
    - メンバーアカウントは、開発環境やテスト環境などの環境ごとや、AWSを利用する部門ごとなど、利用用途に応じて作成されることが多い

AWS Cost and Usage Reportによる詳細レポート・分析

AWSサービス毎のリソース使用状況とコストを１時間ごとや日時で収集し、CSV形式でS3へ保存、またはRedshiftに格納したり、Amazon QuickSightで分析・可視化したりできる

Consolidated Builingを使用している場合は、マスターアカウントのレポートにメンバーアカウントの情報が表示されます

- 取得可能な情報は以下の様なものがある
    - bill/PrayerAccountId:支払いアカウントのAWSアカウントID
    - product/ProductName:利用しているAWSサービスの名前
    - product/region:AWSサービスのRegion
    - lineItem/UsageAmount:指定した期間に発生したリソース使用量

定期的に監視分析することで、リザーブドインスタンスの見直しといったコスト最適化のための計画を検討することができる

- 参考
    - 以前はDetailed Biling Report（請求明細レポート）が利用されていましたが、現在は利用は推奨されていない

Cost Explorerによるコストの可視化と傾向分析

戦術のAWS Cost and Usage Reportのデータをグラフィカルに可視化したり検索したりできるサービス

例えばEC2のリソース使用量やコストを時系列にグラフ化し、詳細に分析したい場合は、様々な条件や期間を指定してフィルタリング・検索が可能

- 可視化
    - 最大で過去１２ヶ月分のコストデータをグラフ表示できる
        - Monthly costs by service:サービス毎の月次コスト推移
        - Monthly costs linked account:AWSアカウントごとの月次コスト推移

        - Daily costs:日次コスト推移
        - Monthly EC2 running costs and usage:月次EC2使用時間・コストの推移
        - RI Utilization:リザーブドインスタンスのコスト推移やオンデマンドインスタンスの場合と比較したコスト削減額など

- 分析
    - サービス、AWSアカウント、Region、AZ、インスタンスタイプ、タグなどの条件でフィルタリング表示できます
- 予測
    - 可視化や分析だけでなく、過去の使用量のトレンドから今後３ヶ月間のコストを予測する
- 参考
    - 実際の現場では、部門ごとのAWSリソース使用量に応じたコストを配賦したいケースがある。Cost Explorerで部門を示すタグを定義しておくと、タグでフィルタリングすることで部門ごとのコストを集計することができる

予算設定による利用超過の監視

利用予算を設定することができる

- 予算額の設定
    - コストやリソース使用量など監視する対象を選び、月次や四半期などのタイミングを指定して監視する予算額や使用量を設定
- 通知の設定
    - 設定した予算額を超過しそうな場合、または超過した場合にCloudWatchアラームやSNSトピック、メール等による通知が可能

Trusted Advisorによるコスト最適化

ここまでは、コストの詳細把握、可視化と分析・予測・監視などに使えるAWSサービスを説明した

コスト管理に関連する最後のタスクはコスト最適化

コスト最適化のヒントを推奨するサービスとして、AWS Trusted Advisorが提供されている！

Trusted AdvisorはAWSのベストプラクティスに基づいて、

- コスト最適化
- セキュリティ
- 耐障害性
- パフォーマンス
- サービスの制限

５項目でユーザのAWS利用状況をチェックし、改善スベき事項を推奨するサービス

例えば以下のような観点でコスト診断が可能

- 使用率の低いEC2インスタンス

- EC2リザーブドインスタンスの最適化
- 関連付けられていないElastic IPアドレス
- 試験対策
    - Trusted Advisorでチェック可能な５角項目（コスト最適化、セキュリティ、耐障害性、パフォーマンス、サービスの制限）は重要なので覚えておく
    - 特にコストの最適化の観点では、EIPが課金される条件とコスト削減の方法は重要！
- 参考
    - **Trusted Advisorのコスト最適化機能を利用するには、ビジネスまたはエンタープライズレベルのAWSサポートプランを選択している必要がある**

5-5. 演習問題

- EC2上でアプリケーションを運用している。このアプリケーションは２４時間３６５日かつ長時間かどうしていて、常に一定のアクセス負荷がある。コスト最適化するために、EC2のドの料金オプションを選択すべきでしょうか
    - リザーブドインスタンス
- 複数のEC2上でアプリケーションを運用している。このアプリケーションはログ分析の分散バッチ処理を行っており、EC2が障害などえ停止した場合でも単独で再実行できる様に設計されている。コスト最適化のために、EC2のどの料金オプションを選択すべき？
    - スポットインスタンス
- EC2インスタンスサイズがc4.8xlargeで、I/Oが多い分析アプリケーションが稼働している。CPU使用率は殆ど使われていない場合、コスト最適化しつつ、分析アプリケーションのパフォーマンスを向上させるにはどの様にしたらよいでしょうか
    - より小さいサイズでI/O性能が高い最新のc5インスタンスに変更する
        - **→インスタンスファミリーの詳細についてある程度覚えておかないと回答できない**
- 500TBのログ・ファイルを保持している。不定期にログファイルにアクセスする必要が有り、通常は30分以内にログデータを取得したいと考えています。次のS3ストレージクラスのうち、最もコスト効果が高いものはどれ
    - 標準低頻度アクセス
- IOPSに対する課金があるEBSストレージタイプは次のうちどれ
    - プロビジョンドIOPS SSD
- EC2上のデータベース・サーバーで読み込み・書き込みが頻繁に発生している場合、性能を一定に保つために最もコスト効果が高いEBSストレージタイプはどれ
    - プロビジョンドIOPS SSD
        - データベース・サーバーなどに使用される！
- SQSの利用用途として適切なものはどれ
    - アップロードされたメディアファイルを複数のEC2インスタンスで非同期処理する場合のジョブ制御
- ストリーミングデータの処理をAWSで行いたいと考えています。複数の独自アプリケーションから同時に同じストリーミングデータを読み込み、処理する必要がある場合に利用するAWSサービスとして適切なものはどれ
    - Amazon Kinesis Data Streams
        - ほぼリアルタイムでストリームデータを保存することができ、登録されたデータはEMRやAmazon Lambdaなどのサービス上に構築された独自アプリケーションで処理するこ

とができる
    - また、流れてくる大量データを効率的に処理するために「シャード」と呼ばれる単位でデータを分割して並列処理を行うことができる
- Consolidated billing（一括請求）で統合可能なAWSアカウントはどれ
    - メンバーアカウント
    - マスターアカウント
- Trusted Advisorでチェック可能な項目はどれ
    - コスト最適化
    - セキュリティ
    - 耐障害性
    - パフォーマンス
    - ※あと、「サービスの制限」がある

第6章　AWSにおける運用管理

6-1. AWSにおける運用の考え方

システム構築やアプリケーション開発を迅速に行うことは重要ですが、その後のリリース作業や運用管理を迅速・正確に行うことも重要

本設ではAWS上での効率的な運用方法を説明

AWSにおける運用の考え方

AWSにはクラウド上で高い運用性を実現するために以下のような設計原則があります

- コードによるオペレーション実行
- 定期的に、小規模で、もとに戻すことができる変更を行う
- 運用手順を見直す
- 障害発生を想定する
- 運用の失敗を元に改善する

6-2. 作業の自動化

コードによるインフラストラクチャ構成管理

- Infrastructure as Code(IaC)という考え方
    - アプリケーション開発で実践されているコード管理・作業自動化という手法をインフラストラクチャ構築にも導入して、インフラストラクチャ構築作業内容をコードで記述・管理できるようにする
- AWSでは定義ファイルをコーディングし、コードを実行することでリソースをプロビジョニングすることができる
- 参考
    - バージョン管理とは、主にアプリケーション開発やドキュメント作成時などの履歴を管理すること

CloudFormationのコード管理

AWSにおいてIaCを実現したのが、「AWS CloudFormation」

「テンプレート」と呼ばれるプロビジョニングのための定義ファイルを用意し実行することで定義内容に従ってAWSリソースを構築することができる

テンプレートはJSONまたはYAMLフォーマットで記述することができる

- JSONとYAMLのフォーマット
    - 書き方（フォーマット）の説明。理解済みのため割愛

CloudFormationの設定

詳細な設定方法はWebページ参照

https://aws.amazon.com/jp/cloudformation/aws-cloudformation-templates/

※割愛

- 試験対策
    - CloudFormationテンプレートの記述方法を覚えておく！
- 参考
    - CloudFormation設定の詳細な書式はAWSのWebページで確認できる

自動化できるサービスとその機能

プロビジョニングやデプロイ、構成管理の作業が効率化できる主なサービス

サービス名	説明
AWS CodeCommit	Gitベースのリポジトリを利用したソース管理サービス
AWS CodePipeline	アプリケーションのビルド、テスト、デプロイまでの処理手順を定義し、実行できるサービス
Code Deploy	アプリケーションのデプロイを自動化するサービス
Elastic Beanstalk	Webアプリケーションやサービスをサーバーにデプロイでき、また、実行環境の管理も行うことができるサービス
OpsWorks	構築手順どおりにサーバー構築作業を自動化することができる構成管理サービス
Cloud Formation	AWS内のすべてのインフラストラクチャリソースを自動でプロビジョニングできるサービス

6-3. 運用に関するその他のサービス

モニタリング

AWSではシステムをモニタリングできるサービスとして、「Amazon CloudWatch」がある

詳細は1-10.「運用管理サービス」を参照

サポート

- サービスの稼働状況監視
    - AWS Service Health Dashboard
        - ユーザに関係のない情報も提供されているので、以下も提供
    - AWS Personal Service Health Dashboard
        - ユーザに関係のある情報を提供
            - →AWSコンソールのベルマークをクリックして、すべてのアラートを表示、から画面表示できることを確認した
- AWSサポート
    - 問い合わせ窓口
    - AWSのエンジニアがユーザーをサポートするサービス
- サポートプランの比較https://aws.amazon.com/jp/premiumsupport/compare-plans/
    - リンクから確認
    - 書籍内容は割愛する

6-4. 演習問題

- CloudFormationのテンプレート作成に利用できるフォーマットはどれか
    - JSON
    - YAML
- JSONフォーマットの書式として適切なものはどれか
    - 割愛
- CloudFormationの主要な設定項目のうち、テンプレート実行時に利用できる値を指定するための項目はどれか
    - Parameters
- CloudFormationのテンプレートを作成したいと考えています。ネットワークACLのエントリ（ルール）作成に関する記述をする場合、Resoures項目のプロパティで設定する内容として誤っているものは
    - Address
        - Addressは存在しない
        - CloudFormationテンプレートの書き方は再度復習が必要
- 次のうち、デプロイとプロビジョニングの両方ができるサービスはどれか
    - AWS Elastic Beanstalk
    - AWS OpsWorks
- 次のうち、EC2などのAWSリソースを監視することができるサービスはどれ
    - AWS CloudWatch
- AWSサポートプランのうち、「非常事態：業務が危険にさらされており、アプリケーションの重要機能が利用できない」という場合の問い合わせに対して、１５分以内の目標時間を掲げているサポートプランはどれ
    - エンタープライズ

索引

著者紹介

奥付